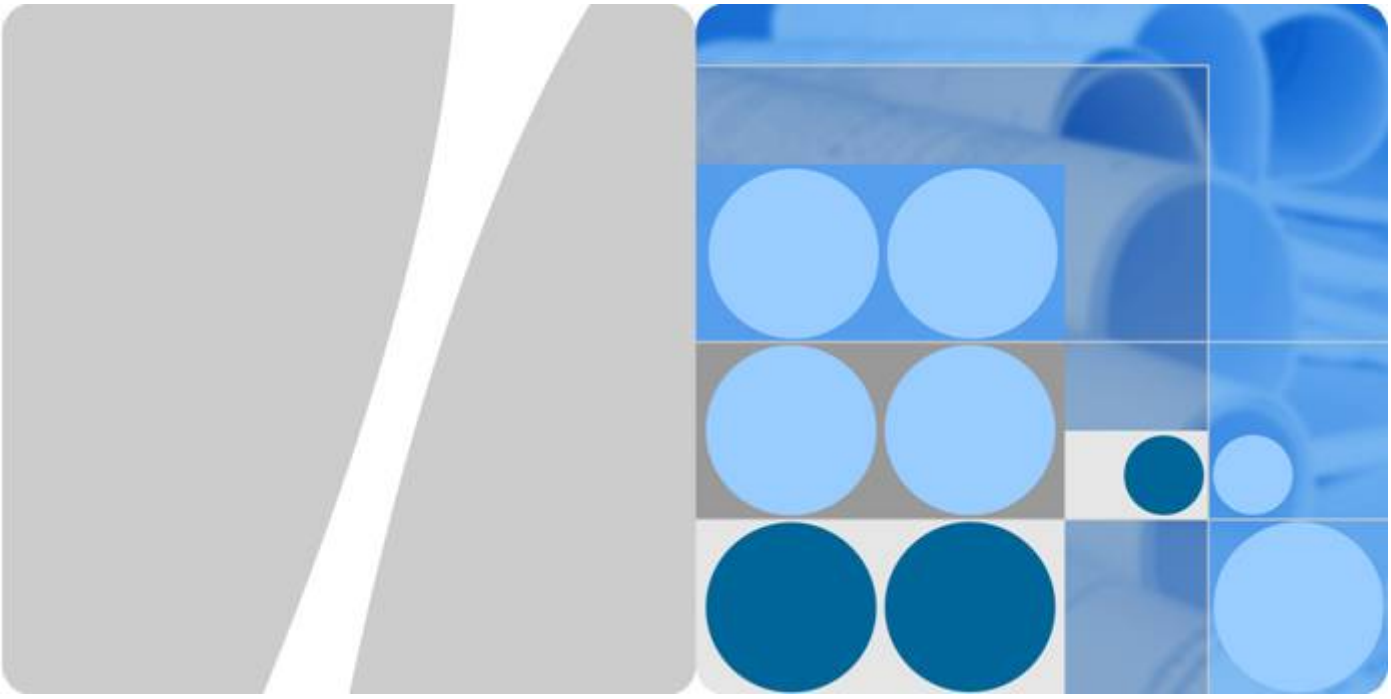


Object Storage Service

API Reference (OBS)

Date 2022-04-15

Contents

1 Before You Start

1.1 Overview

Welcome to the *Object Storage Service API Reference*. Object Storage Service (OBS) provides massive, secure, reliable, and cost-effective data storage capabilities for users to store data of any type and size. It is suitable for scenarios such as enterprise backup/archiving, video on demand (VoD), and video surveillance.

This document describes how to use application programming interfaces (APIs) to perform operations on OBS, such as creating, modifying, and deleting bucket, as well as uploading, downloading, and deleting objects. For details about all supported operations, see **API Overview**.

Before calling OBS APIs, ensure that you have fully understood relevant concepts. For details, see **Basic Concepts**.

1.2 API Calling

OBS provides Representational State Transfer (REST) APIs, allowing you to use HTTP or HTTPS requests to call them. For details, see **Calling APIs**.

1.3 Endpoints

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. For the endpoints of services, see **Regions and Endpoints**.

1.4 Basic Concepts

Basic Concepts Related to OBS APIs

- Account

  Your cloud service account. The account has full access permissions for all the resources and cloud services that are subscribed under the account. The account can also reset user passwords and grant permissions to users.

- User

  You can create users under a domain on Identity and Access Management (IAM), and authorize the users with permissions required for accessing cloud services. Each IAM user has its own identity credentials (password and access keys).

  On the **My Credentials** page on the console, you can view the domain ID and user ID, you can also manage the access keys of the domain and IAM users.

  Access keys of the domain and its IAM users are required for authentication when calling APIs.

- Bucket

  A bucket is a container where objects are stored. It is the top namespace in OBS. Each object must reside in a bucket. For example, if the object named **picture.jpg** is stored in the **photo** bucket, you can use the following URL to access the object: **http://photo.obs.*region*.*example*.com/picture.jpg**.

- Objects

  An object is a basic data unit on OBS. A bucket can store multiple objects, and OBS does not distinguish between object types. Objects are serialized in OBS. An object may be a text, a video, or any other types of files. In OBS, the size of a file can range from 0 bytes to 48.8 TB. However, when an object is uploaded through the **PutObject** operation, it cannot exceed the maximum size of 5 GB. Use the multipart upload method, if the object size is larger than 5 GB.

- Region

  A region is a geographic area in which cloud resources are deployed. Availability zones (AZs) in the same region can communicate with each other over an intranet, while AZs in different regions are isolated from each other. Deploying cloud resources in different regions can better suit certain user requirements or comply with local laws or regulations.

  Each bucket in OBS must reside in a region. You can specify the region when creating the bucket. Once a bucket is created, its region cannot be changed. Select the most appropriate region for a bucket based on the location, cost, and regulatory compliance requirements. For details about regions, see **Endpoints**.

2 API Overview

API Operations on Buckets

Table 2-1 API operations on buckets

Operation	Description
Listing Buckets	Queries the list of buckets created by the user.
Creating a Bucket	Creates a bucket. You can add different request headers to specify the region and permission control policy.
Listing Objects in a Bucket	Lists objects in a bucket. You can add different request headers to obtain objects that match the specified prefix, identifier, and other requirements.
Obtaining Bucket Metadata	Checks whether the bucket metadata exists. You can query the information about the bucket region, OBS version number, and CORS configuration.
Obtaining Bucket Location	Obtains the bucket region information.
Deleting Buckets	Deletes a specified bucket. Before deleting a bucket, ensure that the bucket is empty.

API Operations on Advanced Bucket Settings

Table 2-2 API operations on advanced bucket settings

Operation	Description
Configuring a Bucket Policy	Creates or modifies a bucket policy. If the specified bucket already has a policy, the policy in the request will overwrite the existing one.

Operation	Description
Obtaining Bucket Policy Information	Obtains the policy information of a specified bucket.
Deleting a Bucket Policy	Deletes the policy of a specified bucket.
Configuring a Bucket ACL	Configures the ACL of a specified bucket. You can control the read and write permissions of a bucket through ACL settings.
Obtaining Bucket ACL Information	Obtains the ACL information of a specified bucket.
Configuring Logging for a Bucket	Enables or disables the log management function of a bucket. When this function is enabled, a log record is generated for each operation on a bucket. Multiple log records are packed into a log file, which will be saved in a specified location.
Obtaining a Bucket Logging Configuration	Obtains the logging configuration of the current bucket.
Configuring Bucket Lifecycle Rules	Configures rules to automatically delete objects in a bucket.
Obtaining Bucket Lifecycle Configuration	Obtains the lifecycle rules configured for a specified bucket.
Deleting Lifecycle Rules	Deletes the lifecycle configuration of a bucket.
Configuring Versioning for a Bucket	Enables or disables versioning for a bucket. When this function is enabled, objects of different versions can be retrieved and restored, and data can be quickly restored in case of accidental operations or application faults.
Obtaining Bucket Versioning Status	Obtains the versioning status of a specified bucket.
Configuring Event Notification for a Bucket	Configures the event notification for a bucket to ensure that the bucket owner is notified about events occur on the bucket in a secure and timely manner.
Obtaining the Event Notification Configuration of a Bucket	Obtains the notification configuration of a bucket.
Configuring Tags for a Bucket	Adds a tag to an existing bucket. After tags are added to a bucket, all data records generated by the requests for this bucket will take the same tags. Thus, reports can be categorized for detailed cost analysis.

Operation	Description
Obtaining Bucket Tags	Obtains the tags of a specified bucket.
Deleting Tags	Deletes the tags of a specified bucket.
Configuring Bucket Storage Quota	Sets the bucket space quota to limit the maximum storage capacity of the bucket.
Querying Bucket Storage Quota	Obtains the bucket space quota.
Querying Information About Used Space in a Bucket	Obtains the number of objects in a bucket and the space occupied by the objects.
Configuring Bucket Inventories	Configures an inventory rule for a bucket. OBS provides bucket inventories to facilitate your management of objects in a bucket. You can configure bucket inventories to periodically list objects in a bucket. During the listing of objects, object metadata is saved in a CSV file, which is uploaded to the specified bucket.
Obtaining Bucket Inventories	Obtains an inventory rule of a specified bucket.
Listing Bucket Inventories	Obtains all inventory rules of a specified bucket.
Deleting Bucket Inventories	Deletes an inventory rule of a specified bucket.

API Operations for Static Website Hosting

Table 2-3 API Operations for Static Website Hosting

Operation	Description
Configuring Static Website Hosting for a Bucket	Creates or updates the website hosting configuration of a bucket. OBS allows you to store static web page resources such as HTML web pages, flash files, videos, and audios in a bucket. When a client accesses these resources from the website endpoint of the bucket, the browser can directly resolve and present the resources to the client.
Obtaining the Static Website Hosting Configuration of a Bucket	Obtains the website hosting configuration of a bucket.
Deleting the Static Website Hosting Configuration of a Bucket	Deletes the website hosting configuration of a bucket.

Operation	Description
Configuring Bucket CORS	Configures the cross-origin resource sharing (CORS) configuration of a bucket. OBS allows static web page resources to be stored in buckets. The buckets can be used as website resources. A website hosted by OBS can respond to cross-domain requests from another website only after the CORS rule is configured.
Obtaining the CORS Configuration of a Bucket	Obtains the CORS configuration of a bucket.
Deleting the CORS Configuration of a Bucket	Deletes the CORS configuration of a bucket.
OPTIONS Bucket	Checks whether the client has the permission to perform operations on the server. It is usually performed before the cross-domain access.
OPTIONS Object	Checks whether the client has the permission to perform operations on the server. It is usually performed before the cross-domain access.

API Operations on Objects

Table 2-4 API operations on objects

Operation	Description
Uploading Objects - PUT	Uploads simple objects to a specified bucket.
Uploading Objects - POST	Uploads objects to a specified bucket based on tables.
Copying Objects	Creates a copy for an existing object in OBS.
Downloading Objects	Downloads objects.
Querying Object Metadata	Obtains the object metadata. Information such as object expiration time, version number, and CORS configuration is the object metadata.
Deleting an Object	Deletes a specified object. You can also carry the versionId field to delete the specified object version.

Operation	Description
Deleting Objects	Deletes a batch of objects from a bucket permanently. Objects deleted in this way cannot be recovered.
Configuring an Object ACL	Configures the ACL of a specified object. You can control the read and write permissions of objects through ACL settings.
Obtaining Object ACL Configuration	Obtains the ACL configuration of a specified object.

API Operations for Multipart Tasks

Table 2-5 API operations for multipart tasks

Operation	Description
Listing Initialized Multipart Tasks in a Bucket	Queries all the multipart upload tasks that have not been merged or canceled in a bucket.
Initializing a Multipart Task	Initiates a multipart upload task, and obtains the globally unique multipart upload task ID for subsequent operations, such as uploading, merging, and listing parts.
Multipart Upload	Uploads parts for a specific multipart task.
Uploading a Part of an Object - Copy	Copies an object or a part of the object as a part of a multipart task.
Listing Uploaded Parts of an Object	Queries information about all parts of a multipart task.
Merging Parts into a Complete Object	Merges the specified parts into a complete object.
Canceling a Multipart Upload Task	Cancels a multipart upload task.

3 Calling APIs

3.1 Constructing a Request

This section describes the structure of a REST API request.

Request URI

OBS uses URI to locate specific buckets, objects, and their parameters. Use URIs when you want to operate resources.

The following provides a common URI format. The parameters in square brackets [ ] are optional.

protocol://[bucket.]domain[:port][/object][?param]

Table 3-1 URI parameters

Parameter	Description	Mandatory
protocol	Protocol used for sending requests, which can be either HTTP or HTTPS. HTTPS is a protocol that ensures secure access to resources. OBS supports both HTTP and HTTPS.	Yes
bucket	Resource path of a bucket, identifying only one bucket in OBS	No
domain	Domain name or IP address of the server for saving resources	Yes

Parameter	Description	Mandatory
port	Port enabled for protocols used for sending requests. The value varies with software server deployment. If no port number is specified, the protocol uses the default value. Each transmission protocol has its default port number. For example, HTTP uses port number **80** and HTTPS uses port number **443** by default. In OBS, HTTP port number is **80** and that of HTTPS is **443**.	No
object	An object path used in the request	No
param	A specific resource contained by a bucket or object. Default value of this parameter indicates that the bucket or object itself is obtained.	No

NOTICE

All API requests except those for the bucket list must contain the bucket name. Based on the DNS resolution performance and reliability, OBS requires that the bucket name must be placed in front of the **domain** when a request carrying a bucket name is constructed to form a third-level domain name, also mentioned as virtual hosting access domain name.

For example, you have a bucket named **test-bucket** in the **a1** region, and you want to access the ACL of an object named **test-object** in the bucket. The correct URL is **https://test-bucket.obs.a1.example.com/test-object?acl**.

Request Method

HTTP methods, which are also called operations or actions, specify the type of operations that you are requesting.

Table 3-2 HTTP request methods supported by the OBS

Method	Description
GET	Requests the server to return a specific resource, for example, a bucket list or object.
PUT	Requests the server to update a specific resource, for example, creating a bucket or uploading an object.
POST	Requests the server to add a resource or perform a special operation, for example, part uploading or merging.
DELETE	Requests the server to delete specified resources, for example, an object.

Method	Description
HEAD	Requests the server to return the digest of a specific resource, for example, object metadata.
OPTIONS	The request server checks whether the user has the operation permission for a resource. The CORS needs to be configured for the bucket.

Request Headers

Refers to optional and additional request fields, for example a field required by a specific URI or HTTP method. For details about the fields of common request headers, see **Table 3-3**.

Table 3-3 Common request headers

Header	Description	Mandatory
Authorization	Signature information contained in a request message Type: string No default value. Conditional: optional for anonymous requests and required for other requests.	Conditionally required
Content-Length	The message length (excluding headers) defined in RFC 2616 Type: string No default value. Conditional: required for PUT requests and those requests that load XML content.	Conditionally required
Content-Type	The content type of the requested resource, for example, **text/plain** Type: string No default value.	No
Date	Time when a request is initiated, for example, **Wed, 27 Jun 2018 13:39:15 +0000**. Type: string No default value. Conditional: optional for anonymous requests or those requests containing header **x-obs-date**, required for other requests.	Conditionally required

Header	Description	Mandatory
Host	The host address. For example, **bucketname.obs.*region*.example.com**. Type: string No default value.	Yes

(Optional) Request Body

A request body is generally sent in a structured format (for example, JSON or XML). It corresponds to **Content-Type** in the request header and is used to transfer content other than the request header. If the request body contains Chinese characters, these characters must be coded in UTF-8.

The request body varies according to the APIs. Certain APIs do not require the request body, such as the GET and DELETE APIs.

Sending a Request

There are two methods to initiate requests based on the constructed request messages:

- cURL

  cURL is a command-line tool used to perform URL operations and transmit information. cURL acts as an HTTP client that can send HTTP requests to the server and receive response messages. cURL is applicable to API debugging. For more information about cURL, visit **https://curl.haxx.se/**. cURL cannot calculate signatures. When cURL is used, only anonymous public OBS resources can be accessed.

- Coding

  You can use code to make API calls, and to assemble, send, and process request messages.

3.2 Authentication

3.2.1 User Signature Authentication

OBS signs a request using AK/SK. When a client is sending a request to OBS, the message header must contain the SK, request time, request type, and other information of the signature.

- AK: access key ID, which is a unique identifier associated with a secret access key (SK). The AK and SK are used together to obtain an encrypted signature for a request. Format example: **HCY8BGCN1YM5ZWYOK1MH**
- SK: secret access key, which is used together with the AK to sign requests, identify a request sender, and prevent the request from being modified. Format example: **9zYwf1uabSQY0JTnFqbUqG7vcfqYBaTdXde2GUcq**

A user can obtain the AK and SK from IAM. For details, see **Obtaining Access Keys (AK/SK)**.

OBS provides three signature calculation methods based on application scenarios: **Authentication of Signature in a Header**, **Authentication of Signature in a URL**, and **Authentication of Signature Carried in the Table Uploaded Through a Browser**.

Table 3-4 shows the user signature verification process in which a signature is carried in a header. For details about the parameters and code examples of authentication of signature in a header, see **Authentication of Signature in a Header**.

Table 3-4 Signature calculation and verification procedure

Procedure		Example
Signature calculation	1. Construct an HTTP message.	PUT /object HTTP/1.1 Host: bucket.obs.region.example.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913
	2. Calculate **StringToSign** based on the signature rule.	StringToSign = HTTP-Verb + "\n" + Content-MD5 + "\n" + Content-Type + "\n" + Date + "\n" + CanonicalizedHeaders + CanonicalizedResource
	3. Prepare the AK and SK.	AK: ****** SK: ******
	4. Calculate **Signature**.	Signature = Base64(HMAC-SHA1(**SecretAccessKeyID**, UTF-8-Encoding-Of(**StringToSign**)))
	5. Add a signature header and send the request to OBS.	PUT /object HTTP/1.1 Host: bucket.obs.region.example.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913 Authorization: OBS **AccessKeyID**:**Signature**
Signature authentication	6. Receive the HTTP message.	PUT /object HTTP/1.1 Host: bucket.obs.region.example.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913 Authorization: OBS **AccessKeyID**:**Signature**

Procedure		Example
	7. Obtain the SK based on the AK in the request.	Obtain the AK from the authorization header and obtain the SK of the user from IAM.
	8. Calculate **StringToSign** based on the signature rule.	StringToSign = HTTP-Verb + "\n" + Content-MD5 + "\n" + Content-Type + "\n" + Date + "\n" + CanonicalizedHeaders + CanonicalizedResource
	9. Calculate **Signature**.	Signature = Base64(HMAC-SHA1(**SecretAccessKeyID**, UTF-8-Encoding-Of(**StringToSign**)))
	10. Authenticate the signature.	Verify that the value of **Signature** in the authorization header is the same as the value of **Signature** calculated by the server. If the two values are the same, the signature verification is successful. If the two values are different, the signature verification fails.

3.2.2 Authentication of Signature in a Header

For all API operations, the most common identity authentication is to carry signatures in headers.

In the header, the signature is carried in the authorization header field of the HTTP message. The format of the message header is as follows:

```
Authorization: OBS AccessKeyID:signature
```

The signature algorithm process is as follows:

1. Construct the request character string (StringToSign).

2. Perform UTF-8 encoding on the result obtained from the preceding step.

3. Use the SK to perform the HMAC-SHA1 signature calculation on the result obtained from step 2.

4. Perform Base64 encoding on the result of step 3 to obtain the signature.

The StringToSign is constructed according to the following rules. **Table 3-5** describes the parameters.

```
StringToSign =
   HTTP-Verb + "\n" +
   Content-MD5 + "\n" +
   Content-Type + "\n" +
   Date + "\n" +
   CanonicalizedHeaders + CanonicalizedResource
```

Table 3-5 Parameters required for constructing a StringToSign

Parameter	Description
HTTP-Verb	Indicates an HTTP request method supported by the OBS REST API. The value can be an HTTP verb such as **PUT**, **GET**, or **DELETE**.
Content-MD5	Base64-encoded 128-bit MD5 digest of the message according to RFC 1864. This parameter can be empty. For details, see **Table 3-10** and the algorithm examples below the table.
Content-Type	Specifies the message type, for example, **text/plain**. If a request does not contain this header field, this parameter is deemed as an empty string. For details, see **Table 3-6**.
Date	Time when a request is initiated. This parameter uses the RFC 1123 time format. If the deviation between the time specified by this parameter and the server time is over 15 minutes, the server returns error 403. This parameter is an empty string when the **x-obs-date** is specified. For details, see **Table 3-10**. If an operation (for example, obtaining an object content) is temporarily authorized, this parameter is not required.

Parameter	Description
CanonicalizedHeaders	OBS request header field in an HTTP request header, referring to header fields started with **x-obs-**, for example, **x-obs-date**, **x-obs-acl**, and **x-obs-meta-***. 1. All characters of keywords in a request header field must be converted to lowercase letters (content values must be case sensitive). If a request contains multiple header fields, these fields should be organized by keyword in the alphabetical order from a to z. 2. If multiple header fields in a request have the same prefix, combine the header fields into one. For example, **x-obs-meta-name:name1** and **x-obs-meta-name:name2** should be reorganized into **x-obs-meta-name:name1,name2**. Use comma to separate the values. 3. Keywords in the request header field cannot contain non-ASCII or unrecognizable characters, which are also not advisable for values in the request header field. If the two types of characters are necessary, they should be encoded and decoded on the client side. Either URL encoding or Base64 encoding is acceptable, but the server does not perform decoding. 4. Delete meaningless spaces and tabs in a header field. For example, **x-obs-meta-name: name** (with a meaningless space in the front of **name**) must be changed to **x-obs-meta-name:name**. 5. Each header field occupies a separate line. See **Table 3-8**.

Parameter	Description
CanonicalizedResource	Indicates the OBS resource specified by an HTTP request. This parameter is constructed as follows: <Bucket name + Object name> + [Subresource 1] + [Subresource 2] + ... 1. Bucket name and object name, for example, ***/bucket/object***. If no object name is specified, for example, ***/bucket/***, the entire bucket is listed. If no bucket name is specified either, the value of this field is **/**. 2. If a subresource (such as **?acl** and **?logging**) exists, the subresource must be added. OBS supports a variety of sub-resources, including acl, attname, cors, delete, deletebucket, inventory, length, lifecycle, location, logging, metadata, modify, name, notification, partNumber, policy, position, quota, replication, response-cache-control, response-content-disposition, response-content-encoding, response-content-language, response-content-type, response-expires, storagePolicy, storageinfo, tagging, torrent, uploadId, uploads, versionId, versioning, versions, website, and x-obs-security-token. 3. If there are multiple subresources, sort them in the alphabetical order from a to z, and use **&** to combine the subresources. **NOTE** ● A subresource is unique. Do not add subresources with the same keyword (for example, **key=value1&key=value2**) in the same request URL. In this case, signature is computed only based on the first subresource, and only the value of the first subresource takes effect on the actual service. ● Using the **GetObject** API as an example, assume there is a bucket named **bucket-test** and an object named **object-test** in the bucket, and the object version is **xxx**. When obtaining the object, you need to rewrite Content-Type to **text/plain**. Then, the **CanonicalizedResource** calculated by the signature is **/bucket-test/object-test?response-content-type=text/plain&versionId=xxx**.

The following tables provide some examples of generating StringToSign.

Table 3-6 Obtaining an object

Request Header	StringToSign
GET /object.txt HTTP/1.1 Host: bucket.obs.*region*.example.com Date: Sat, 12 Oct 2015 08:12:38 GMT	GET \n \n \n Sat, 12 Oct 2015 08:12:38 GMT\n /bucket/object.txt

Table 3-7 Using temporary AK/SK and security token to upload objects

Request Header	StringToSign
PUT /object.txt HTTP/1.1 User-Agent: curl/7.15.5 Host: bucket.obs.*region*.example.com x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT x-obs-security-token: YwkaRTbdY8g7q.... content-type: text/plain Content-Length: 5913339	PUT\n \n text/plain\n \n x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT\n x-obs-security-token:YwkaRTbdY8g7q....\n /bucket/object.txt

NOTE

For details about how to obtain a temporary AK/SK pair and security token, see **Obtaining a Temporary AK/SK Pair**.

Table 3-8 An object upload request containing header fields

Request Header	StringToSign
PUT /object.txt HTTP/1.1 User-Agent: curl/7.15.5 Host: bucket.obs.*region*.example.com Date: Mon, 14 Oct 2015 12:08:34 GMT x-obs-acl: public-read content-type: text/plain Content-Length: 5913339	PUT\n \n text/plain\n Mon, 14 Oct 2015 12:08:34 GMT\n x-obs-acl:public-read\n /bucket/object.txt

Table 3-9 Obtaining an object ACL

Request Header	StringToSign
GET /object.txt?acl HTTP/1.1 Host: bucket.obs.*region*.example.com Date: Sat, 12 Oct 2015 08:12:38 GMT	GET \n \n \n Sat, 12 Oct 2015 08:12:38 GMT\n /bucket/object.txt?acl

Table 3-10 An object upload request carrying the Content-MD5 header

Request Header	StringToSign
PUT /object.txt HTTP/1.1 Host: bucket.obs.*region*.example.com x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT Content-MD5: l5pU0r4+sgO9Emgl1KMQUg== Content-Length: 5913339	PUT\n l5pU0r4+sgO9Emgl1KMQUg==\n \n \n x-obs-date:Tue, 15 Oct 2015 07:20:09 GMT\n /bucket/object.txt

Content-MD5 Algorithm in Java

```
import java.security.MessageDigest;
import sun.misc.BASE64Encoder;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;

public class Md5{
    public static void main(String[] args) {
        try {
            String exampleString = "blog";
            MessageDigest messageDigest = MessageDigest.getInstance("MD5");
            BASE64Encoder encoder = new BASE64Encoder();
            String contentMd5 = encoder.encode(messageDigest.digest(exampleString.getBytes("utf-8")));
            System.out.println("Content-MD5:" + contentMd5);
        } catch (NoSuchAlgorithmException | UnsupportedEncodingException e)
        {
            e.printStackTrace();
        }
    }
}
```

The signature is generated as follows based on the StringToSign and SK. The hash-based message authentication code algorithm (HMAC algorithm) is used to generate the signature.

```
Signature = Base64( HMAC-SHA1( YourSecretAccessKeyID, UTF-8-Encoding-Of( StringToSign ) ) )
```

For example, to create a private bucket named **newbucketname2** in a region, the client request format is as follows:

```
PUT / HTTP/1.1
Host: newbucketname2.obs.region.example.com
```

```
Content-Length: length
Date: Fri, 06 Jul 2018 03:45:51 GMT
x-obs-acl:private
Authorization: OBS UDSIAMSTUBTEST000254:ydH8ffpcbS6YpeOMcEZfn0wE90c=

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
    <Location>region</Location>
</CreateBucketConfiguration>
```

Signature Algorithm in Java

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Base64;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.TreeMap;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

import org.omg.CosNaming.IstringHelper;


public class SignDemo {

    private static final String SIGN_SEP = "\n";

    private static final String OBS_PREFIX = "x-obs-";

    private static final String DEFAULT_ENCODING = "UTF-8";

    private static final List<String> SUB_RESOURCES = Collections.unmodifiableList(Arrays.asList(
            "CDNNotifyConfiguration", "acl", "attname",  "cors", "delete",
            "deletebucket", "inventory", "length", "lifecycle", "location", "logging",
            "metadata", "modify", "name", "notification", "partNumber", "policy", "position", "quota",
            "replication", "response-cache-control", "response-content-disposition",
            "response-content-encoding", "response-content-language", "response-content-type", "response-
expires",
            "storagePolicy", "storageinfo", "tagging", "torrent", "truncate",
            "uploadId", "uploads", "versionId", "versioning", "versions", "website",
            "x-obs-security-token"));

    private String ak;

    private String sk;

     public String urlEncode(String input) throws UnsupportedEncodingException
    {
        return URLEncoder.encode(input, DEFAULT_ENCODING)
        .replaceAll("%7E", "~") //for browser
        .replaceAll("%2F", "/")
        .replaceAll("%20", "+");
    }

    private String join(List<?> items, String delimiter)
    {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < items.size(); i++)
        {
    String item = items.get(i).toString();
            sb.append(item);
```

```
            if (i < items.size() - 1)
            {
                sb.append(delimiter);
            }
        }
        return sb.toString();
    }

    private boolean isValid(String input) {
        return input != null && !input.equals("");
    }

    public String hamcSha1(String input) throws NoSuchAlgorithmException, InvalidKeyException,
UnsupportedEncodingException {
        SecretKeySpec signingKey = new SecretKeySpec(this.sk.getBytes(DEFAULT_ENCODING), "HmacSHA1");
        Mac mac = Mac.getInstance("HmacSHA1");
        mac.init(signingKey);
        return Base64.getEncoder().encodeToString(mac.doFinal(input.getBytes(DEFAULT_ENCODING)));
    }

    private String stringToSign(String httpMethod, Map<String, String[]> headers, Map<String, String>
queries,
            String bucketName, String objectName) throws Exception{
        String contentMd5 = "";
        String contentType = "";
        String date = "";

        TreeMap<String, String> canonicalizedHeaders = new TreeMap<String, String>();

        String key;
        List<String> temp = new ArrayList<String>();
        for(Map.Entry<String, String[]> entry : headers.entrySet()) {
            key = entry.getKey();
            if(key == null || entry.getValue() == null || entry.getValue().length == 0) {
                continue;
            }

            key = key.trim().toLowerCase(Locale.ENGLISH);
            if(key.equals("content-md5")) {
                contentMd5 = entry.getValue()[0];
                continue;
            }

            if(key.equals("content-type")) {
                contentType = entry.getValue()[0];
                continue;
            }

            if(key.equals("date")) {
                date = entry.getValue()[0];
                continue;
            }

            if(key.startsWith(OBS_PREFIX)) {

                for(String value : entry.getValue()) {
                    if(value != null) {
                        temp.add(value.trim());
                    }
                }
                canonicalizedHeaders.put(key, this.join(temp, ","));
                temp.clear();
            }
        }

        if(canonicalizedHeaders.containsKey("x-obs-date")) {
            date = "";
        }
```

```
        // handle method/content-md5/content-type/date
        StringBuilder stringToSign = new StringBuilder();
        stringToSign.append(httpMethod).append(SIGN_SEP)
            .append(contentMd5).append(SIGN_SEP)
            .append(contentType).append(SIGN_SEP)
            .append(date).append(SIGN_SEP);

        // handle canonicalizedHeaders
        for(Map.Entry<String, String> entry : canonicalizedHeaders.entrySet()) {
            stringToSign.append(entry.getKey()).append(":").append(entry.getValue()).append(SIGN_SEP);
        }

        // handle CanonicalizedResource
        stringToSign.append("/");
        if(this.isValid(bucketName)) {
            stringToSign.append(bucketName).append("/");
            if(this.isValid(objectName)) {
                stringToSign.append(this.urlEncode(objectName));
            }
        }

        TreeMap<String, String> canonicalizedResource = new TreeMap<String, String>();
        for(Map.Entry<String, String> entry : queries.entrySet()) {
            key = entry.getKey();
            if(key == null) {
                continue;
            }

            if(SUB_RESOURCES.contains(key)) {
                canonicalizedResource.put(key, entry.getValue());
            }
        }

        if(canonicalizedResource.size() > 0) {
            stringToSign.append("?");
            for(Map.Entry<String, String> entry : canonicalizedResource.entrySet()) {
                stringToSign.append(entry.getKey());
                if(this.isValid(entry.getValue())) {
                    stringToSign.append("=").append(entry.getValue());
                }
            }
        }

//        System.out.println(String.format("StringToSign:%s%s", SIGN_SEP, stringToSign.toString()));

        return stringToSign.toString();
    }

    public String headerSignature(String httpMethod, Map<String, String[]> headers, Map<String, String>
queries,
            String bucketName, String objectName) throws Exception {

        //1. stringToSign
        String stringToSign = this.stringToSign(httpMethod, headers, queries, bucketName, objectName);

        //2. signature
        return String.format("OBS %s:%s", this.ak, this.hamcSha1(stringToSign));
    }


    public String querySignature(String httpMethod, Map<String, String[]> headers, Map<String, String>
queries,
            String bucketName, String objectName, long expires) throws Exception {
        if(headers.containsKey("x-obs-date")) {
            headers.put("x-obs-date", new String[] {String.valueOf(expires)});
        }else {
            headers.put("date", new String[] {String.valueOf(expires)});
        }
```

```
        //1. stringToSign
        String stringToSign = this.stringToSign(httpMethod, headers, queries, bucketName, objectName);

        //2. signature
        return this.urlEncode(this.hamcSha1(stringToSign));
    }

    public static void main(String[] args) throws Exception {

        SignDemo demo = new SignDemo();
        demo.ak = "<your-access-key-id>";
        demo.sk = "<your-secret-key-id>";

        String bucketName = "bucket-test";
        String objectName = "hello.jpg";
        Map<String, String[]> headers = new HashMap<String, String[]>();
        headers.put("date", new String[] {"Sat, 12 Oct 2015 08:12:38 GMT"});
        headers.put("x-obs-acl", new String[] {"public-read"});
        headers.put("x-obs-meta-key1", new String[] {"value1"});
        headers.put("x-obs-meta-key2", new String[] {"value2", "value3"});
        Map<String, String> queries = new HashMap<String, String>();
        queries.put("acl", null);

        System.out.println(demo.headerSignature("PUT", headers, queries, bucketName, objectName));
    }

}
```

The calculation result of the signature is as follows (it varies with the execution time): YdH8ffpcbS6YpeOMcEZfn0wE90c=

Signature Algorithm in Python

```
import sys
import hashlib
import hmac
import binascii
from datetime import datetime
IS_PYTHON2 = sys.version_info.major == 2 or sys.version < '3'

yourSecretAccessKeyID = '275hSvB6EEOorBNsMDEfOaICQnilYaPZhXUaSK64'
httpMethod = "PUT"
contentType = "application/xml"
# "date" is the time when the request was actually generated
date = datetime.utcnow().strftime('%a, %d %b %Y %H:%M:%S GMT')
canonicalizedHeaders = "x-obs-acl:private"
CanonicalizedResource = "/newbucketname2"
canonical_string = httpMethod + "\n" + "\n" + contentType + "\n" + date + "\n" + canonicalizedHeaders +
CanonicalizedResource
if IS_PYTHON2:
    hashed = hmac.new(yourSecretAccessKeyID, canonical_string, hashlib.sha1)
    encode_canonical = binascii.b2a_base64(hashed.digest())[:-1]
else:
    hashed = hmac.new(yourSecretAccessKeyID.encode('UTF-8'),
canonical_string.encode('UTF-8'),hashlib.sha1)
    encode_canonical = binascii.b2a_base64(hashed.digest())[:-1].decode('UTF-8')
print encode_canonical
```

The calculation result of the signature is as follows (it varies with the execution time): YdH8ffpcbS6YpeOMcEZfn0wE90c=

3.2.3 Authentication of Signature in a URL

OBS allows users to construct a URL for a specific operation. The URL contains information such as the user's AK, signature, validity period, and resources. Any user who obtains the URL can perform the operation. After receiving the request,

the OBS deems that the operation is performed by the user who issues the URL. For example, if the URL of an object download request carries signature information is constructed, the user who obtains the URL can download the object, but the URL is valid only within the expiration time specified by the parameter of **Expires**. The URL that carries the signature is used to allow others to use the pre-issued URL for identity authentication when the SK is not provided, and perform the predefined operation.

The format of the message containing the signature request in the URL is as follows:

```
GET /ObjectKey?AccessKeyId=AccessKeyID&Expires=ExpiresValue&Signature=signature HTTP/1.1
Host: bucketname.obs.region.example.com
```

The format of the message containing the temporary AK/SK and security token in the URL for downloading objects is as follows:

```
GET /ObjectKey?AccessKeyId=AccessKeyID&Expires=ExpiresValue&Signature=signature&x-obs-security-token=securitytoken HTTP/1.1
Host: bucketname.obs.region.example.com
```

Table 3-11 describes the parameters.

Table 3-11 Request parameters

Parameter	Description	Mandatory
AccessKeyId	AK information of the issuer. OBS determines the identity of the issuer based on the AK and considers that the URL is accessed by the issuer. Type: string	Yes
Expires	Indicates when the temporarily authorized URL expires, in seconds. The time must be in Coordinated Universal Time (UTC) format and later than 00:00:00 on January 1, 1970. Type: string	Yes
Signature	The signature generated using the SK and the expiration time. Type: string	Yes
x-obs-security-token	During temporary authentication, the temporary AK/SK and security token must be used at the same time and the **x-obs-security-token** field must be added to the request header.	No

The signature computing process is as follows:

1. Construct the StringToSign.
2. Perform UTF-8 encoding on the result obtained from the preceding step.

3. Use the SK to perform the HMAC-SHA1 signature calculation on the result obtained from step 2.

4. Perform Base64 encoding on the result obtained from step 3.

5. Perform URL encoding on the result of step 4 to obtain the signature.

The StringToSign is constructed according to the following rules. **Table 3-12** describes the parameters.

```
StringToSign =
    HTTP-Verb + "\n" +
    Content-MD5 + "\n" +
    Content-Type + "\n" +
    Expires + "\n" +
    CanonicalizedHeaders +    CanonicalizedResource;
```

Table 3-12 Parameters required for constructing a StringToSign

Parameter	Description
HTTP-Verb	Indicates an HTTP request method supported by the OBS REST API. The value can be an HTTP verb such as **PUT**, **GET**, or **DELETE**.
Content-MD5	Base64-encoded 128-bit MD5 digest of the message according to RFC 1864. This parameter can be empty.
Content-Type	Specifies the message type, for example, **text/plain**. If a request does not contain this header field, this parameter is deemed as an empty string.
Expires	Expiration time of the temporary authorization, that is, the value of parameter **Expires** in the request message: **ExpiresValue**.

Parameter	Description
CanonicalizedHeaders	OBS request header field in an HTTP request header, referring to header fields started with **x-obs-**, for example, **x-obs-date**, **x-obs-acl**, and **x-obs-meta-***. 1. All characters of keywords in the header field must be converted to lower-case letters. If a request contains multiple header fields, these fields should be organized by keywords in the alphabetical order from a to z. 2. If multiple header fields in a request have the same prefix, combine the header fields into one. For example, **x-obs-meta-name:name1** and **x-obs-meta-name:name2** should be reorganized into **x-obs-meta-name:name1,name2**. Use comma to separate the values. 3. Keywords in the request header field cannot contain non-ASCII or unrecognizable characters, which are also not advisable for values in the request header field. If the two types of characters are necessary, they should be encoded and decoded on the client side. Either URL encoding or Base64 encoding is acceptable, but the server does not perform decoding. 4. Delete meaningless spaces and tabs in a header field. For example, **x-obs-meta-name: name** (with a meaningless space in the front of **name**) must be changed to **x-obs-meta-name:name**. 5. Each header field occupies a separate line.

Parameter	Description
CanonicalizedResource	Indicates the OBS resource specified by an HTTP request. This parameter is constructed as follows: <Bucket name + Object name> + [Subresource 1] + [Subresource 2] + ... 1. Bucket name and object name, for example, ***/bucket/object***. If no object name is specified, for example, ***/bucket/***, the entire bucket is listed. If no bucket name is specified either, the value of this field is **/**. 2. If a subresource (such as **?acl** and **?logging**) exists, the subresource must be added. OBS supports a variety of sub-resources, including acl, attname, cors, delete, deletebucket, inventory, length, lifecycle, location, logging, metadata, modify, name, notification, partNumber, policy, position, quota, replication, response-cache-control, response-content-disposition, response-content-encoding, response-content-language, response-content-type, response-expires, storagePolicy, storageinfo, tagging, torrent, uploadId, uploads, versionId, versioning, versions, website, and x-obs-security-token. 3. If there are multiple subresources, sort them in the alphabetical order from a to z, and use **&** to combine the subresources. **NOTE** • A subresource is unique. Do not add subresources with the same keyword (for example, **key=value1&key=value2**) in the same request URL. In this case, signature is computed only based on the first subresource, and only the value of the first subresource takes effect on the actual service. • Using the **GetObject** API as an example, assume there is a bucket named **bucket-test** and an object named **object-test** in the bucket, and the object version is **xxx**. When obtaining the object, you need to rewrite Content-Type to **text/plain**. Then, the **CanonicalizedResource** calculated by the signature is **/bucket-test/object-test?response-content-type=text/plain&versionId=xxx**.

The signature is generated as follows based on the StringToSign and SK. The hash-based message authentication code algorithm (HMAC algorithm) is used to generate the signature.

```
Signature = URL-Encode( Base64( HMAC-SHA1( YourSecretAccessKeyID, UTF-8-Encoding-
Of( StringToSign ) ) ) )
```

The method for calculating the signature carried in the URL is different from that for calculating the authorization signature carried in a header.

- The signature in the URL must be encoded using the URL after Base64 encoding.

- **Expires** in **StringToSign** corresponds to **Date** in authorization information.

Generate a predefined URL instance for the browser by carrying the signature in the URL.

Table 3-13 Request that has the signature carried in the URL and the StringToSign

Request Headers	StringToSign
GET /objectkey?AccessKeyId=MFyfvK41ba2giqM7Uio6PznpdUKGpownRZlmVmHc&Expires=1532779451&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D HTTP/1.1 Host: examplebucket.obs.*region*.example.com	GET \n \n \n 1532779451\n /examplebucket/objectkey

Table 3-14 Object download request that has the temporary AK/SK and security token carried in the URL and the StringToSign

Request Header	StringToSign
GET /objectkey?AccessKeyId=MFyfvK41ba2giqM7Uio6PznpdUKGpownRZlmVmHc&Expires=1532779451&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D&x-obs-security-token=YwkaRTbdY8g7q.... HTTP/1.1 Host: examplebucket.obs.*region*.example.com	GET \n \n \n 1532779451\n /examplebucket/objectkey?x-obs-security-token=YwkaRTbdY8g7q....

Calculation rule of the signature

```
Signature = URL-Encode( Base64( HMAC-SHA1( YourSecretAccessKeyID, UTF-8-Encoding-Of( StringToSign ) ) ) )
```

Calculate the signature and use the host as the prefix of the URL to generate a predefined URL:

http(s)://examplebucket.obs.*region*.example.com/objectkey?AccessKeyId=AccessKeyID&Expires=1532779451&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D

If you enter the address in the browser, then the object **objectkey** in the **examplebucket** bucket can be downloaded. The validity period of this link is **1532779451** (indicating Sat Jul 28 20:04:11 CST 2018).

In the Linux operating system, when running the **curl** command, you need to add a forward slash (\) to escape the character (&). The following command can download the **objectkey** object to the **output** file:

curl http(s)://examplebucket.obs.*region*.example.com/objectkey?AccessKeyId=AccessKeyID\&Expires=1532779451\&Signature=0Akylf43Bm3mD1bh2rM3dmVp1Bo%3D -X GET -o output

NOTE

If you want to use the pre-defined URL generated by the signature carried in the URL in the browser, do not use Content-MD5, Content-Type, or CanonicalizedHeaders that can only be carried in the header to calculate the signature. Otherwise, the browser cannot carry these parameters. After the request is sent to the server, a message is displayed indicating that the signature is incorrect.

Signature Algorithm in Java

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Base64;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.TreeMap;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

import org.omg.CosNaming.IstringHelper;


public class SignDemo {

    private static final String SIGN_SEP = "\n";

    private static final String OBS_PREFIX = "x-obs-";

    private static final String DEFAULT_ENCODING = "UTF-8";

    private static final List<String> SUB_RESOURCES = Collections.unmodifiableList(Arrays.asList(
            "acl", "attname", "cors",  "delete",
            "deletebucket", "inventory", "length", "lifecycle", "location", "logging",
            "metadata", "modify", "name", "notification", "partNumber", "policy", "position", "quota",
            "replication", "response-cache-control", "response-content-disposition",
            "response-content-encoding", "response-content-language", "response-content-type", "response-
expires",
            "storagePolicy", "storageinfo", "tagging", "torrent",
            "uploadId", "uploads", "versionId", "versioning", "versions", "website",
             "x-obs-security-token"));

    private String ak;

    private String sk;

     public String urlEncode(String input) throws UnsupportedEncodingException
    {
        return URLEncoder.encode(input, DEFAULT_ENCODING)
        .replaceAll("%7E", "~") //for browser
```

```
        .replaceAll("%2F", "/");
    }

    private String join(List<?> items, String delimiter)
    {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < items.size(); i++)
        {
    String item = items.get(i).toString();
            sb.append(item);
            if (i < items.size() - 1)
            {
                sb.append(delimiter);
            }
        }
        return sb.toString();
    }

    private boolean isValid(String input) {
        return input != null && !input.equals("");
    }

    public String hamcSha1(String input) throws NoSuchAlgorithmException, InvalidKeyException,
UnsupportedEncodingException {
        SecretKeySpec signingKey = new SecretKeySpec(this.sk.getBytes(DEFAULT_ENCODING), "HmacSHA1");
        Mac mac = Mac.getInstance("HmacSHA1");
        mac.init(signingKey);
        return Base64.getEncoder().encodeToString(mac.doFinal(input.getBytes(DEFAULT_ENCODING)));
    }

    private String stringToSign(String httpMethod, Map<String, String[]> headers, Map<String, String>
queries,
            String bucketName, String objectName) throws Exception{
        String contentMd5 = "";
        String contentType = "";
        String date = "";

        TreeMap<String, String> canonicalizedHeaders = new TreeMap<String, String>();

        String key;
        List<String> temp = new ArrayList<String>();
        for(Map.Entry<String, String[]> entry : headers.entrySet()) {
            key = entry.getKey();
            if(key == null || entry.getValue() == null || entry.getValue().length == 0) {
                continue;
            }

            key = key.trim().toLowerCase(Locale.ENGLISH);
            if(key.equals("content-md5")) {
                contentMd5 = entry.getValue()[0];
                continue;
            }

            if(key.equals("content-type")) {
                contentType = entry.getValue()[0];
                continue;
            }

            if(key.equals("date")) {
                date = entry.getValue()[0];
                continue;
            }

            if(key.startsWith(OBS_PREFIX)) {

                for(String value : entry.getValue()) {
                    if(value != null) {
                        temp.add(value.trim());
                    }
```

```
                }
                canonicalizedHeaders.put(key, this.join(temp, ","));
                temp.clear();
            }
        }

        if(canonicalizedHeaders.containsKey("x-obs-date")) {
            date = "";
        }


        // handle method/content-md5/content-type/date
        StringBuilder stringToSign = new StringBuilder();
        stringToSign.append(httpMethod).append(SIGN_SEP)
            .append(contentMd5).append(SIGN_SEP)
            .append(contentType).append(SIGN_SEP)
            .append(date).append(SIGN_SEP);

        // handle canonicalizedHeaders
        for(Map.Entry<String, String> entry : canonicalizedHeaders.entrySet()) {
            stringToSign.append(entry.getKey()).append(":").append(entry.getValue()).append(SIGN_SEP);
        }

        // handle CanonicalizedResource
        stringToSign.append("/");
        if(this.isValid(bucketName)) {
            stringToSign.append(bucketName).append("/");
            if(this.isValid(objectName)) {
                stringToSign.append(this.urlEncode(objectName));
            }
        }

        TreeMap<String, String> canonicalizedResource = new TreeMap<String, String>();
        for(Map.Entry<String, String> entry : queries.entrySet()) {
            key = entry.getKey();
            if(key == null) {
                continue;
            }

            if(SUB_RESOURCES.contains(key)) {
                canonicalizedResource.put(key, entry.getValue());
            }
        }

        if(canonicalizedResource.size() > 0) {
            stringToSign.append("?");
            for(Map.Entry<String, String> entry : canonicalizedResource.entrySet()) {
                stringToSign.append(entry.getKey());
                if(this.isValid(entry.getValue())) {
                    stringToSign.append("=").append(entry.getValue());
                }
                            stringToSign.append("&");
                }
                        stringToSign.deleteCharAt(stringToSign.length()-1);
        }

//        System.out.println(String.format("StringToSign:%s%s", SIGN_SEP, stringToSign.toString()));

        return stringToSign.toString();
    }

    public String headerSignature(String httpMethod, Map<String, String[]> headers, Map<String, String>
queries,
            String bucketName, String objectName) throws Exception {

        //1. stringToSign
        String stringToSign = this.stringToSign(httpMethod, headers, queries, bucketName, objectName);

        //2. signature
```

```
        return String.format("OBS %s:%s", this.ak, this.hamcSha1(stringToSign));
    }

    public String querySignature(String httpMethod, Map<String, String[]> headers, Map<String, String>
queries,
            String bucketName, String objectName, long expires) throws Exception {
        if(headers.containsKey("x-obs-date")) {
            headers.put("x-obs-date", new String[] {String.valueOf(expires)});
        }else {
            headers.put("date", new String[] {String.valueOf(expires)});
        }
        //1. stringToSign
        String stringToSign = this.stringToSign(httpMethod, headers, queries, bucketName, objectName);

        //2. signature
        return this.urlEncode(this.hamcSha1(stringToSign));
    }

        public String getURL(String endpoint, Map<String, String> queries,
                String bucketName, String objectName, String signature, long expires) {
                StringBuilder URL = new StringBuilder();
                URL.append("https://").append(bucketName).append(".").append(endpoint).append("/").
                    append(objectName).append("?");
                String key;
                for (Map.Entry<String, String> entry : queries.entrySet()) {
                    key = entry.getKey();
                    if (key == null) {
                        continue;
                    }
                    if (SUB_RESOURCES.contains(key)) {
                        String value = entry.getValue();
                        URL.append(key);
                        if (value != null) {
                            URL.append("=").append(value).append("&");
                        } else {
                            URL.append("&");
                        }
                    }
                }
                URL.append("AccessKeyId=").append(this.ak).append("&Expires=").append(expires).
                    append("&Signature=").append(signature);
                return URL.toString();
        }

    public static void main(String[] args) throws Exception {

        SignDemo demo = new SignDemo();
        demo.ak = "<your-access-key-id>";
        demo.sk = "<your-secret-key-id>";
                String endpoint = "<your-endpoint>";

        String bucketName = "bucket-test";
        String objectName = "hello.jpg";
// A header cannot be carried if you want to use a URL to access OBS through a browser. If a header is
added to headers, the signature does not match. To use the headers, it must be processed by the client.
        Map<String, String[]> headers = new HashMap<String, String[]>();
        Map<String, String> queries = new HashMap<String, String>();

                //  Expiration time of the request message. Set it to expire in 24 hours.
                long expires = (System.currentTimeMillis() + 86400000L) / 1000;
                String signature = demo.querySignature("GET", headers, queries, bucketName, objectName,
expires);
                System.out.println(signature);
                String URL = demo.getURL(endpoint, queries, bucketName, objectName, signature, expires);
                System.out.println(URL);
    }

}
```

3.2.4 Authentication of Signature Carried in the Table Uploaded Through a Browser

OBS supports browser-based object upload using the POST method. Signatures of such requests are uploaded in tables. First, create a security policy and specify the requirements in the request, for example, Bucket name and object name prefix. Then, create a signature based on this policy. The request form to be signed must contain valid signature and policy. Finally, create a table to upload the object to the bucket.

The signature calculation process is as follows:

1. The policy content is encoded in UTF-8 format.
2. Perform Base64 encoding on the result obtained from the preceding step.
3. Use the SK to perform the HMAC-SHA1 signature calculation on the result obtained from step 2.
4. Perform Base64 encoding on the result of step 3 to obtain the signature.

```
StringToSign = Base64( UTF-8-Encoding-Of( policy ) )
Signature = Base64( HMAC-SHA1( YourSecretAccessKeyID, StringToSign ) )
```

The content of the policy is as follows:

```
{ "expiration": "2017-12-31T12:00:00.000Z",
  "conditions": [
    {"x-obs-acl": "public-read" },
    {"x-obs-security-token": "YwkaRTbdY8g7q...." },
    {"bucket": "book" },
    ["starts-with", "$key", "user/"]
  ]
}
```

The policy contains the validity period and conditions.

Expiration

Indicates the validity period of the signature, which is expressed in the format according to ISO 8601 UTC. For example, **expiration: 2017-12-31T12:00:00.000Z** in the example means that the request is invalid after 12:00:00 on December 31, 2017. This field is mandatory in the policy and can only be in the format of **yyyy-MM-dd'T'HH:mm:ss'Z'** or **yyyy-MM-dd'T'HH:mm:ss.SSS'Z'**.

Conditions

A mechanism used to verify the validity of a request. Conditions are used to define the content that must be contained in a request. In the example, the requested bucket name is **book**, the object name is prefixed with **user/**, and the ACL of the object is public read. All items in the form, excluding **AccessKeyId**, **signature**, **file**, **policy**, **token**, **field names**, and the prefix **x-ignore-**, must be included in the policy. The following table lists the items that should be contained in **Conditions**.

Table 3-15 Conditions contained in a policy

Element	Description
x-obs-acl	ACL in the request. Supports exact match and conditional match such as **starts-with**.
content-length-range	Maximum and minimum length of an object to be uploaded. The value can be a range.
Cache-Control, Content-Type, Content-Disposition, Content-Encoding, Expires	Headers specially for REST requests Supports exact match and conditional match such as **starts-with**.
key	Name of an object to be uploaded. Supports exact match and conditional match such as **starts-with**.
bucket	Name of the requested bucket. Supports exact match.
success_action_redirect	Redirection address after the upload is successful. For details, see **Uploading Objects - POST**. Supports exact match and conditional match such as **starts-with**.
success_action_status	If **success_action_redirect** is not specified, the status code is returned to the client when the upload is successful. For details, see **Uploading Objects - POST**. Supports exact match.
x-obs-meta-*	User-defined metadata. Keywords in an element cannot contain non-ASCII or unrecognizable characters. If non-ASCII or unrecognizable characters are necessary, they should be encoded and decoded on the client side. Either URL encoding or Base64 encoding is acceptable, but the server does not perform decoding. Supports exact match and conditional match such as **starts-with**.
x-obs-*	Other header fields with prefix **x-obs-**. Supports exact match and conditional match such as **starts-with**.

Element	Description
x-obs-security-token	Field name in the request header. Mandatory field for the temporary AK/SK and security token authentication.

The policy conditions can be matched in the following ways:

Table 3-16 Policy condition matching methods

Matching Method	Description
Exact Matches	Exact match by default. The value in the POST table must be the same as that in the policy. For example, if object ACL is set to **public-read** when the object is uploaded, the value of the **x-obs-acl** element in the table is **public-read**. Therefore, the conditions in the policy can be set to {"x-obs-acl": "public-read"} or ["eq", "$x-obs-acl", "public-read"], which are equivalent.
Starts With	If this condition is used, the value set in the POST table must start with a fixed character string. For example, if the name of uploaded objects must be prefixed with **user/**, the value of the **key** element in the table can be **user/test1**, **user/test2**, and so on. Therefore, conditions in the policy can be set to: **["starts-with", "$key", "user/"]**
Matching Any Content	The corresponding element in the POST table can be any value. For example, if the redirection address upon request success can be any address, the value of the **success_action_redirect** element in the table can be any value. Therefore, conditions in the policy can be set to: **["starts-with", "$success_action_redirect", ""]**

Matching Method	Description
Specifying Ranges	The content length of the **file** element in the POST table can be a specified range and is used only to limit the object size. For example, if the size of the uploaded object is between 1 MB to 10 MB, the content length of the **file** element in the table can be from **1048576** to **10485760**. Therefore, conditions in the policy can be set to (the value does not contain quotation marks) **["content-length-range", 1048576, 10485760]**

NOTE

A policy is in the JSON format. Conditions can be put in curly brackets {} and square brackets []. The key and value elements of the table are written in the curly brackets {}, which are separated by colons (:). The square brackets [] contain the condition type, key, and value, which are separated by commas (,). The dollar sign ($) in front of the key indicates that the key is a variable.

The following characters must be escaped in a policy:

Table 3-17 Characters that must be escaped in a policy

Character After Escape	Real Character
\\	Backslash (\)
\$	Dollar symbol ($)
\b	Backspace
\f	Page up and down
\n	Line breaks
\r	Enter
\t	Horizontal table
\v	Vertical table
\u***xxxx***	All Unicode characters

The following tables provide examples of requests and policies.

Table 3-18 Uploading the **testfile.txt** object to the **examplebucket** bucket and setting the object ACL to **public-read**

Request	policy

POST / HTTP/1.1
Host: examplebucket.obs.*region*.example.com
Content-Type: multipart/form-data; boundary=7e32233530b26
Content-Length: 1250
--7e32233530b26
Content-Disposition: form-data; name="key"
testfile.txt
--7e32233530b26
Content-Disposition: form-data; name="x-obs-acl"
public-read
--7e32233530b26
Content-Disposition: form-data; name="content-type"
text/plain
--7e32233530b26
Content-Disposition: form-data; name="AccessKeyId"
UDSIAMSTUBTEST000002
--7e32233530b26
Content-Disposition: form-data; name="policy"
ewogICJleHBpcmF0aW9uIjogIjIwMTktMDctMDFUMTI6MDA6MDAuMDAwWiIsCiAgImNvbmRpdGlvbnMiOiBbCiAgICB7ImJ1Y2tldCI6ICJleGFtcGxlYnVja2V0IiB9LAogICAgWyJlcSIsICIka2V5IiwgInRlc3RmaWxlLnR4dCJdLAogICAgeyJ4LW9icy1hY2wiOiAicHVibGljLXJlYWQiIH0sCiAgICBbImVxIiwgIiRDb250ZW50LVR5cGUiLCAidGV4dC9wbGFpbiJdLAogICAgWyJjb250ZW50LWxlbmd0aC1yYW5nZSIsIDYsIDEwXQogIF0KfQo=
--7e32233530b26
	{
	"expiration": "2019-07-01T12:00:00.000Z",
	"conditions": [
	{"bucket": "examplebucket" },
	["eq", "$key", "testfile.txt"],
	{"x-obs-acl": "public-read" },
	["eq", "$Content-Type", "text/plain"]
	]
	}

Request	policy
Content-Disposition: form-data; name="signature" xxl7bZs/5FgtBUggOdQ88DPZUo0= --7e32233530b26 Content-Disposition: form-data; name="file"; filename="E:\TEST_FILE\TEST.txt" Content-Type: text/plain 123456 --7e32233530b26 Content-Disposition: form-data; name="submit" Upload --7e32233530b26--	

Table 3-19 Uploading the **file/obj1** object to the **examplebucket** bucket and configuring the four user-defined metadata of the object

Request	policy

POST / HTTP/1.1
Host: examplebucket.obs.*region*.example.com
Content-Type: multipart/form-data; boundary=7e329d630b26
Content-Length: 1597
 	
--7e3542930b26
Content-Disposition: form-data; name="key"
 	
file/obj1
--7e3542930b26
Content-Disposition: form-data; name="AccessKeyId"
 	
UDSIAMSTUBTEST000002
--7e3542930b26
Content-Disposition: form-data; name="policy"
 	
ewogICJleHBpcmF0aW9uIjogIjIwMTkt
MDctMDFUMTI6MDA6MDAuMDAwWi
IsCiAgImNvbmRpdGlvbnMiOiBbCiA-
gICB7ImJ1Y2tldCI6ICJleGFtcGxlYnV-
ja2V0IiB9LAogICAgWyJzdGFydHMtd2l0
aCIsICIka2V5IiwgImZpbGUvIl0sCiAgICB
7Ingtb2JzLW1ldGEtdGVzdDEiOiJ2YWx1
ZTEifSwKICAgIFsiZXEiL-
CAiJHgtb2JzLW1ldGEtdGVzdDIiLCAidm
FsdWUyIl0sCiAgICBbInN0YXJ0cy13aXR
oIiwgIiR4LW9icy1tZXRhLXRlc3QzIiwgI
mRvYyJdLAogICAgWyJzdG-
FydHMtd2l0aCIsICIkeC1vYnMtbWV0YS
10ZXN0NCIsICIiXQogIF0KfQo=
--7e3542930b26
Content-Disposition: form-data; name="signature"
 	
HTId8OCBisn6FfdWKqSJP9RN4Oo=
--7e3542930b26
Content-Disposition: form-data; name="x-obs-meta-test1"
 	
value1
	{
	"expiration": "2019-07-01T12:00:00.000Z",
	"conditions": [
	{"bucket": "examplebucket" },
	["starts-with", "$key", "file/"],
	{"x-obs-meta-test1":"value1"},
	["eq", "$x-obs-meta-test2", "value2"],
	["starts-with", "$x-obs-meta-test3", "doc"],
	["starts-with", "$x-obs-meta-test4", ""]
	]
	}

Request	policy
--7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test2" value2 --7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test3" doc123 --7e3542930b26 Content-Disposition: form-data; name="x-obs-meta-test4" my --7e3542930b26 Content-Disposition: form-data; name="file"; filename="E:\TEST_FILE\TEST.txt" Content-Type: text/plain 123456 --7e3542930b26 Content-Disposition: form-data; name="submit" Upload --7e3542930b26--	

3.3 Returned Values

After sending a request, you will receive a response, including the status code, response header, and response body.

Status Codes

A status code is a group of digits ranging from 2*xx* (indicating successes) to 4*xx* or 5*xx* (indicating errors). It indicates the status of a response. For more information, see **Status Codes**.

Response Headers

A response header corresponds to a request header, for example, Content-Type.

For details about common response headers, see **Table 3-20**.

Table 3-20 Common response headers

Header	Description
Content-Length	The length (in bytes) of the response body. Type: string Default value: none
Connection	Indicates whether the connection to the server is a long connection or a short connection. Type: string Valid values: keep-alive | close Default value: none
Date	The date and time at which OBS responds to the request. Type: string Default value: none
ETag	128-bit MD5 digest of the Base64 code of an object. ETag is the unique identifier of the object content. It can be used to identify whether the object content is changed. For example, if ETag value is **A** when an object is uploaded and the ETag value has changed to **B** when the object is downloaded, it indicates that the object content is changed. The actual ETag is the hash value of the object, which only reflects the changed content rather than the metadata. An uploaded object or copied object has a unique ETag after being encrypted using MD5. If an object is uploaded in the multipart mode, the MD5 splits ETag regardless of the encryption method. In this case, the ETag is not an MD5 digest. Type: string
x-obs-id-2	A special symbol that helps troubleshoot faults. Type: string Default value: none
x-obs-request-id	The value created by OBS to uniquely identify the request. OBS uses this value to troubleshoot faults. Type: string Default value: none

(Optional) Response Body

A response body is generally returned in a structured format (for example, JSON or XML), corresponding to **Content-Type** in the response header, and is used to transfer content other than the response header.

4 Getting Started

4.1 Creating a Bucket

Scenarios

A bucket is a container that stores objects in OBS. You need to create a bucket before storing data in OBS.

The following describes how to call the API for **creating a bucket** in a specified region. For details about how to call an API, see **Calling APIs**.

Prerequisites

- You have obtained the AK and SK. For details about how to obtain the AK and SK, see **Obtaining Access Keys (AK/SK)**.
- You need to plan the region where the bucket resides and determine the endpoint for calling an API based on the region. For details, see **Regions and Endpoints**.

Once a region is determined, it cannot be modified after the bucket is created.

Creating a Bucket Named bucket001 in the a1 Region

In this example, an Apache HttpClient is used.

```
package com.obsclient;

import java.io.*;

import org.apache.http.Header;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpPut;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;

public class TestMain {

    public static String accessKey = "UDSIAMSTUBTEST000012"; //The value of this parameter is the AK obtained.
     public static String securityKey = "Udsiamstubtest000000UDSIAMSTUBTEST000012";  //The value of this parameter is the SK obtained.
```

```
    public static String region = "a1"; // The value is the region where the planned bucket resides.
    public static String createBucketTemplate =
            "<CreateBucketConfiguration " +
            "xmlns=\"http://obs.a1.example.com/doc/2015-06-30/\">\n" +
            "<Location>" + region + "</Location>\n" +
            "</CreateBucketConfiguration>";

    public static void main(String[] str) {

         createBucket();

    }

    private static void createBucket() {
        CloseableHttpClient httpClient = HttpClients.createDefault();
        String requesttime = DateUtils.formatDate(System.currentTimeMillis());
        String contentType = "application/xml";


        HttpPut httpPut = new HttpPut("http://bucket001.obs.a1.example.com");
        httpPut.addHeader("Date", requesttime);
        httpPut.addHeader("Content-Type", contentType);

        /**Calculate the signature based on the request.**/
        String contentMD5 = "";
        String canonicalizedHeaders = "";
        String canonicalizedResource = "/bucket001/";
        // Content-MD5 and Content-Type fields do not contain line breaks. The data format is RFC 1123,
which is the same as the time in the request.
        String canonicalString = "PUT" + "\n" + contentMD5 + "\n" + contentType + "\n" + requesttime + "\n"
+ canonicalizedHeaders + canonicalizedResource;
        System.out.println("StringToSign:[" + canonicalString + "]");
        String signature = null;
        CloseableHttpResponse httpResponse = null;
        try {
            signature = Signature.signWithHmacSha1(securityKey, canonicalString);

             // Added the Authorization: OBS AccessKeyID:signature field to the header.
            httpPut.addHeader("Authorization", "OBS " + accessKey + ":" + signature);

            // Add a body.
            httpPut.setEntity(new StringEntity(createBucketTemplate));

            httpResponse = httpClient.execute(httpPut);

            // Prints the sending request information and the received response message.
            System.out.println("Request Message:");
            System.out.println(httpPut.getRequestLine());
            for (Header header : httpPut.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }

            System.out.println("Response Message:");
            System.out.println(httpResponse.getStatusLine());
            for (Header header : httpResponse.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    httpResponse.getEntity().getContent()));

            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = reader.readLine()) != null) {
                response.append(inputLine);
            }
            reader.close();

            // print result
```

```
            System.out.println(response.toString());
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                httpClient.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

    }

}
```

The format of the **Date** header field **DateUtils** is as follows:

```
package com.obsclient;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Locale;
import java.util.TimeZone;

public class DateUtils {

    public static String formatDate(long time)
    {
        DateFormat serverDateFormat = new SimpleDateFormat("EEE, dd MMM yyyy HH:mm:ss z",
Locale.ENGLISH);
        serverDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
        return serverDateFormat.format(time);
    }
}
```

The method of calculating the signature character string is as follows:

```
package com.obsclient;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;
import java.security.InvalidKeyException;
import java.util.Base64;

public class Signature {
    public static String signWithHmacSha1(String sk, String canonicalString) throws
UnsupportedEncodingException {

        try {
            SecretKeySpec signingKey = new SecretKeySpec(sk.getBytes("UTF-8"), "HmacSHA1");
            Mac mac = Mac.getInstance("HmacSHA1");
            mac.init(signingKey);
            return Base64.getEncoder().encodeToString(mac.doFinal(canonicalString.getBytes("UTF-8")));
        } catch (NoSuchAlgorithmException | InvalidKeyException | UnsupportedEncodingException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

4.2 Listing Buckets

Scenarios

If you want to view information about all buckets created by yourself, you can call the API for listing buckets.

The following describes how to call the API for **listing buckets**. For details about how to call an API, see **Calling APIs**.

Prerequisites

- You have obtained the AK and SK. For details about how to obtain the AK and SK, see **Obtaining Access Keys (AK/SK)**.
- You need to specify the region where the buckets to be listed reside and determine the endpoint for calling an API based on the region. For details, see **Regions and Endpoints**.

Obtaining the Bucket List in the a1 Region

In this example, an Apache HttpClient is used.

```
package com.obsclient;

import java.io.*;
import java.util.ArrayList;
import java.util.List;

import org.apache.http.Header;
import org.apache.http.HttpEntity;
import org.apache.http.NameValuePair;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.methods.HttpPut;
import org.apache.http.entity.InputStreamEntity;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.message.BasicNameValuePair;


public class TestMain {

    public static String accessKey = "UDSIAMSTUBTEST000012"; //The value of this parameter is the AK obtained.
     public static String securityKey = "Udsiamstubtest000000UDSIAMSTUBTEST000012";  //The value of this parameter is the SK obtained.

    public static void main(String[] str) {

        listAllMyBuckets();

    }


    private static void listAllMyBuckets() {
        CloseableHttpClient httpClient = HttpClients.createDefault();
        String requesttime = DateUtils.formatDate(System.currentTimeMillis());
```

```
        HttpGet httpGet = new HttpGet("http://obs.a1.example.com");
        httpGet.addHeader("Date", requesttime);

        /**Calculate the signature based on the request.**/
        String contentMD5 = "";
        String contentType = "";
        String canonicalizedHeaders = "";
        String canonicalizedResource = "/";
        // Content-MD5 and Content-Type fields do not contain line breaks. The data format is RFC 1123,
which is the same as the time in the request.
        String canonicalString = "GET" + "\n" + contentMD5 + "\n" + contentType + "\n" + requesttime + "\n"
+ canonicalizedHeaders + canonicalizedResource;
        System.out.println("StringToSign:[" + canonicalString + "]");
        String signature = null;
        try {
            signature = Signature.signWithHmacSha1(securityKey, canonicalString);

             // Added the Authorization: OBS AccessKeyID:signature field to the header.
            httpGet.addHeader("Authorization", "OBS " + accessKey + ":" + signature);
            CloseableHttpResponse httpResponse = httpClient.execute(httpGet);

            // Prints the sending request information and the received response message.
            System.out.println("Request Message:");
            System.out.println(httpGet.getRequestLine());
            for (Header header : httpGet.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }

            System.out.println("Response Message:");
            System.out.println(httpResponse.getStatusLine());
            for (Header header : httpResponse.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    httpResponse.getEntity().getContent()));

            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = reader.readLine()) != null) {
                response.append(inputLine);
            }
            reader.close();
            // print result
            System.out.println(response.toString());
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();

        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                httpClient.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

    }

}
```

The format of the **Date** header field **DateUtils** is as follows:

```
package com.obsclient;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
```

```
import java.util.Locale;
import java.util.TimeZone;

public class DateUtils {

    public static String formatDate(long time)
    {
        DateFormat serverDateFormat = new SimpleDateFormat("EEE, dd MMM yyyy HH:mm:ss z",
Locale.ENGLISH);
        serverDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
        return serverDateFormat.format(time);
    }
}
```

The method of calculating the signature character string is as follows:

```
package com.obsclient;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;
import java.security.InvalidKeyException;
import java.util.Base64;

public class Signature {
    public static String signWithHmacSha1(String sk, String canonicalString) throws
UnsupportedEncodingException {

        try {
            SecretKeySpec signingKey = new SecretKeySpec(sk.getBytes("UTF-8"), "HmacSHA1");
            Mac mac = Mac.getInstance("HmacSHA1");
            mac.init(signingKey);
            return Base64.getEncoder().encodeToString(mac.doFinal(canonicalString.getBytes("UTF-8")));
        } catch (NoSuchAlgorithmException | InvalidKeyException | UnsupportedEncodingException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

4.3 Uploading an Object

Scenarios

You can upload files of any type to OBS buckets for storage.

The following describes how to call the API for **uploading objects using the PUT method** to a specified bucket. For details about how to call an API, see **Calling APIs**.

Prerequisites

- You have obtained the AK and SK. For details, see **Obtaining Access Keys (AK/SK)**.
- At least one bucket is available.
- The file to be uploaded has been prepared and you know the complete local path of the file.
- You need to know the region of the bucket which you want to upload the file to and determine the endpoint for calling an API based on the region. For details, see **Regions and Endpoints**.

Uploading the Object objecttest1 to Bucket bucket001 in the a1 Region

In this example, an Apache HttpClient is used.

```
package com.obsclient;

import java.io.*;
import java.util.ArrayList;
import java.util.List;

import org.apache.http.Header;
import org.apache.http.HttpEntity;
import org.apache.http.NameValuePair;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.methods.HttpPut;
import org.apache.http.entity.InputStreamEntity;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.message.BasicNameValuePair;


public class TestMain {

    public static String accessKey = "UDSIAMSTUBTEST000012"; //The value of this parameter is the AK obtained.
     public static String securityKey = "Udsiamstubtest000000UDSIAMSTUBTEST000012";  //The value of this parameter is the SK obtained.

    public static void main(String[] str) {

        putObjectToBucket();

    }


    private static void putObjectToBucket() {

        InputStream inputStream = null;
        CloseableHttpClient httpClient = HttpClients.createDefault();
        CloseableHttpResponse httpResponse = null;
        String requestTime = DateUtils.formatDate(System.currentTimeMillis());


        HttpPut httpPut = new HttpPut("http://bucket001.obs.a1.example.com/objecttest1");
        httpPut.addHeader("Date", requestTime);

         /**Calculate the signature based on the request.**/
        String contentMD5 = "";
        String contentType = "";
        String canonicalizedHeaders = "";
        String canonicalizedResource = "/bucket001/objecttest1";
        // Content-MD5 and Content-Type fields do not contain line breaks. The data format is RFC 1123, which is the same as the time in the request.
        String canonicalString = "PUT" + "\n" + contentMD5 + "\n" + contentType + "\n" + requestTime + "\n" + canonicalizedHeaders + canonicalizedResource;
        System.out.println("StringToSign:[" + canonicalString + "]");
        String signature = null;
        try {
            signature = Signature.signWithHmacSha1(securityKey, canonicalString);
            // Directory for storing uploaded files
            inputStream = new FileInputStream("D:\\OBSobject\\text01.txt");
            InputStreamEntity entity = new InputStreamEntity(inputStream);
            httpPut.setEntity(entity);

           // Added the Authorization: OBS AccessKeyID:signature field to the header.
```

```
            httpPut.addHeader("Authorization", "OBS " + accessKey + ":" + signature);
            httpResponse = httpClient.execute(httpPut);

           // Prints the sending request information and the received response message.
            System.out.println("Request Message:");
            System.out.println(httpPut.getRequestLine());
            for (Header header : httpPut.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }

            System.out.println("Response Message:");
            System.out.println(httpResponse.getStatusLine());
            for (Header header : httpResponse.getAllHeaders()) {
                System.out.println(header.getName() + ":" + header.getValue());
            }
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    httpResponse.getEntity().getContent()));

            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = reader.readLine()) != null) {
                response.append(inputLine);
            }
            reader.close();

            // print result
            System.out.println(response.toString());


        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();

        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                httpClient.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

}
```

The format of the Date header field DateUtils is as follows:

```
package com.obsclient;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Locale;
import java.util.TimeZone;

public class DateUtils {

    public static String formatDate(long time)
    {
        DateFormat serverDateFormat = new SimpleDateFormat("EEE, dd MMM yyyy HH:mm:ss z",
Locale.ENGLISH);
        serverDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
        return serverDateFormat.format(time);
    }
}
```

The method of calculating the signature character string is as follows:

```
package com.obsclient;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.io.UnsupportedEncodingException;
import java.security.NoSuchAlgorithmException;
import java.security.InvalidKeyException;
import java.util.Base64;

public class Signature {
    public static String signWithHmacSha1(String sk, String canonicalString) throws
UnsupportedEncodingException {

        try {
            SecretKeySpec signingKey = new SecretKeySpec(sk.getBytes("UTF-8"), "HmacSHA1");
            Mac mac = Mac.getInstance("HmacSHA1");
            mac.init(signingKey);
            return Base64.getEncoder().encodeToString(mac.doFinal(canonicalString.getBytes("UTF-8")));
        } catch (NoSuchAlgorithmException | InvalidKeyException | UnsupportedEncodingException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

5 APIs

5.1 Operations on Buckets

5.1.1 Listing Buckets

Functions

You can perform this operation to list all buckets that you have created.

Request Syntax

```
GET / HTTP/1.1
Host: obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request header uses common message fields. For details, see **Table 3-3**.

Request Elements

The request does not use request elements.

Response Syntax

```
GET HTTP/1.1 status_code
Content-Type: type
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListAllMyBucketsResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Owner>
      <ID>id</ID>
```

```
  </Owner>
  <Buckets>
    <Bucket>
      <Name>bucketName</Name>
      <CreationDate>date</CreationDate>
      <Location>region</Location>
    </Bucket>
    ...
  </Buckets>
</ListAllMyBucketsResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains the XML list of buckets owned by the user. **Table 5-1** describes the elements.

Table 5-1 Response elements

Element	Description
ListAllMyBucketsResult	List of buckets created by the user Type: XML
Owner	Bucket owner information, including the tenant ID. Type: XML
ID	Domain ID (account ID) of a user. Type: string
Buckets	Buckets owned by the user Type: XML
Bucket	Details about a bucket Type: XML
Name	Bucket name Type: string
CreationDate	Creation time of the bucket Type: string
Location	Location of the bucket Type: string

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
GET / HTTP/1.1
User-Agent: curl/7.29.0
Host: obs.region.example.com
Accept: */*
Date: Mon, 25 Jun 2018 05:37:12 +0000
Authorization: OBS GKDF4C7Q6SI0IPGTXTJN:9HXkVQIiQKw33UEmyBI4rWrzmic=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435722C11379647A8A00A
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSGGDRUM62QZi3hGP8Fz3gOloYCfZ39U
Content-Type: application/xml
Date: Mon, 25 Jun 2018 05:37:12 GMT
Content-Length: 460

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListAllMyBucketsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>783fc6652cf246c096ea836694f71855</ID>
  </Owner>
  <Buckets>
    <Bucket>
      <Name>examplebucket01</Name>
      <CreationDate>2018-06-21T09:15:01.032Z</CreationDate>
      <Location>region</Location>
    </Bucket>
    <Bucket>
      <Name>examplebucket02</Name>
      <CreationDate>2018-06-22T03:56:33.700Z</CreationDate>
      <Location>region</Location>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

5.1.2 Creating a Bucket

Functions

This operation is used to create a bucket with a specified name.

NOTE

- By default, a user can have a maximum of 100 buckets.
- The name of a deleted bucket can be reused for a bucket at least 30 minutes after the deletion.

A bucket name must be unique in OBS. If a user creates a bucket with the same name as that of an existing bucket under the same account and in the same region, a 200 code (indicating success) is returned. In scenarios other than the preceding one, the request for creating a bucket with the same name as that of an existing one will receive the 409 code (indicating that a namesake bucket already exists). To set an access control policy for the bucket to be created, you can add the **x-obs-acl** parameter to request headers.

Request Syntax

```
PUT / HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
```

```
Date: date
Authorization: authorization
<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>location</Location>
</CreateBucketConfiguration>
```

Request Parameters

This request contains no parameter.

Request Headers

The operation message header is the same as that of a common request. For details, see **Table 3-3**. However, this request can contain additional headers. The following table describes the additional headers for this request.

Table 5-2 Additional request headers

Header	Description	Mandatory
x-obs-acl	When creating a bucket, you can add this header to set the permission control policy for the bucket. The predefined common policies are as follows: **private**, **public-read**, **public-read-write**, **public-read-delivered**, and **public-read-write-delivered**. Type: string	No
x-obs-grant-read	This header grants the read permission to all users under an account. It allows you to list objects in a bucket, list multipart tasks in a bucket, list multi-version objects in a bucket, and obtain bucket metadata. Type: string Example: **x-obs-grant-read:id=Tenant ID**.	No
x-obs-grant-write	This header grants the write permission to all users under an account. Therefore, the users can create, delete, and overwrite all objects in a bucket, and can initialize parts, upload parts, copy parts, merge parts, and cancel multipart upload tasks. Type: string Example: **x-obs-grant-write:id=Tenant ID**.	No
x-obs-grant-read-acp	This header grants the ACL read permission to all users under an account. Therefore, the users can read the bucket ACL information. Type: string Example: **x-obs-grant-read-acp:id=Account ID**.	No

Header	Description	Mandatory
x-obs-grant-write-acp	This header grants the ACL write permission to all users under an account. Therefore, the users can modify the ACL of the bucket. Type: string Example: **x-obs-grant-write-acp:id=Account ID**.	No
x-obs-grant-full-control	This header grants the full control permission to all users under an account. Type: string Example: **x-obs-grant-full-control:id=Account ID**.	No
x-obs-grant-read-delivered	This header grants the read permission to all users under an account. By default, the read permission is applied to all objects in the bucket. Type: string Example: **x-obs-grant-read-delivered:id=Account ID**.	No
x-obs-grant-full-control-delivered	This header grants the full control permission to all users under an account. By default, the FULL_CONTROL permission is applied to all objects in the bucket. Type: string Example: **x-obs-grant-full-control-delivered:id=Account ID**.	No

Request Elements

This request can use additional elements. For details about additional elements, see **Table 5-3**.

Table 5-3 Additional request elements

Element	Description	Mandatory
Location	Specifies the region where a bucket will be created. • When creating a bucket using the endpoint of the default region, note the following: – If **Location** is not specified, the bucket is created in the default region. – If Location is specified to other region, the bucket is created in the specified region. • When creating a bucket using the endpoint of a non-default region, **Location** must be specified to the region corresponding to the endpoint. For details about OBS regions and endpoints, see **Regions and Endpoints**. Type: string	No

Response Syntax

```
HTTP/1.1 status_code
Location: location
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request 1

Create a bucket.

```
PUT / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 157

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
    <Location>region</Location>
</CreateBucketConfiguration>
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

Sample Request 2

Create a bucket with a specified ACL.

```
PUT / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:25:05 GMT
x-obs-acl:public-read
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:75/Y4Ng1izvzc1nTGxpMXTE6ynw=
Content-Length: 157

<CreateBucketConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
    <Location>region</Location>
</CreateBucketConfiguration>
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435CE298386946AE4C482
Location: /examplebucket
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT9W2tcvLmMJ+plfdopaD62S0npbaRUz
Date: WED, 01 Jul 2015 02:25:06 GMT
Content-Length: 0
```

5.1.3 Listing Objects in a Bucket

Functions

This operation lists objects in a bucket. To use this operation, you must have the permission to read the bucket.

If you specify only the bucket name in the request URI, for example **GET / BucketName**, OBS returns descriptions for some or all objects (a maximum of 1000 objects) in the bucket. If you also specify one or more parameters among

prefix, **marker**, **max-keys**, and **delimiter** in the request, OBS returns a list of objects as specified.

You can also add the **versions** parameter to the request to list multiple versions of an object in a bucket.

Request Syntax

```
GET / HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Syntax (for multi-version objects)

```
GET /?versions HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request uses parameters to list some objects in the bucket. **Table 5-4** describes the parameters.

Table 5-4 Request parameters

Parameter	Description	Mandatory
prefix	Lists objects that begin with the specified prefix. Type: string	No
marker	Specifies a marker when listing objects in a bucket. With a marker configured, objects after this marker will be returned in alphabetical order. Type: string	No
max-keys	Sets the maximum number of objects (in alphabetical order) returned in the response body. The value ranges from 1 to 1,000. If the value has exceeded the upper limit, 1,000 objects are returned by default. Type: integer	No

Parameter	Description	Mandatory
delimiter	Separator used to group object names. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one **CommonPrefixes**. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one **CommonPrefixes**. For example, there are three objects (**abcd**, **abcde**, and **bbcde**) in a bucket. If **delimiter** is set to **d** and **prefix** is set to **a**, objects **abcd** and **abcde** are grouped into a **CommonPrefixes** with **abcd** as the prefix. If only **delimiter** is set to **d**, objects **abcd** and **abcde** are grouped into a **CommonPrefixes** with **abcd** as the prefix, and **bbcde** is grouped separately into another **CommonPrefixes** with **bbcd** as the prefix. Type: string	No
key-marker	Position to start with when objects are listed Type: string Valid value: value of **NextKeyMarker** in the response body of the last request	No
version-id-marker	This parameter applies only to versioning objects. Specifies the version ID to start with when objects in a bucket are listed. Objects are listed in alphabetical order (a maximum of 1,000 objects are displayed at a time). This parameter is used together with the **key-marker** in the request. If the value of **version-id-marker** is not a version ID specified by **key-marker**, **version-id-marker** is invalid. Type: string Valid value: object version ID, that is, the value of **NextVersionIdMarker** in the response body of the last request	No

Request Headers

This request uses common request headers. **Table 3-3** lists the common request headers.

Request Elements

This request contains no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
```

```
x-obs-bucket-location: region
Content-Type: application/xml
Content-Length: length
<Response Body>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains the XML list of buckets owned by the user. **Table 5-5** describes the elements.

Table 5-5 Response elements

Element	Description
ListBucketResult	A list of objects in a bucket Type: XML
Contents	Object metadata Type: XML Ancestor: ListBucketResult
CommonPrefixes	Group information. If you specify a delimiter in the request, the response contains group information in **CommonPrefixes**. Type: XML Ancestor: ListBucketResult
Delimiter	The delimiter parameter specified in a request Type: string Ancestor: ListBucketResult

Element	Description
ETag	128-bit MD5 digest of the Base64 code of an object. ETag is the unique identifier of the object content. It can be used to identify whether the object content is changed. For example, if ETag value is **A** when an object is uploaded and the ETag value has changed to **B** when the object is downloaded, it indicates that the object content is changed. The actual ETag is the hash value of the object, which only reflects the changed content rather than the metadata. An uploaded object or copied object has a unique ETag after being encrypted using MD5. (If the object is encrypted on the server side, the ETag value is not the MD5 digest of the object, but the unique identifier calculated through server-side encryption.) Type: string Ancestor: ListBucketResult.Contents
Type	Object type. This parameter is returned when the object type is not **Normal**. Type: string Ancestor: ListBucketResult.Contents
ID	Tenant ID of the object owner Type: string Ancestor: ListBucketResult.Contents.Owner
IsTruncated	Determines whether the returned list of objects is truncated. The value **true** indicates that the list was truncated and **false** indicates that the list was not truncated. Type: boolean Ancestor: ListBucketResult
Key	Object name Type: string Ancestor: ListBucketResult.Contents

Element	Description
LastModified	Time (UTC) when an object was last modified Type: date Ancestor: ListBucketResult.Contents
Marker	Marker for the position from which objects in a bucket will be listed Type: string Ancestor: ListBucketResult
NextMarker	A marker for the last returned object in the list. **NextMarker** is returned when not all the objects are listed. You can set the **Marker** value to list the remaining objects in follow-up requests. Type: string Ancestor: ListBucketResult
MaxKeys	Maximum number of objects returned Type: string Ancestor: ListBucketResult
Name	Name of the requested bucket Type: string Ancestor: ListBucketResult
Owner	User information, including the domain ID and username Type: XML Ancestor: ListBucketResult.Contents
Prefix	Prefix of an object name. Only objects whose names have this prefix are listed. Type: string Ancestor: ListBucketResult
Size	Object size in bytes Type: string Ancestor: ListBucketResult.Contents

Table 5-6 Elements in the response message for listing multi-version objects.

Element	Description
ListVersionsResult	Container for the list of objects (including objects with multiple version IDs) Type: container
Name	Bucket name Type: string Ancestor: ListVersionsResult
Prefix	Prefix of an object name. Only objects whose names have this prefix are listed. Type: string Ancestor: ListVersionsResult
KeyMarker	Marker for the object key from which objects will be listed Type: string Ancestor: ListVersionsResult
VersionIdMarker	Object version ID to start with when objects are listed Type: string Ancestor: ListVersionsResult
NextKeyMarker	Key marker for the last returned object in the list. **NextKeyMarker** is returned when not all the objects are listed. You can set the **KeyMarker** value to list the remaining objects in follow-up requests. Type: string Ancestor: ListVersionsResult
NextVersionIdMarker	Version ID marker for the last returned object in the list. **NextVersionIdMarker** is returned when not all the objects are listed. You can set the **VersionIdMarker** value to list the remaining objects in follow-up requests. Type: string Ancestor: ListVersionsResult
MaxKeys	Maximum number of objects returned Type: string Ancestor: ListVersionsResult

Element	Description
IsTruncated	Indicates whether the returned list of objects is truncated. The value **true** indicates that the list was truncated and **false** indicates that the list was not truncated. Type: boolean Ancestor: ListVersionsResult
Version	Container of version information Type: container Ancestor: ListVersionsResult
DeleteMarker	Container for objects with deletion markers Type: container Ancestor: ListVersionsResult
Key	Object name Type: string Ancestor: ListVersionsResult.Version | ListVersionsResult.DeleteMarker
VersionId	Object version ID Type: string Ancestor: ListVersionsResult, Version | ListVersionsResult, DeleteMarker
IsLatest	Whether the object is the latest version. If the parameter value is **true**, the object is the latest version. Type: boolean Ancestor: ListVersionsResult.Version | ListVersionsResult.DeleteMarker
LastModified	Time (UTC) when an object was last modified Type: date Ancestor: ListVersionsResult.Version | ListVersionsResult.DeleteMarker

Element	Description
ETag	128-bit MD5 digest of the Base64 code of an object. ETag is the unique identifier of the object content. It can be used to identify whether the object content is changed. The actual ETag is the hash value of the object. For example, if ETag value is **A** when an object is uploaded and the ETag value has changed to **B** when the object is downloaded, it indicates that the object content is changed. The ETag only reflects the changed content rather than the metadata. An uploaded object or copied object has a unique ETag after being encrypted using MD5. Type: string Ancestor: ListVersionsResult.Version
Type	Object type. This parameter is returned when the object type is not **Normal**. Type: string Ancestor: ListVersionsResult.Version
Size	Object size in bytes Type: string Ancestor: ListVersionsResult.Version
Owner	User information, including the domain ID and username Type: container Ancestor: ListVersionsResult.Version | ListVersionsResult.DeleteMarker
ID	ID of the domain to which the object owner belongs Type: string Ancestor: ListVersionsResult.Version.Owner | ListVersionsResult.DeleteMarker.Owner
CommonPrefixes	Group information. If you specify a delimiter in the request, the response contains group information in **CommonPrefixes**. Type: container Ancestor: ListVersionsResult

Element	Description
Prefix	Indicates a different prefix in the group information in **CommonPrefixes**. Type: string Ancestor: ListVersionsResult.CommonPrefixes

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request 1

List all objects.

```
GET / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:28:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:KiyoYze4pmRNPYfmlXBfRTVxt8c=
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D34E379ABD93320CB9
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSXiN7GPL/yXM6OSBaYCUV1zcY5OelWp
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:23:30 GMT
Content-Length: 586

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Name>examplebucket</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>object001</Key>
    <LastModified>2015-07-01T00:32:16.482Z</LastModified>
    <ETag>"2fa3bcaaec668adc5da177e67a122d7c"</ETag>
    <Size>12041</Size>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
    </Owner>
  </Contents>
</ListBucketResult>
```

Sample Request 2

Filter objects.

A user has a bucket named **examplebucket**. The bucket has three objects named **newfile**, **obj001**, and **obj002**. If you want to view only object **obj002**, the request message format is as follows:

```
GET /examplebucket/?marker=obj002&prefix=obj HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:28:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:KiyoYze4pmRNPYfmlXBfRTVxt8c=
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D758FBA857E0801874
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCShn/xAyk/xHBX6qgGSB36WXrbco0X80
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:29:48 GMT
Content-Length: 707

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketResult xmlns="http://obs.example.com/doc/2015-06-30/">
<Name>examplebucket</Name>
<Prefix>obj</Prefix>
<Marker>obj002</Marker>
<MaxKeys>1000</MaxKeys>
<IsTruncated>false</IsTruncated>
  <Contents>
    <Key>obj002</Key>
    <LastModified>2015-07-01T02:11:19.775Z</LastModified>
    <ETag>"a72e382246ac83e86bd203389849e71d"</ETag>
    <Size>9</Size>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
    </Owner>
  </Contents>
</ListBucketResult>
```

Sample Request 3

Versioning

```
GET /?versions HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:29:45 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:iZeDESIMxBK2YODk7vIeVpyO8DI=
```

Sample Response 3

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D758FBA857E0801874
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCShn/xAyk/xHBX6qgGSB36WXrbco0X80
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:29:48 GMT
Content-Length: 707

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListVersionsResult xmlns="http://obs.example.com/doc/2015-06-30/">
 <Name>bucket02</Name>
  <Prefix/>
  <KeyMarker/>
  <VersionIdMarker/>
  <MaxKeys>1000</MaxKeys>
```

```
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>object001</Key>
    <VersionId>00011000000000013F16000001643A22E476FFFF9046024ECA3655445346485a</VersionId>
    <IsLatest>true</IsLatest>
    <LastModified>2015-07-01T00:32:16.482Z</LastModified>
    <ETag>"2fa3bcaaec668adc5da177e67a122d7c"</ETag>
    <Size>12041</Size>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
    </Owner>
  </Contents>
</ListVersionsResult>
```

5.1.4 Obtaining Bucket Metadata

Functions

This operation queries the metadata of a bucket. To use this operation, you must have the permission to read the bucket.

Request Syntax

```
HEAD / HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Table 5-7 lists the header fields required when obtaining CORS configuration information.

Table 5-7 Request headers for obtaining CORS configuration

Header	Description	Mandatory
Origin	Origin of the cross-domain request specified by the pre-request. Generally, it is a domain name. Type: string	Yes
Access-Control-Request-Headers	HTTP headers of a request. The request can use multiple HTTP headers. Type: string	No

Request Elements

This request contains no elements.

Response Syntax

```
HTTP/1.1 status_code
x-obs-bucket-location: region
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Table 5-8 lists additional response header parameters that are used except for the common response header parameters.

Table 5-8 Additional response header parameters

Header	Description
x-obs-bucket-location	The region where the bucket resides. Type: string
x-obs-version	OBS version of the bucket. Type: string
Access-Control-Allow-Origin	Indicates that the origin is included in the response if the origin in the request meets the CORS configuration requirements when CORS is configured for buckets. Type: string
Access-Control-Allow-Headers	Indicates that the headers are included in the response if headers in the request meet the CORS configuration requirements when CORS is configured for buckets. Type: string
Access-Control-Max-Age	Value of **MaxAgeSeconds** in the CORS configuration of the server when CORS is configured for buckets. Type: integer
Access-Control-Allow-Methods	Indicates that methods in the rule are included in the response if **Access-Control-Request-Method** in the request meets the CORS configuration requirements when CORS is configured for buckets. Type: string Valid values: **GET**, **PUT**, **HEAD**, **POST**, and **DELETE**.

Header	Description
Access-Control-Expose-Headers	Value of **ExposeHeader** in the CORS configuration of a server when CORS is configured for buckets. Type: string

Response Elements

This response involves no elements.

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request 1

No header field for obtaining CORS configuration is carried.

```
HEAD / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:30:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:niCQCuGIZpETKIyx1dttxHZyYlk=
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016439C734E0788404623FA8
Content-Type: application/xml
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSxwLpq9Hzf3OnaXr+pI/OPLKdrtiQAF
Date: WED, 01 Jul 2015 02:30:25 GMT
x-obs-bucket-location: region
x-obs-version: 3.0
Content-Length: 0
```

Sample Request 2

Obtain bucket metadata and CORS configuration information after CORS is configured for the bucket.

```
HEAD / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:30:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:niCQCuGIZpETKIyx1dttxHZyYlk=
Origin:www.example.com
Access-Control-Request-Headers:AllowedHeader_1
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016439C734E0788404623FA8
```

```
Content-Type: application/xml
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSxwLpq9Hzf3OnaXr+pI/OPLKdrtiQAF
Date: WED, 01 Jul 2015 02:30:25 GMT
x-obs-bucket-location: region
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT
Access-Control-Allow-Headers: AllowedHeader_1
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: ExposeHeader_1
x-obs-version: 3.0
Content-Length: 0
```

5.1.5 Obtaining Bucket Location

Functions

This operation obtains the location of a bucket. To use this operation, you must have the permission to read the bucket.

Request Syntax

```
GET /?location HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameters.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request contains no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Location xmlns="http://obs.region.example.com/doc/2015-06-30/">region</Location>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements of information about a bucket's region. **Table 5-9** describes the elements.

Table 5-9 Response elements

Element	Description
Location	Indicates the region where the bucket resides. Type: string

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?location HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:30:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:1DrmbCV+lhz3zV7uywlj7lrh0MY=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435D9F27CB2758E9B41A5
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSKWoJmaMyRXqofHgapbETDyI2LM9rUw
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:30:25 GMT
Content-Length: 128

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Location xmlns="http://obs.region.example.com/doc/2015-06-30/">region</Location>
```

5.1.6 Deleting Buckets

Functions

This operation deletes specified buckets. This operation can be performed only by the bucket owner and users who have been authorized (via a policy) with the permission to delete the bucket. The bucket to be deleted must be an empty bucket. If a bucket has an object or a multipart task, the bucket is not empty. You can list objects and multipart upload tasks in a bucket to check whether the bucket is empty.

Note:

If the server returns a **5XX** error or times out when a bucket is being deleted, the system needs to synchronize the bucket information. During this period, the bucket information may be inaccurate. Therefore, wait a while and then check whether the bucket is successfully deleted. If the bucket can still be queried, send the deletion request again.

Request Syntax

```
DELETE / HTTP/1.1
Host: bucketname.obs.region.example.com
```

```
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common request headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:31:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF260000016435DE6D67C35F9B969C47
x-obs-id-2: 32AAAQAAEAABKAAQAAEAABAAAQAAEAABCTukraCnXLsb7lEw4ZKjzDWWhzXdgme3
Date: WED, 01 Jul 2015 02:31:25 GMT
```

5.2 Advanced Bucket Settings

5.2.1 Configuring a Bucket Policy

Functions

This operation creates or modifies policies for buckets. If the specified bucket already has a policy, the policy in the request will overwrite the existing one. There is no limit on the number of bucket policies (statements) for a bucket. However, the total size of JSON descriptions of all bucket policies in a bucket cannot exceed 20 KB.

To perform this operation, the user must be the bucket owner or the bucket owner's IAM user that has permissions required for configuring bucket policies.

Request Syntax

```
PUT /?policy HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: signatureValue
Policy written in JSON
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

The request body is a JSON string containing bucket policy information.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details, see **Table 7-3**.

Sample Request 1

Grant permissions to an OBS tenant.

Grant permissions to the tenant whose ID is **783fc6652cf246c096ea836694f71855**.

For details about how to obtain the tenant ID, see **Obtaining the Domain ID and User ID**.

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:32:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Sid": "Stmt1375240018061",
         "Action": [
            "GetBucketLogging"
         ],
         "Effect": "Allow",
         "Resource": "logging.bucket",
         "Principal": {
            "ID": [
               "domain/783fc6652cf246c096ea836694f71855:user/*"
            ]
         }
      }
   ]
}
```

Sample Response 1

```
HTTP/1.1 204 No Content
x-obs-request-id: 7B6DFC9BC71DD58B061285551605709
x-obs-id-2: N0I2REZDOUJDNzFERDU4QjA2MTI4NTU1MTYwNTcwOUFBQUFBQUFBYmJiYmJiYmJD
Date: WED, 01 Jul 2015 02:32:25 GMT
Content-Length: 0
Server: OBS
```

Sample Request 2

Grant permissions to an OBS user.

The user ID is **71f3901173514e6988115ea2c26d1999**, and the account ID is **783fc6652cf246c096ea836694f71855**.

For details about how to obtain the account ID and user ID, see **Obtaining the Domain ID and User ID**.

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:33:28 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Sid": "Stmt1375240018062",
         "Action": [
            "PutBucketLogging"
         ],
         "Effect": "Allow",
         "Resource": "examplebucket",
         "Principal": {
            "ID": [
               "domain/783fc6652cf246c096ea836694f71855:user/71f3901173514e6988115ea2c26d1999"
            ]
```

```
        }
      }
    ]
}
```

Sample Response 2

```
HTTP/1.1 204 No Content
x-obs-request-id: 7B6DFC9BC71DD58B061285551605709
x-obs-id-2: N0I2REZDOUJDNzFERDU4QjA2MTI4NTU1MTYwNTcwOUFBQUFBQUFBYmJiYmJiYmJD
Date: WED, 01 Jul 2015 02:33:28 GMT
Content-Length: 0
Server: OBS
```

Sample Request 3

Deny all users except the specified one all the operation permissions.

The user ID is **71f3901173514e6988115ea2c26d1999**, and the account ID is **783fc6652cf246c096ea836694f71855**.

For details about how to obtain the account ID and user ID, see **Obtaining the Domain ID and User ID**.

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:34:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [
      {
         "Effect": "Deny",
         "Action": ["*"],
         "Resource": [
            "examplebucket/*",
            "examplebucket"
         ],
         "NotPrincipal": {
            "ID": [
               "domain/783fc6652cf246c096ea836694f71855:user/71f3901173514e6988115ea2c26d1999",
               "domain/783fc6652cf246c096ea836694f71855"
            ]
         }
      }
   ]
}
```

Sample Response 3

```
HTTP/1.1 204 No Content
x-obs-request-id: A603000001604A7DFE4A4AF31E301891
x-obs-id-2: BKOvGmTlt6sda5X4G89PuMO4fabObGYmnpRGkaMba1LqPt0fCACEuCMllAObRK1n
Date: WED, 01 Jul 2015 02:34:34 GMT
Content-Length: 0
Server: OBS
```

Sample Request 4

Request to allow only the specified domain name and external link requests that have no referer headers by using the URL validation whitelist.

URL validation whitelist: **http://storage.example.com**

```
PUT /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
```

```
Date: WED, 01 Jul 2015 02:34:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

{
   "Statement": [{
      "Effect": "Deny",
      "Action": [
      "GetObject",
      "GetObjectVersion"
      ],
      "Principal": {
         "ID": ["*"]
      },
      "Resource": ["examplebucket/*"],
      "Condition": {
         "StringNotLike": {
            "Referer": [
            "http://storage.example.com*",
            "${null}"
            ]
         }
      }
   }]
}
```

Sample Response 4

```
HTTP/1.1 204 No Content
x-obs-request-id: A603000001604A7DFE4A4AF31E301891
x-obs-id-2: BKOvGmTlt6sda5X4G89PuMO4fabObGYmnpRGkaMba1LqPt0fCACEuCMllAObRK1n
Date: WED, 01 Jul 2015 02:34:34 GMT
Content-Length: 0
Server: OBS
```

5.2.2 Obtaining Bucket Policy Information

Functions

This operation uses the sub-resources of policy to return the policy information of a specified bucket.

To perform this operation, the user must be the bucket owner or the bucket owner's IAM user that has permissions required for obtaining bucket policies.

This operation cannot be performed in the following scenarios, and the 404 error code "NoSuchBucketPolicy" is returned:

- The specified bucket policy does not exist.
- The standard bucket policy is set to **Private** and no custom bucket policy is configured.

Request Syntax

```
GET /?policy HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: application/xml
Date: date
Policy Content
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

The response body is a JSON string containing bucket policy information.

Error Responses

No special error responses are returned. For details, see **Table 7-3**.

Sample Request

```
GET /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:35:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

Sample Response

```
HTTP/1.1 200 OK
x-obs-request-id: A603000001604A7DFE4A4AF31E301891
x-obs-id-2: BKOvGmTlt6sda5X4G89PuMO4fabObGYmnpRGkaMba1LqPt0fCACEuCMllAObRK1n
Date: WED, 01 Jul 2015 02:35:46 GMT
Content-Length: 509
Server: OBS

{
   "Statement":[
      {
         "Sid":"Stmt1375240018061",
         "Effect":"Allow",
         "Principal":{
            "ID":[
               "domain/domainiddomainiddomainiddo006666:user/useriduseriduseriduseridus004001",
               "domain/domainiddomainiddomainiddo006667:user/*"
            ]
         },
         "Action":[
            "*"
         ],
         "Resource":[
            "examplebucket"
         ]
      }
   ]
}
```

5.2.3 Deleting a Bucket Policy

Functions

This operation uses the policy sub-resources to delete the policy of a specified bucket.

To perform this operation, the user must be the bucket owner or the bucket owner's IAM user that has permissions required for deleting bucket policies.

The 204 error code "No Content" is returned regardless of whether a requested bucket policy exists or not.

Request Syntax

```
DELETE /?policy HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: text/xml
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details, see **Table 7-3**.

Sample Request

```
DELETE /?policy HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 02:36:06 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

Sample Response

```
HTTP/1.1 204 No Content
x-obs-request-id: 9006000001643AAAF70BF6152D71BE8A
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSB4oWmNX3gVGGLr1cRPWjOhffEbq1XV
Date: WED, 01 Jul 2015 02:36:06 GMT
Server: OBS
```

5.2.4 Configuring a Bucket ACL

Functions

This operation controls access permissions for buckets. By default, only the creator of a bucket has the permission to read and write the bucket. You can also set other access permissions. For example, you can set a public read policy to grant the read permission to all users.

You can configure an ACL when creating a bucket, and modify or obtain the ACLs of existing buckets using the API operations. A bucket ACL supports a maximum of 100 grants.

Request Syntax

```
PUT /?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-Type: application/xml
Content-Length: length

<AccessControlPolicy>
   <Owner>
      <ID>ID</ID>
   </Owner>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>domainId</ID>
         </Grantee>
         <Permission>permission</Permission>
         <Delivered>false</Delivered>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

Request Parameters

This request contains no parameters.

Request Headers

You can change the ACL of a bucket by using the header settings. Each ACL configured with the header setting has a set of predefined grantees and authorized permissions. If you want to authorize access permissions by adding the header to a request, you must add the following header and specify the value.

Table 5-10 Optional header for specifying canned ACLs

Name	Description	Mandatory
x-obs-acl	Uses the canned ACL for a bucket. Value options: **private** | **public-read** | **public-read-write** | **public-read-delivered** | **public-read-write-delivered** Type: string	No

Request Elements

This request carries ACL information in elements to specify an ACL. **Table 3-3** describes the elements.

Table 5-11 Additional request elements

Element	Description	Mandatory
Owner	Bucket owner information, including the ID Type: XML	Yes
ID	Account ID of the authorized user Type: string	Yes
Grant	Container for the grantee and the granted permissions A single bucket ACL can contain no more than 100 grants. Type: XML	No
Grantee	Grantee information Type: XML	No
Canned	Grants permissions to all users. Value range: Everyone Type: Enumeration	No
Delivered	Indicates whether the bucket ACL is applied to all objects in the bucket. Type: boolean The default value is **false**.	No

Element	Description	Mandatory
Permission	Permissions to be granted Value options: **READ** | **WRITE** | **FULL_CONTROL** Type: Enumeration	No
AccessControlList	Indicates an ACL, which consists of three elements: **Grant**, **Grantee**, and **Permission**. Type: XML	Yes

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details, see **Table 7-3**.

Sample Request

```
PUT /?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:37:22 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:iqSPeUBl66PwXDApxjRKk6hlcN4=
Content-Length: 727

<AccessControlPolicy xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
      <Delivered>false</Delivered>
    </Grant>
```

```
  <Grant>
    <Grantee>
      <Canned>Everyone</Canned>
    </Grantee>
    <Permission>READ_ACP</Permission>
  </Grant>
 </AccessControlList>
</AccessControlPolicy>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164361F2954B4D063164704
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCT78HTIBuhe0FbtSptrb/akwELtwyPKs
Date: WED, 01 Jul 2015 02:37:22 GMT
Content-Length: 0
```

5.2.5 Obtaining Bucket ACL Information

Functions

This operation returns the ACL information of a bucket. To obtain the ACL of a bucket, you need to have the **READ_ACP** or **FULL_CONTROL** permission for the bucket.

Request Syntax

```
GET /?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Owner>
      <ID>id</ID>
   </Owner>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>id</ID>
```

```
        </Grantee>
        <Permission>permission</Permission>
        <Delivered>false</Delivered>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response returns information (in the form of elements) about the bucket ACL. **Table 5-12** describes the elements.

Table 5-12 Response elements

Element	Description
Owner	Bucket owner Type: XML
ID	Account ID Type: string
AccessControlList	Indicates the ACL that records all users who have permissions to access the bucket and the permissions granted to the users. Type: XML
Grant	Container for the grantee and the granted permissions Type: XML
Grantee	Grantee information Type: XML
Canned	Grants permissions to all users. Type: Enumeration The value must be **Everyone**.
Delivered	Indicates whether the bucket ACL is applied to objects in the bucket. Type: boolean
Permission	Grantee's permission for a bucket Type: string

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:39:28 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:X7HtzGsIEkzJbd8vo1DRu30vVrs=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436B69D82F14E93528658
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSjTh8661+HF5y8uAnTOBIpNO133hji+
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:39:28 GMT
Content-Length: 784

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
      <Delivered>false</Delivered>
    </Grant>
    <Grant>
      <Grantee>
        <Canned>Everyone</Canned>
      </Grantee>
      <Permission>READ_ACP</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

5.2.6 Configuring Logging for a Bucket

Functions

When a bucket is created, the logging function is not enabled by default. To generate logs recording operations on buckets, you need to enable the logging function for the bucket. After the logging function is enabled, a log is generated for each operation on a bucket and multiple logs are packed into a log file. When enabling the logging function, you need to specify a location where log files are stored. They can be stored in the bucket for which the logging is enabled, or in other buckets that you have the required permissions. However, the bucket where

log files are stored and the bucket for which the logging is enabled must be in the same region.

Log files are generated by OBS and uploaded to the bucket where logs are stored. Therefore, OBS needs to be authorized to upload generated log files. Before configuring the logging function, you need to create an agency for OBS in IAM, the agency name is configured as a parameter of the bucket, and the logging function must be configured under the **LoggingEnabled** tag in the XML file. You only need to authorize the agency with the upload permissions for the target bucket.

Example of agency permissions

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Action": [
                "obs:object:PutObject"
            ],
            "Resource": [
                "OBS:*:*:object:mybucketlogs/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

To disable the bucket logging function, upload a logging file with an empty **BucketLoggingStatus** tag.

Request Syntax

```
PUT /?logging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: signatureValue
<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus>
  <Agency>agency-name</Agency>
  <LoggingEnabled>
    <TargetBucket>mybucketlogs</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>
    <TargetGrants>
      <Grant>
        <Grantee>
          <ID>domainID</ID>
        </Grantee>
        <Permission>READ</Permission>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

Table 5-13 Request elements

Element	Description	Mandatory
BucketLoggingStatus	Container for logging status information Type: container	Yes
Agency	Name of the IAM agency created by the owner of the target bucket on IAM. Type: string	You must set this parameter when enabling the logging function. Do not set this parameter when disabling the logging function.
LoggingEnabled	Container for logging information. Present this element when enabling the logging function. Otherwise, absent it. You can add specific logging information in this element. Type: container	You must set this parameter when enabling the logging function. Do not set this parameter when disabling the logging function.
Grant	Container for the grantee and the grantee's logging permissions. It describes who has the permission to access the generated log files. Type: container	No
Grantee	Container for the user that is granted with the logging permission. Type: container	No
ID	Account ID of the authorized user, which is globally unique. Type: string	No

Element	Description	Mandatory
Permission	Permissions of the grantee to the generated logs. Type: string Value options: **FULL_CONTROL** | **READ** | **WRITE**	No
TargetBucket	When enabling the logging function, the owner of the bucket being logged can specify a target bucket to store the generated log files. Ensure that the bucket owner who configures the logging function has the **FULL_CONTROL** permission for the bucket that stores log files. Log files generated for multiple buckets can be stored in the same target bucket. If you do so, you need to specify different TargetPrefixes to classify logs for different buckets. Type: string	You must set this parameter when enabling the logging function. Do not set this parameter when disabling the logging function.
TargetPrefix	You can specify a prefix using this element so that log files are named with this prefix. Type: string	You must set this parameter when enabling the logging function. Do not set this parameter when disabling the logging function.
TargetGrants	Container for granting information. Type: container	No

Naming rules for access logs

```
<TargetPrefix>YYYY-mm-DD-HH-MM-SS-<UniqueString>
```

- <*TargetPrefix*> is the log name prefix specified by the user.
- YYYY-mm-DD-HH-MM-SS indicates the date and time when the log is generated.

- *<UniqueString>* indicates a character string generated by OBS.

The following is an example of a log file name:

```
bucket-log2015-06-29-12-22-07-N7MXLAF1BDG7MPDV
```

- **bucket-log** is the target prefix specified by the user.
- **2015-06-29-12-22-07** indicates the time when the log is generated.
- **N7MXLAF1BDG7MPDV** is a string automatically generated by OBS

Format of bucket access logs

The following shows an access log delivered to the target bucket:

```
787f2f92b20943998a4fe2ab75eb09b8 bucket [13/Aug/2015:01:43:42 +0000] xx.xx.xx.xx
787f2f92b20943998a4fe2ab75eb09b8 281599BACAD9376ECE141B842B94535B
REST.GET.BUCKET.LOCATION - "GET /bucket?location HTTP/1.1" 200 - 211 - 6 6 "-"  "HttpClient" - -
```

Each access log contains the following information:

Table 5-14 Format of bucket access logs

Parameter	Example	Description
BucketOwner	787f2f92b20943998a4fe2ab75eb09b8	ID of the bucket owner
Bucket	bucket	Bucket name
Time	[13/Aug/2015:01:43:42 +0000]	Request timestamp
Remote IP	xx.xx.xx.xx	Request IP address
Requester	787f2f92b20943998a4fe2ab75eb09b8	ID of the requester
RequestID	281599BACAD9376ECE141B842B94535B	Request ID
Operation	REST.GET.BUCKET.LOCATION	Operation
Key	-	Object name
Request-URI	GET /bucket?location HTTP/1.1	Request URI
HTTPStatus	200	Response code
ErrorCode	-	Error code
BytesSent	211	Size of the HTTP response, expressed in bytes
ObjectSize	-	Object size
TotalTime	6	Processing time on the server Unit: ms

Parameter	Example	Description
Turn-AroundTime	6	Total request processing time Unit: ms
Referer	-	Referer header of the request
User-Agent	HttpClient	User-Agent header of the request
VersionID	-	Version ID contained in a request
STSLogUrn	-	Federated authentication and agency information

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /?logging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:40:06 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mCOjER/L4ZZUY9qr6AOnkEiwvVk=
Content-Length: 528

<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus>
  <Agency>agencyGrantPutLogging</Agency>
  <LoggingEnabled>
    <TargetBucket>log-bucket</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>
    <TargetGrants>
```

```
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </TargetGrants>
 </LoggingEnabled>
</BucketLoggingStatus>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643663CE53B6AF31C619FD
x-obs-id-2: 32AAAQAAEAABSAAkpAIAABAAAQAAEAABCT9CjuOx8cETSRbqkm35s1dL/tLhRNdZ
Date: WED, 01 Jul 2015 02:40:06 GMT
Content-Length: 0
```

5.2.7 Obtaining a Bucket Logging Configuration

Functions

This operation queries the logging status of a bucket. It uses the logging sub-resource to return the logging status of a bucket.

Only the bucket owner or users granted the **GetBucketLogging** permission can query the bucket logging status.

Request Syntax

```
GET /?logging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Agency>agency-name</Agency>
<LoggingEnabled>
   <TargetBucket>bucketName</TargetBucket>
   <TargetPrefix>prefix</TargetPrefix>
```

```
    <TargetGrants>
      <Grant>
        <Grantee>
          <ID>id</ID>
        </Grantee>
        <Permission>permission</Permission>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to specify the bucket logging status. **Table 5-15** describes the elements.

Table 5-15 Response elements

Element	Description
BucketLoggingStatus	Container for logging status information Type: container
Agency	Name of the agency created by the owner of the logging bucket for uploading log files by OBS Type: string
LoggingEnabled	Container for logging information. This element enables or disables the logging function. Present this element when enabling the logging. Otherwise, absent it. Type: container
Grant	Container for the grantee and the granted permissions Type: container
Grantee	Container for the user that is granted with the logging permission Type: container
ID	Grantee domain ID, a globally unique ID Type: string

Element	Description
Permission	Logging permission granted to the grantee for a bucket. The bucket owner is automatically granted the **FULL_CONTROL** permission when creating the bucket. Logging permissions control access to different logs. Type: string Value options: **FULL_CONTROL** | **READ** | **WRITE**
TargetBucket	When enabling the logging function, the owner of the bucket being logged can specify a target bucket to store the generated log files. Log files generated for multiple buckets can be stored in the same target bucket. If you do so, you need to specify different TargetPrefixes to classify logs for different buckets. Type: string
TargetPrefix	You can specify a prefix using this element so that log files are named with this prefix. Type: string
TargetGrants	Container for granting information Type: container

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?logging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 02:42:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:hUk+jTnR07hcKwJh4ousF2E1U3E=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436B8EEE7FBA2AA3335E3
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCShuQJoWFpS77C8bOv1mqURv0UY+0ejx
Content-Type: application/xml
Date: WED, 01 Jul 2015 02:42:46 GMT
Content-Length: 429

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<BucketLoggingStatus xmlns="http://obs.example.com/doc/2015-06-30/">
  <Agency>agency-name</Agency>
  <LoggingEnabled>
    <TargetBucket>log-bucket</TargetBucket>
    <TargetPrefix>mybucket-access_log-/</TargetPrefix>
```

```
  <TargetGrants>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </TargetGrants>
 </LoggingEnabled>
</BucketLoggingStatus>
```

5.2.8 Configuring Bucket Lifecycle Rules

Functions

This operation configures lifecycle rules that can delete objects from a bucket at a specified time. Typical application scenarios:

- Delete periodically uploaded files. Some files uploaded periodically need only to be retained for only one week or one month.
- Delete files that are frequently accessed within a certain period of time but are seldom accessed afterward. You can archive these files and then schedule the time for deletion.

You can perform this operation to create or update the lifecycle configuration of a bucket.

NOTE

Objects are permanently deleted upon the expiration of their lifecycle time, and the deleted objects cannot be restored.

To perform this operation, you must have the **PutLifecycleConfiguration** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

The lifecycle configuration enables OBS to delete objects at a scheduled time. To prevent a user from doing so, the following permissions granted to the user must be revoked:

- DeleteObject
- DeleteObjectVersion
- PutLifecycleConfiguration

If you want to forbid a user to set the bucket lifecycle configuration, revoke the **PutLifecycleConfiguration** permission from the user.

Request Syntax

```
PUT /?lifecycle HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
Content-MD5: MD5
<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration>
   <Rule>
      <ID>id</ID>
```

```
    <Prefix>prefix</Prefix>
    <Status>status</Status>
    <Expiration>
       <Days>days</Days>
    </Expiration>
    <NoncurrentVersionExpiration>
       <NoncurrentDays>days</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>
```

Request Parameters

This request contains no parameter.

Request Headers

Table 5-16 lists the request header.

Table 5-16 Request headers

Header	Description	Mandatory
Content-MD5	Base64-encoded 128-bit MD5 digest of the message according to RFC 1864. Type: string Example: n58IG6hfM7vqI4K0vnWpog==	Yes

Request Elements

In this request, you need to specify the lifecycle configuration in the request body. The lifecycle configuration can be uploaded in the form of an XML file with elements described in **Table 5-17**.

- If the versioning of a bucket is enabled or suspended, you can set **NoncurrentVersionExpiration** to control the lifecycle of historical object versions. The lifecycle of a historical version depends on the time when it becomes a historical one (time when the version is replaced by a new version) and the value of **NoncurrentDays**. , if **NoncurrentDays** is set to **1**, an object version will be deleted only after it becomes a historical one for one day. If the version V1 of object A is created on the first date of a month and new version V2 is uploaded on the fifth date of the month, V1 becomes a historical version. At 00:00 on the seventh date of the month, V1 will expire. The deletion of the object after the expiration time may be delayed. The delay is within 48 hours.
- Objects are processed according to the following procedures, if their latest versions meet the expiration rule and versioning is enabled or suspended for the bucket.
  - Versioning enabled:

    If the latest version of the object does not have the DeleteMarker, the object generates a new DeleteMarker.

    If the object of the latest version has the DeleteMarker and the object has this version only, this version will be deleted.

If the object of the latest version has the DeleteMarker and the object has other versions, all versions of the object remain unchanged.

- Versioning suspended:

  If the latest version of the object does not have the DeleteMarker and is not the null version, the object generates a new DeleteMarker for the null version.

  If the latest version of the object does not have the DeleteMarker but is the null version, this null version is overwritten by a new DeleteMarker generated for the null version.

  If the object of the latest version has the DeleteMarker and the object has this version only, this version will be deleted.

  If the object of the latest version has the DeleteMarker and the object has other versions, all versions of the object remain unchanged.

Table 5-17 Response elements for lifecycle configuration

Name	Description	Mandatory
Date	Specifies that OBS executes lifecycle rules for objects before the specified date. The date must be compliant with the ISO8601 format, and the time must be compliant with the UTC format of 00:00:00. For example: **2018-01-01T00:00:00.000Z**, which indicates that objects whose last modification time is earlier than **2018-01-01T00:00:00.000Z** are deleted. Objects whose last modification time is equal to or later than the specified time are not deleted. Type: string Ancestor node: Expiration	Required if the **Days** element is absent.
Days	Specifies the number of days (since the latest update to the latest object version) after which the lifecycle rule takes effect. Type: positive integer Ancestor node: Expiration	Required if the **Date** element is absent.
Expiration	Container for the object expiration rule (only applicable to the latest versions of objects). Type: XML Children node: Date or Days Ancestor node: Rule	Yes

Name	Description	Mandatory
ID	Unique identifier of a rule. The value can contain a maximum of 255 characters. Type: string Ancestor node: Rule	No
LifecycleConfiguration	Container for lifecycle rules. You can add multiple rules. The total size of the rules cannot exceed 20 KB. Type: XML Children node: Rule Ancestor node: none	Yes
NoncurrentDays	Number of days when the specified rule takes effect after the object becomes a historical version (only applicable to an object's historical version). Type: positive integer Ancestor node: NoncurrentVersionExpiration	Required if the **NoncurrentVersionExpiration** element is present.
NoncurrentVersionExpiration	Container for the expiration time of objects' historical versions. If versioning is enabled or suspended for a bucket, you can set **NoncurrentVersionExpiration** to delete historical versions of objects that match the lifecycle rule (only applicable to the historical versions of objects). Type: XML Children node: NoncurrentDays Ancestor node: Rule	No
Prefix	Object name prefix identifying one or more objects to which the rule applies. Type: string Ancestor node: Rule	Yes
Rule	Container for a specific lifecycle rule. Type: container Ancestor node: LifecycleConfiguration	Yes
Status	Indicates whether the rule is enabled. Type: string Ancestor node: Rule Value options: Enabled, Disabled	Yes

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /?lifecycle HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:05:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:DpSAlmLX/BTdjxU5HOEwflhM0WI=
Content-MD5: ujCZn5p3fmczNiQQxdsGaQ==
Content-Length: 919

<?xml version="1.0" encoding="utf-8"?>
<LifecycleConfiguration>
  <Rule>
    <ID>delete-2-days</ID>
    <Prefix>test/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>70</Days>
    </Expiration>
    <NoncurrentVersionExpiration>
      <NoncurrentDays>70</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643670AC06E7B9A7767921
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSvK6z8HV6nrJh49gsB5vqzpgtohkiFm
Date: WED, 01 Jul 2015 03:05:34 GMT
Content-Length: 0
```

5.2.9 Obtaining Bucket Lifecycle Configuration

Functions

This operation obtains the bucket lifecycle configuration.

To perform this operation, you must have the **GetLifecycleConfiguration** permission. By default, only the bucket owner can perform this operation. The

bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

Request Syntax

```
GET /?lifecycle HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<LifecycleConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Rule>
      <ID>id</ID>
      <Prefix>prefix</Prefix>
      <Status>status</Status>
      <Expiration>
         <Date>date</Date>
      </Expiration>
      <NoncurrentVersionExpiration>
         <NoncurrentDays>days</NoncurrentDays>
      </NoncurrentVersionExpiration>
   </Rule>
</LifecycleConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to detail the configuration. **Table 5-18** describes the elements.

Table 5-18 Response elements for lifecycle configuration

Element	Description
Date	Specifies that OBS executes lifecycle rules for objects before the specified date. The date must be compliant with the ISO8601 format, and the time must be compliant with the UTC format of 00:00:00. For example: **2018-01-01T00:00:00.000Z**, which indicates that objects whose last modification time is earlier than **2018-01-01T00:00:00.000Z** are deleted. Objects whose last modification time is equal to or later than the specified time are not deleted. Type: string Ancestor node: Expiration
Days	Specifies the number of days (since the latest update to the latest object version) after which the lifecycle rule is executed. Type: positive integer Ancestor node: Expiration
Expiration	Container for the object expiration rule. Type: XML Children node: Date or Days Ancestor node: Rule
ID	Unique identifier of a rule. The value can contain a maximum of 255 characters. Type: string Ancestor node: Rule
LifecycleConfiguration	Container for lifecycle rules. You can add multiple rules. The total size of the rules cannot exceed 20 KB. Type: XML Children node: Rule Ancestor node: none
NoncurrentDays	Number of days when the specified rule takes effect after the object becomes a historical version. Type: positive integer Ancestor node: NoncurrentVersionExpiration

Element	Description
NoncurrentVersionExpiration	Container for the expiration time of objects' historical versions. If versioning is enabled or suspended for a bucket, you can set **NoncurrentVersionExpiration** to delete objects whose life cycles have expired. Type: XML Children node: NoncurrentDays Ancestor node: Rule
Prefix	Object name prefix identifying one or more objects to which the rule applies. Type: string Ancestor node: Rule
Rule	Container for a specific lifecycle rule. Type: container Ancestor node: LifecycleConfiguration
Status	Indicates whether the rule is enabled. Type: string Ancestor node: Rule Value options: Enabled, Disabled

Error Responses

Table 5-19 describes possible special errors in the request.

Table 5-19 Special error

Error Code	Description	HTTP Status Code
NoSuchLifecycleConfiguration	The bucket lifecycle configuration does not exist.	404 Not Found

For other errors, see **Table 7-3**.

Sample Request

```
GET /?lifecycle HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:06:56 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:/Nof9FCNANfzIXDS0NDp1IfDu8I=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436BA5684FF5A10370EDB
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSEMKZSIeboCA1eAukgYOOAd7oX3ZONn
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:06:56 GMT
Content-Length: 919

<?xml version="1.0" encoding="utf-8"?>
<LifecycleConfiguration>
  <Rule>
    <ID>delete-2-days</ID>
    <Prefix>test/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>2</Days>
    </Expiration>
    <NoncurrentVersionExpiration>
      <NoncurrentDays>5</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>
```

5.2.10 Deleting Lifecycle Rules

Functions

This operation deletes the lifecycle configuration of a bucket. After the lifecycle configuration of a bucket is deleted, OBS will not automatically delete objects in that bucket.

To perform this operation, you must have the **PutLifecycleConfiguration** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

Request Syntax

```
DELETE /?lifecycle HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: Authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
```

```
Content-Type: text/xml
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE /?lifecycle HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:12:22 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:5DGAS7SBbMC1YTC4tNXY57Zl2Fo=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF260000016436C2550A1EEA97614A98
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSB7A0KZEBOCutgcfZvaGVthTGOJSuyk
Date: WED, 01 Jul 2015 03:12:22 GMT
```

5.2.11 Configuring Versioning for a Bucket

Functions

This operation restores an object that is mistakenly overwritten or deleted. You can use versioning to save, query, and restore objects of different versions. Versioning allows you to easily recover lost data due to misoperations or program faults. Versioning can also be used for retaining and archiving data.

By default, versioning is disabled for a bucket.

You can perform this operation to enable or suspend versioning for a bucket.

After versioning is enabled for a bucket:

- OBS creates a unique version ID for each uploaded object. Namesake objects are not overwritten and are distinguished by their own version IDs.
- You can download objects by specifying version IDs. By default, the latest object is downloaded if the version ID is not specified.
- Objects can be deleted by version ID. If an object is deleted with no version ID specified, the object is only attached with a deletion marker and a unique version ID but is not physically deleted.

- The latest objects in a bucket are returned by default after a GET Object request. You can also send a request to obtain a bucket's objects with all version IDs.

After versioning is suspended for a bucket:

- Existing objects with version IDs are not affected.
- The system creates version ID **null** to an uploaded object and the object will be overwritten after a namesake one is uploaded.
- You can download objects by specifying version IDs. By default, the latest object is downloaded if the version ID is not specified.
- Objects can be deleted by version ID. If an object is deleted with no version ID specified, the object is attached with a deletion marker whose version ID is **null**. The object with version ID **null** is physically deleted.

Only the bucket owner can set versioning for the bucket.

Request Syntax

```
PUT /?versioning HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-Length: length

<VersioningConfiguration>
   <Status>status</Status>
</VersioningConfiguration>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request contains elements to configure the bucket versioning in XML format. **Table 5-20** lists the request elements.

Table 5-20 Elements for configuring bucket versioning

Element	Description	Mandatory
VersioningConfiguration	Root node for configuring versioning Ancestor node: none	Yes
Status	Versioning status of the bucket Type: enumeration Ancestor node: VersioningConfiguration Value options: Enabled, Suspended	Yes

Response Syntax

```
HTTP/1.1 status_code
Date: date

Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /?versioning HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:14:18 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:sc2PM13Wlfcoc/YZLK0MwsI2Zpo=
Content-Length: 89

<VersioningConfiguration>
   <Status>Enabled</Status>
</VersioningConfiguration>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643672B973EEBC5FBBF909
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSH6rPRHjQCa62fcNpCCPs7+1Aq/hKzE
Date: Date: WED, 01 Jul 2015 03:14:18 GMT
Content-Length: 0
```

5.2.12 Obtaining Bucket Versioning Status

Functions

This operation allows a bucket owner to get the versioning status of the bucket.

If versioning is not configured for a bucket, no versioning status information will be returned following this operation.

Request Syntax

```
GET /?versioning HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length

<VersioningConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Status>status</Status>
</VersioningConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to specify the bucket versioning status. **Table 5-21** describes the elements.

Table 5-21 Response elements

Element	Description
VersioningConfiguration	Element of versioning status information. Type: element
Status	Versioning status of the bucket. Type: enumeration Value options: Enabled, Suspended

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?versioning HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
```

```
Date: WED, 01 Jul 2015 03:15:20 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:4N5qQIoluLO9xMY0m+8lIn/UWXM=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436BBA4930622B4FC9F17
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQIrNJ5/Ag6EPN8DAwWlPWgBc/xfBnx
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:15:20 GMT
Content-Length: 180

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VersioningConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

5.2.13 Configuring Event Notification for a Bucket

Functions

This operation notifies users of their operations on buckets, allowing users know events happened on buckets in a timely manner.

By default, the notification function of a bucket is not enabled, and the **NotificationConfiguration** element is **null**. If you want to disable the function, set the **NotificationConfiguration** element to **null**.

```
<NotificationConfiguration>
</NotificationConfiguration>
```

After receiving a request for configuring event notification, OBS verifies whether the specified SMN topic exists and whether the topic is authorized to OBS. If the topic exists and is authorized to OBS, OBS sends a test notification to the topic subscriber.

To perform this operation, you must have the **PutBucketNotification** permission. By default, the permission is granted to the bucket owner only. However, it can be granted to other users by configuring the bucket policy.

Request Syntax

```
PUT /?notification HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string

<NotificationConfiguration>
   <TopicConfiguration>
      <Id>ConfigurationId</Id>
      <Filter>
         <Object>
            <FilterRule>
               <Name>prefix</Name>
               <Value>prefix-value</Value>
            </FilterRule>
            <FilterRule>
               <Name>suffix</Name>
               <Value>suffix-value</Value>
            </FilterRule>
         </Object>
      </Filter>
      <Topic>TopicARN</Topic>
```

```
    <Event>event-type</Event>
    <Event>event-type</Event>
    ...
  </TopicConfiguration>
  ...
</NotificationConfiguration>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request contains elements to specify the notification configuration for the bucket in XML format. For details about the configuration elements, see **Table 5-22**.

Table 5-22 Request elements for notification function configuration

Element	Description	Mandatory
NotificationConfiguration	Root element for configuring the event notification function of a bucket. If the sub element is **null**, the function is disabled. Type: element Ancestor: none Children: no or multiple TopicConfigurations	Yes
TopicConfiguration	Element for configuring the event notification topic. Type: element Ancestor: NotificationConfiguration Children: Id, Filter, Topic, Event, or Events	No

Element	Description	Mandatory
Topic	URN of the event notification topic. When OBS detects a specific event in the bucket, it publishes a notification message to the topic. The topic value can be found in SMN topics. Type: string Ancestor: TopicConfiguration Template: <Topic>urn:smn:region:project_id:smn_topic</Topic> Example: <Topic>urn:smn:eu-de:d745b885f14941369b2d2138e7a65bef:obs_test</Topic>	Required if **TopicConfiguration** is added
Id	Unique ID of each event notification. If the user does not specify an ID, the system assigns an ID automatically. Type: string Ancestor: TopicConfiguration	No
Filter	Element used to store rules of filtering object names. Type: element Ancestor: TopicConfiguration Children: Object	No
Object	Element that defines the filtering rule. The rule filters objects based on the prefixes and suffixes of object names. Type: element Ancestor: Filter Children: one or more FilterRules	No
FilterRule	Element that defines key-value pairs of the filtering rule Type: element Ancestor: Object Children: Name, Value	No
Name	Prefix or suffix of object names for filtering Type: string Ancestor: FilterRule Value options: prefix, suffix	No

Element	Description	Mandatory
Value	Key word of object names. Based on the prefix or suffix defined by **Name**, enter the key word for filtering objects. A longer string of characters delivers a more accurate filtering result. A maximum of 1024 characters are supported. Type: string Ancestor: FilterRule	No
Event	Type of events that need to be notified **NOTE** Multiple event types can be added in one TopicConfiguration item. Type: string Value options: The following values can be used to upload an object: • ObjectCreated:Put • ObjectCreated:Post • ObjectCreated:Copy • ObjectCreated:CompleteMultipartUpload Or use wildcard characters to support all upload operations: • ObjectCreated:* The following values can be used to delete an object: • ObjectRemoved:Delete • ObjectRemoved:DeleteMarkerCreated Or use wildcard characters to support all delete operations: • ObjectRemoved:* Ancestor: TopicConfiguration	Required if **TopicConfiguration** is added

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Content-Type: type
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

When this operation is being called, the system checks whether the **NotificationConfiguration** element is valid and whether the configuration is valid. The following table lists the common errors and possible causes of this operation.

Table 5-23 Error codes and possible causes

Error Code	Description	HTTP Status Code
InvalidArgument	Possible causes of this error are: • The specified event is not supported. • The specified URN does not exist or is incorrect. • The specified region in the URN is different as the region where the bucket resides. • The specified filtering rules overlap.	400 Bad Request
AccessDenied	The operator is not the bucket owner and not granted with the **PutBucketNotification** permission.	403 Forbidden

Sample Request

```
PUT /?notification HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:15:45 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uRTt8YTkAqJCUfWfYkveEcIGAC0=
Content-Length: 538

<NotificationConfiguration>
  <TopicConfiguration>
    <Id>ConfigurationId</Id>
    <Filter>
      <Object>
        <FilterRule>
          <Name>prefix</Name>
          <Value>object</Value>
        </FilterRule>
        <FilterRule>
          <Name>suffix</Name>
          <Value>txt</Value>
```

```
      </FilterRule>
    </Object>
  </Filter>
  <Topic>urn:smn:region:4b29a3cb5bd64581bda5714566814bb7:tet555</Topic>
  <Event>ObjectCreated:Put</Event>
 </TopicConfiguration>
</NotificationConfiguration>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 9046000001643C8E80C19FAC4D8068E3
x-obs-id-2: 32AAAQAAEAABSAAkgAIAABAAAQAAEAABCTFAxJPTib3GkcQ7nVVs4C8Z6NNcfVDu
Date: WED, 01 Jul 2015 03:15:46 GMT
Content-Length: 0
```

5.2.14 Obtaining the Event Notification Configuration of a Bucket

Functions

This operation obtains the notification configuration of a bucket.

To perform this operation, you must have the **GetBucketNotification** permission. By default, the permission is granted to the bucket owner only. However, it can be granted to other users by configuring the bucket policy or user policy.

Request Syntax

```
GET /?notification HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: type
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8"?>
<NotificationConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
   <TopicConfiguration>
      <Id>ConfigurationId</Id>
      <Filter>
         <Object>
```

```
            <FilterRule>
               <Name>prefix</Name>
               <Value>prefix-value</Value>
            </FilterRule>
            <FilterRule>
               <Name>suffix</Name>
               <Value>suffix-value</Value>
            </FilterRule>
         </Object>
      </Filter>
      <Topic>TopicARN</Topic>
      <Event>event-type</Event>
      <Event>event-type</Event>
      ...
   </TopicConfiguration>
</NotificationConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to detail the configuration. **Table 5-24** describes the elements.

Table 5-24 Response elements for configuring event notification

Element	Description
NotificationConfiguration	Element for configuring the event notification function of a bucket. If this element is **null**, the function is disabled. Type: element Ancestor: none Children: one or more TopicConfiguration
TopicConfiguration	Element for configuring the event notification topic. Type: element Ancestor: NotificationConfiguration Children: Id, Filter, Topic, Event, or Events
Topic	URN of the event notification topic. After detecting a specific event in the bucket, OBS sends a message to the topic. Type: string Ancestor: TopicConfiguration
Id	Unique ID of each event notification. If the user does not specify an ID, the system assigns an ID automatically. Type: string Ancestor: TopicConfiguration

Element	Description
Filter	Element used to store rules of filtering object names. Type: element Ancestor: TopicConfiguration Children: Object
Object	Element used to store rules of filtering object names. Type: element Ancestor: TopicConfiguration
FilterRule	Element that defines key-value pairs of the filtering rule. Type: element Ancestor: Object Children: Name, Value
Name	Prefix or suffix of object names for filtering Type: string Ancestor: FilterRule Value options: prefix, suffix
Value	Keywords of object names so that objects can be filtered based on the prefixes or suffixes Type: string Ancestor: FilterRule

Element	Description
Event	Type of events that need to be notified **NOTE** Multiple event types can be added in one TopicConfiguration item. Type: string Value options: The following values can be used to upload an object: • ObjectCreated:Put • ObjectCreated:Post • ObjectCreated:Copy • ObjectCreated:CompleteMultipartUpload Or use wildcard characters to support all upload operations: • ObjectCreated:* The following values can be used to delete an object: • ObjectRemoved:Delete • ObjectRemoved:DeleteMarkerCreated Or use wildcard characters to support all delete operations: • ObjectRemoved:* Ancestor: TopicConfiguration

Error Responses

No special error responses are involved. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?notification HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:16:32 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:r5+2zwPTKwupMg6lkeTUUqPcHfQ=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 900B000001643FDDD751B37BA87590D8
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSJRBSladTn5ZCVw6ZIY/DAs0zs6z7Hh
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:16:32 GMT
Content-Length: 490

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<NotificationConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
 <TopicConfiguration>
  <Topic>urn:smn:region:4b29a3cb5bd64581bda5714566814bb7:tet522</Topic>
  <Id>ConfigurationId</Id>
  <Filter>
   <Object>
    <FilterRule>
     <Name>prefix</Name>
     <Value>object</Value>
    </FilterRule>
    <FilterRule>
     <Name>suffix</Name>
     <Value>txt</Value>
    </FilterRule>
   </Object>
  </Filter>
  <Event>ObjectCreated:Put</Event>
 </TopicConfiguration>
</NotificationConfiguration>
```

5.2.15 Configuring Tags for a Bucket

Functions

This operation adds tags to a bucket.

After tags are added to a bucket, all data records generated by the requests for this bucket will take the same tags. Thus, reports can be categorized for detailed cost analysis. For example, if a running application uploads data to a bucket, you can tag the bucket with the application name. In this manner, the costs on the application can be analyzed using tags.

To perform this operation, you must have the **PutBucketTagging** permission. By default, only the bucket owner can delete the tags of a bucket. The bucket owner can allow other users to perform this operation by setting a bucket policy or granting them the permission.

NOTE

- A bucket can have a maximum of 10 tags.
- A tag key and key value can contain a maximum of 36 and 43 characters, respectively.
- Tag keys and key values cannot contain commas (,), asterisks (*), vertical bars (|), slashes (/), less-than signs (<), greater-than signs (>), equal signs (=), backslashes (\), or ASCII codes (0x00 to 0x1F).

Request Syntax

```
PUT /?tagging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
Content-MD5: md5
<Tagging>
   <TagSet>
      <Tag>
         <Key>Tag Name</Key>
         <Value>Tag Value</Value>
      </Tag>
   </TagSet>
</Tagging>
```

Request Parameters

This request contains no parameter.

Request Headers

Table 5-25 lists the request header.

Table 5-25 Request headers

Header	Description	Mandatory
Content-MD5	Base64-encoded 128-bit MD5 digest of the message according to RFC 1864. Type: string Example: n58IG6hfM7vqI4K0vnWpog==	Yes

Request Elements

In this request, you must configure bucket tags in the request body. The tag configuration is uploaded in XML format. Table 5-26 describes the configuration elements.

Table 5-26 Bucket tag configuration elements

Header	Description	Mandatory
Tagging	Element of the tag set and tag Type: element Ancestor: none	Yes
TagSet	Element of the tag set Type: element Ancestor: Tagging	Yes
Tag	Element of the tag information Type: element Ancestor: TagSet	Yes
Key	Tag name Type: string Ancestor: Tag	Yes
Value	Tag value Type: string Ancestor: Tag	Yes

Response Syntax

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Length: length
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

In addition common error codes, this API also returns other error codes. The following table lists common errors and possible causes. For details, see **Table 5-27**.

Table 5-27 Bucket tag configuration errors

Error Code	Description	HTTP Status Code
InvalidTagError	An invalid tag is provided when configuring bucket tags.	400 Bad Request
MalformedXMLError	An incorrect XML format is provided when configuring bucket tags.	400 Bad Request

Sample Request

```
PUT /?tagging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:22:50 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:Pf1ZyGvVYg2BzOjokZ/BAeR1mEQ=
Content-MD5: MnAEvkfQIGnBpchOE2U6Og==
Content-Length: 182

<Tagging xmlns="http://obs.example.com/doc/2015-06-30/">
  <TagSet>
    <Tag>
      <Key>TagName1</Key>
      <Value>TageSetVaule1</Value>
    </Tag>
  </TagSet>
</Tagging>
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF26000001643FEBA09B1ED46932CD07
```

```
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSEZp87iEirC6DggPB5cN49pSvHBWClg
Date: Wed, 27 Jun 2018 13:22:50 GMT
```

5.2.16 Obtaining Bucket Tags

Functions

This operation obtains information about tags of a bucket.

To perform this operation, you must have the **GetBucketTagging** permission. By default, only the bucket owner can obtain the tags of a bucket. The bucket owner can allow other users to perform this operation by setting a bucket policy or granting them the permission.

Request Syntax

```
GET /?tagging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
Content-Length: length
Date: date
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Tagging xmlns="http://obs.example.com/doc/2015-06-30/">
   <TagSet>
      <Tag>
         <Key>key</Key>
         <Value>value</Value>
      </Tag>
   </TagSet>
</Tagging>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to detail bucket tag configuration. **Table 5-28** describes the elements.

Table 5-28 Elements for configuring bucket tags

Element	Description
Tagging	Element of the tag set and tag. Type: element Ancestor: none
TagSet	Element of the tag set. Type: element Ancestor: Tagging
Tag	Element of the tag information. Type: element Ancestor: TagSet
Key	Tag name. Type: string Ancestor: Tag
Value	Tag value. Type: string Ancestor: Tag

Error Responses

In addition to common error codes, this API also returns other error codes. The following table lists common errors and possible causes. For details, see **Table 5-29**.

Table 5-29 Bucket tag configuration errors

Error Code	Description	HTTP Status Code
NoSuchTagSet	The specified bucket does not have any tags.	404 Not Found

Sample Request

```
GET /?tagging HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:25:44 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:H1INcyc5i0XlHqYTfuzkPxLZUPM=
```

Sample Response

```
HTTP/1.1 200 OK
x-obs-request-id: 0002B7532E0000015BEB35330C5884X1
```

```
x-obs-id-2: s12w20LYNQqSb7moq4ibgJwmQRSmVQV+rFBqplOGYkXUpXeS/nOmbkyD+E35K79j
Content-Type: application/xml
Date: Wed, 27 Jun 2018 13:25:44 GMT
Content-Length: 441

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Tagging xmlns="http://obs.example.com/doc/2015-06-30/">
  <TagSet>
    <Tag>
      <Key>TagName1</Key>
      <Value>TageSetVaule1</Value>
    </Tag>
  </TagSet>
</Tagging>
```

5.2.17 Deleting Tags

Functions

This operation deletes the tags of a bucket.

To perform this operation, you must have the **PutBucketTagging** permission. By default, only the bucket owner can delete the tags of a bucket. The bucket owner can allow other users to perform this operation by setting a bucket policy or granting them the permission.

Request Syntax

```
DELETE /?tagging HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization string
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Content-Length: length
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE /?tagging HTTP/1.1
User-Agent: curl/7.19.7
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Wed, 27 Jun 2018 13:46:58 GMT
Authorization: authorization string
```

Sample Response

```
HTTP/1.1 204 No Content
x-obs-request-id: 0002B7532E0000015BEB2C212E53A17L
x-obs-id-2: CqT+86nnOkB+Cv9KZoVgZ28pSgMF+uGQBUC68flvkQeq6CxoCz65wWFMNBpXvea4
Content-Length: 0
Date: Wed, 27 Jun 2018 13:46:58 GMT
```

5.2.18 Configuring Bucket Storage Quota

Functions

The bucket storage quota must be a positive integer in the unit of byte. The maximum storage quota is 2^{63} – 1 bytes. The default bucket storage quota is **0**, indicating that the bucket storage quota is not limited.

NOTE

1. For a bucket that has a specified storage quota, you can change the quota to **0** to cancel the quota limitation.
2. The bucket storage quota verification depends on how much space is used in the bucket. However, the used storage space is measured at the background. Therefore, bucket storage quotas may not take effect immediately, and delay is expected. It may occur that the used storage space in a bucket has exceeded the bucket storage quota, or the used storage space remains unchanged after data is deleted from the bucket.
3. For details about the API for querying used storage space, see **Querying Information About Used Space in a Bucket**.
4. If the used storage space in a bucket reaches the upper limit of the bucket storage quota, object upload will fail and the HTTP status code 403 Forbidden will be returned, indicating **InsufficientStorageSpace**. In this case, you can increase the quota, cancel the quota limitation (by changing the quota to **0**), or delete unwanted objects from the bucket.

Request Syntax

```
PUT /?quota HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Length: length
Authorization: authorization
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Quota xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <StorageQuota>value</StorageQuota>
</Quota>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request uses an additional element to specify a bucket quota. **Table 5-30** describes the element.

Table 5-30 Additional request elements

Element	Description	Mandatory
StorageQuota	Specifies the bucket storage quota. The unit is byte. Type: integer	Yes

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /?quota HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:24:37 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:k/rbwnYaqYf0Ae6F0M3OJQ0dmI8=
Content-Length: 106
```

```
<Quota xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <StorageQuota>10240000</StorageQuota>
</Quota>
```

Sample Response

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435E09A2BCA388688AA08
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSHbmBecv7ohDSvqaRObpxzgzJ9+l8xT
Date: WED, 01 Jul 2015 03:24:37 GMT
Content-Length: 0
```

5.2.19 Querying Bucket Storage Quota

Functions

Only the bucket owner can query information about the bucket storage quota. However, an inactive owner is not allowed to get the bucket quota. The bucket storage quota is measured by byte. **0** indicates that no upper limit is set.

Request Syntax

```
GET /?quota HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request contains no element.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: application/xml
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Quota xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <StorageQuota>quota</StorageQuota>
</Quota>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements of information about the bucket quota. **Table 5-31** describes the elements.

Table 5-31 Response elements

Element	Description
Quota	Bucket storage quota. This element contains the StorageQuota element. Type: XML
StorageQuota	Bucket storage quota quantity. The unit is byte. Type: string

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?quota HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:27:45 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:8m4bW1gFCNeXQlfu45uO2gpo7l8=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436B55D8DED9AE26C4D18
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSs2Q5vz5AfpAJ/CMNgCfo2hmDowp7M9
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:27:45 GMT
Content-Length: 150

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Quota xmlns="http://obs.example.com/doc/2015-06-30/">
  <StorageQuota>0</StorageQuota>
</Quota>
```

5.2.20 Querying Information About Used Space in a Bucket

Functions

This operation queries the number of bucket objects and the space occupied by the objects. The size of the object space is a positive integer, measured by bytes.

NOTE

The OBS bucket storage statistics is measured in the background, and the storage data is not updated in real time. Therefore, you are not advised to perform real-time verification on the storage information.

Request Syntax

```
GET /?storageinfo HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameters.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request contains no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<GetBucketStorageInfoResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Size>size</Size>
<ObjectNumber>number</ObjectNumber>
</GetBucketStorageInfoResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements of information about the used storage capacity of a bucket. **Table 5-32** describes the elements.

Table 5-32 Response elements

Element	Description
GetBucketStorageInfoResult	Request result that saves bucket storage information, including the stored data size and the number of objects Type: XML
Size	Size of stored data Type: integer
ObjectNumber	Number of objects returned Type: integer

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?storageinfo HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:31:18 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:bLcdeJGYWw/eEEjMhPZx2MK5R9U=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016435DD2958BFDCDB86B55E
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSitZctaPYVnat49fVMd1O+OWIP1yrg3
Content-Type: application/xml
WED, 01 Jul 2015 03:31:18 GMT
Content-Length: 206

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<GetBucketStorageInfoResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Size>25490</Size>
  <ObjectNumber>24</ObjectNumber>
</GetBucketStorageInfoResult>
```

5.2.21 Configuring Bucket Inventories

Functions

OBS uses the PUT method to configure bucket inventories. Each bucket can have a maximum of 10 inventories.

To perform this operation, ensure that you have the **PutBucketInventoryConfiguration** permission. By default, the bucket owner has this permission and can assign this permission to other users.

Request Syntax

```
PUT /?inventory&id=configuration-id  HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
Content-Length: length
Expect: 100-continue

<InventoryConfiguration>
   <Id>configuration-id</Id>
   <IsEnabled>true</IsEnabled>
   <Filter>
        <Prefix>inventoryTestPrefix</Prefix>
   </Filter>
   <Destination>
        <Format>CSV</Format>
        <Bucket>destbucket</Bucket>
        <Prefix>dest-prefix</Prefix>
   </Destination>
   <Schedule>
```

```
        <Frequency>Daily</Frequency>
    </Schedule>
    <IncludedObjectVersions>All</IncludedObjectVersions>
    <OptionalFields>
        <Field>Size</Field>
        <Field>LastModifiedDate</Field>
        <Field>ETag</Field>
        <Field>StorageClass</Field>
        <Field>IsMultipartUploaded</Field>
        <Field>ReplicationStatus</Field>
        <Field>EncryptionStatus</Field>
    </OptionalFields>
</InventoryConfiguration>
```

Request Parameters

Table 5-33 Request parameters

Parameter	Description	Mandatory
id	ID of the inventory configuration, which must be consistent with the inventory configuration ID in the message body. Type: string Specifications: A maximum of 64 characters There is no default value. Valid characters: letters, digits, hyphens (-), periods (.) and underscores (_)	Yes

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

In this request, you must configure the bucket inventory in the request body. Upload the inventory configuration information in an XML file. **Table 5-34** lists the configuration elements.

Table 5-34 Bucket inventory configuration elements

Element	Description	Mandatory
InventoryConfiguration	Inventory configuration. Type: container Ancestor: none Children: Id, IsEnabled, Filter, Destination, Schedule, IncludedObjectVersions, and OptionalFields	Yes

Element	Description	Mandatory
Id	ID of an inventory configuration, which must be consistent with the inventory configuration ID specified in the request. Type: string Specifications: A maximum of 64 characters There is no default value. Valid characters: letters, digits, hyphens (-), periods (.) and underscores (_) Ancestor: InventoryConfiguration	Yes
IsEnabled	Indicates whether the rule is enabled. If this parameter is set to **true**, the inventory is generated. If not, the inventory will not be generated. Type: boolean Valid values: **true** or **false** Ancestor: InventoryConfiguration	Yes
Filter	Inventory filter configuration. The inventory contains only objects that meet the filter criteria (filtering by object name prefix). If no filter criteria is configured, all objects are included. Type: container Ancestor: InventoryConfiguration Children: Prefix	No
Prefix	Filtering by name prefix. Only objects with the specified name prefix are included in the inventory. Type: string Ancestor: Filter	No
Schedule	Time scheduled for generation of inventories. Type: container Ancestor: InventoryConfiguration Children: Frequency	Yes

Element	Description	Mandatory
Frequency	Intervals when inventories are generated. You can set this parameter to **Daily** or **Weekly**. An inventory is generated within one hour after it is configured for the first time. Then it is generated at the specified intervals. Type: string Ancestor: Schedule Valid values: **Daily** or **Weekly**	Yes
Destination	Destination bucket of an inventory. Type: container Ancestor: InventoryConfiguration	Yes
Format	Inventory format. Only the CSV format is supported. Type: string Ancestor: Destination Valid values: **CSV**	Yes
Bucket	Name of the bucket for saving inventories. Type: string Ancestor: Destination	Yes
Prefix	The name prefix of inventory files. If no prefix is configured, the names of inventory files will start with the **BucketInventory** by default. Type: string Ancestor: Destination	No
IncludedObjectVersions	Indicates whether versions of objects are included in an inventory. ● If this parameter is set to **All**, all the versions of objects are included in the inventory, and versioning related fields are added to the inventory, including: **VersionId**, **IsLatest**, and **DeleteMarker**. ● If this parameter is set to **Current**, the inventory contains only the current objects versions at the time when the inventory is generated. No versioning fields are displayed in the inventory. Type: string Ancestor: InventoryConfiguration Valid values: **All** or **Current**	Yes

Element	Description	Mandatory
OptionalFields	Extra metadata fields that can be added to an inventory. If this parameter is configured, fields specified in this parameter are contained in the inventory. Type: container Ancestor: InventoryConfiguration Children: Field	No
Field	Optional fields. The **OptionalFields** can contain multiple field elements. Type: string Ancestor: OptionalFields Valid values: **Size**, **LastModifiedDate**, **StorageClass**, **ETag**, **IsMultipartUploaded**, **ReplicationStatus**, and **EncryptionStatus**.	No

Response Syntax

```
HTTP/1.1 status_code
x-obs-request-id: request id
x-obs-id-2: id
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

In addition common error codes, this API also returns other error codes. The following lists some common errors and possible causes of this API. For details, see **Table 5-35**.

Table 5-35 Inventory configuration error codes

Error Code	Description	HTTP Status Code
MalformedXML	Incorrect XML format of the inventory.	400 Bad Request
InvalidArgument	Invalid parameter.	400 Bad Request

Error Code	Description	HTTP Status Code
InventoryCountOverLimit	The number of inventories reached the upper limit.	400 Bad Request
PrefixExistInclusionRelationship	The prefix configured for this inventory overlaps with prefixes of existing inventories.	400 Bad Request

Sample Request

```
PUT /?inventory&id=test_id HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 08:17:10 +0000
Authorization: OBS UDSIAMSTUBTEST000001:/e2fqSfzLDb+0M36D4Op/s5KKr0=
Content-Length: 600
Expect: 100-continue

<InventoryConfiguration>
  <Id>test_id</Id>
  <IsEnabled>true</IsEnabled>
  <Filter>
       <Prefix>inventoryTestPrefix</Prefix>
  </Filter>
  <Destination>
       <Format>CSV</Format>
       <Bucket>destbucket</Bucket>
       <Prefix>dest-prefix</Prefix>
  </Destination>
  <Schedule>
        <Frequency>Daily</Frequency>
  </Schedule>
  <IncludedObjectVersions>All</IncludedObjectVersions>
  <OptionalFields>
        <Field>Size</Field>
        <Field>LastModifiedDate</Field>
        <Field>ETag</Field>
        <Field>StorageClass</Field>
        <Field>IsMultipartUploaded</Field>
        <Field>ReplicationStatus</Field>
        <Field>EncryptionStatus</Field>
  </OptionalFields>
</InventoryConfiguration>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001682C8545B0680893425D60AB83
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSIGTuRtBfo7lpHSt0ZknhdDHmllwd/p
Date: Tue, 08 Jan 2019 08:12:38 GMT
Content-Length: 0
```

5.2.22 Obtaining Bucket Inventories

Functions

OBS uses the GET method to obtain a specific inventory of a bucket.

To perform this operation, you must have the **GetBucketInventoryConfiguration** permission. By default, the bucket owner has this permission and can assign this permission to other users.

Request Syntax

```
GET /?inventory&id=configuration-id HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

Request Parameters

Table 5-36 Request parameters

Parameter	Description	Mandatory
id	ID of the inventory configuration that you want to obtain. Type: string Specifications: A maximum of 64 characters There is no default value. Valid characters: letters, digits, hyphens (-), periods (.) and underscores (_)	Yes

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Id>configuration-id</Id>
  <IsEnabled>true</IsEnabled>
  <Destination>
    <Format>CSV</Format>
    <Bucket>destbucket</Bucket>
    <Prefix>prefix</Prefix>
  </Destination>
  <Schedule>
    <Frequency>Daily</Frequency>
```

```
  </Schedule>
  <IncludedObjectVersions>Current</IncludedObjectVersions>
  <OptionalFields>
    <Field>Size</Field>
    <Field>LastModifiedDate</Field>
    <Field>ETag</Field>
    <Field>StorageClass</Field>
    <Field>IsMultipartUploaded</Field>
    <Field>ReplicationStatus</Field>
    <Field>EncryptionStatus</Field>
  </OptionalFields>
</InventoryConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

Table 5-37 lists elements contained in the response body.

Table 5-37 Elements in a response body to the request for bucket inventory configurations

Element	Description
InventoryConfiguration	Inventory configuration. Type: container Ancestor: none Children: Id, IsEnabled, Filter, Destination, Schedule, IncludedObjectVersions, and OptionalFields
Id	ID of an inventory configuration, which must be consistent with the inventory configuration ID specified in the request. Type: string Specifications: A maximum of 64 characters There is no default value. Valid characters: letters, digits, hyphens (-), periods (.) and underscores (_) Ancestor: InventoryConfiguration
IsEnabled	Indicates whether the rule is enabled. If this parameter is set to **true**, the inventory is generated. If not, the inventory will not be generated. Type: boolean Valid values: **true** or **false** Ancestor: InventoryConfiguration

Element	Description
Filter	Inventory filter configuration. The inventory contains only objects that meet the filter criteria (filtering by object name prefix). If no filter criteria is configured, all objects are included. Type: container Ancestor: InventoryConfiguration Children: Prefix
Prefix	Filtering by name prefix. Only objects with the specified name prefix are included in the inventory. Type: string Ancestor: Filter
Schedule	Time scheduled for generation of inventories. Type: container Ancestor: InventoryConfiguration Children: Frequency
Frequency	Intervals when inventories are generated. You can set this parameter to **Daily** or **Weekly**. An inventory is generated within one hour after it is configured for the first time. Then it is generated at the specified intervals. Type: string Ancestor: Schedule Valid values: **Daily** or **Weekly**
Destination	Destination bucket of an inventory. Type: container Ancestor: InventoryConfiguration
Format	Inventory format. Only the CSV format is supported. Type: string Ancestor: Destination Valid values: **CSV**
Bucket	Name of the bucket for saving inventories. Type: string Ancestor: Destination
Prefix	The name prefix of inventory files. If no prefix is configured, the names of inventory files will start with the **BucketInventory** by default. Type: string Ancestor: Destination

Element	Description
IncludedObjectVersions	Indicates whether versions of objects are included in an inventory. • If this parameter is set to **All**, all the versions of objects are included in the inventory, and versioning related fields are added to the inventory, including: **VersionId**, **IsLatest**, and **DeleteMarker**. • If this parameter is set to **Current**, the inventory contains only the current objects versions at the time when the inventory is generated. No versioning fields are displayed in the inventory. Type: string Ancestor: InventoryConfiguration Valid values: **All** or **Current**
OptionalFields	Extra metadata fields that can be added to an inventory. If this parameter is configured, fields specified in this parameter are contained in the inventory. Type: container Ancestor: InventoryConfiguration Children: Field
Field	Optional fields. The **OptionalFields** can contain multiple field elements. Type: string Ancestor: OptionalFields Valid values: **Size**, **LastModifiedDate**, **StorageClass**, **ETag**, **IsMultipartUploaded**, **ReplicationStatus**, and **EncryptionStatus**.

Error Responses

In addition common error codes, this API also returns other error codes. The following table lists common errors and possible causes. For details, see **Table 5-38**.

Table 5-38 Error codes related to obtaining inventory configurations

Error Code	Description	HTTP Status Code
NoSuchInventoryConfiguration	No inventory configuration found matching the specified ID.	404 Not Found

Sample Request

```
GET /?inventory&id=id1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 09:32:24 +0000
Authorization: OBS UDSIAMSTUBTEST000001:ySWncC9M08jNsyXdJLSMJkpi7XM=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001682CB4C2EE6808A0D8DF9F3D00
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSBjn5O7Jv9CqvUMO0BenehRdil1n8rR
Content-Type: application/xml
Date: Tue, 08 Jan 2019 09:04:30 GMT
Content-Length: 626

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Id>id1</Id>
  <IsEnabled>true</IsEnabled>
  <Destination>
    <Format>CSV</Format>
    <Bucket>bucket</Bucket>
    <Prefix>prefix</Prefix>
  </Destination>
  <Schedule>
    <Frequency>Daily</Frequency>
  </Schedule>
  <IncludedObjectVersions>Current</IncludedObjectVersions>
  <OptionalFields>
    <Field>Size</Field>
    <Field>LastModifiedDate</Field>
    <Field>ETag</Field>
    <Field>StorageClass</Field>
    <Field>IsMultipartUploaded</Field>
    <Field>ReplicationStatus</Field>
    <Field>EncryptionStatus</Field>
  </OptionalFields>
</InventoryConfiguration>
```

5.2.23 Listing Bucket Inventories

Functions

OBS uses the GET method without inventory IDs to obtain all inventories of a specified bucket. Obtained inventories are returned together on only one page.

To perform this operation, you must have the **GetBucketInventoryConfiguration** permission. By default, only the bucket owner can delete the tags of a bucket. The bucket owner can allow other users to perform this operation by setting a bucket policy or granting them the permission.

Request Syntax

```
GET /?inventory HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

Request Parameters

This request message does not contain the request parameters.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListInventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
 <InventoryConfiguration>
  <Id>id</Id>
   <IsEnabled>true</IsEnabled>
   <Destination>
    <Format>CSV</Format>
    <Bucket>bucket</Bucket>
    <Prefix>prefix</Prefix>
   </Destination>
   <Schedule>
    <Frequency>Daily</Frequency>
   </Schedule>
   <IncludedObjectVersions>Current</IncludedObjectVersions>
   <OptionalFields>
    <Field>Size</Field>
    <Field>LastModifiedDate</Field>
    <Field>ETag</Field>
    <Field>StorageClass</Field>
    <Field>IsMultipartUploaded</Field>
    <Field>ReplicationStatus</Field>
    <Field>EncryptionStatus</Field>
   </OptionalFields>
 </InventoryConfiguration>
</ListInventoryConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

Table 5-39 lists elements contained in the response body.

Table 5-39 Bucket inventory configuration elements

Element	Description
ListInventoryConfiguration	List of bucket inventories. Type: Container
InventoryConfiguration	Bucket inventory configuration. For details about the configuration elements, see **Table 5-37**. Type: Container Ancestor: ListInventoryConfiguration

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /?inventory HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 09:32:24 +0000
Authorization: OBS UDSIAMSTUBTEST000001:ySWncC9M08jNsyXdJLSMJkpi7XM=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 000001682CB4C2EE6808A0D8DF9F3D00
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSBjn5O7Jv9CqvUMO0BenehRdil1n8rR
Content-Type: application/xml
Date: Tue, 08 Jan 2019 09:04:30 GMT
Content-Length: 626

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListInventoryConfiguration  xmlns="http://obs.region.example.com/doc/2015-06-30/">
 <InventoryConfiguration>
  <Id>id1</Id>
   <IsEnabled>true</IsEnabled>
   <Destination>
     <Format>CSV</Format>
     <Bucket>bucket</Bucket>
     <Prefix>prefix</Prefix>
   </Destination>
   <Schedule>
     <Frequency>Daily</Frequency>
   </Schedule>
   <IncludedObjectVersions>Current</IncludedObjectVersions>
   <OptionalFields>
     <Field>Size</Field>
     <Field>LastModifiedDate</Field>
     <Field>ETag</Field>
     <Field>StorageClass</Field>
     <Field>IsMultipartUploaded</Field>
     <Field>ReplicationStatus</Field>
     <Field>EncryptionStatus</Field>
   </OptionalFields>
 </InventoryConfiguration>
</ListInventoryConfiguration>
```

5.2.24 Deleting Bucket Inventories

Functions

OBS uses the DELETE method to delete inventories (identified by inventory IDs) of a specified bucket.

To perform this operation, you must have the **DeleteBucketInventoryConfiguration** permission. By default, only the bucket owner can delete the tags of a bucket. The bucket owner can allow other users to perform this operation by setting a bucket policy or granting them the permission.

Request Syntax

```
DELETE /?inventory&id=configuration-id HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: date
Authorization: authorization string
```

Request Parameters

Table 5-40 Request parameters

Parameter	Description	Mandatory
id	ID of the inventory to be deleted. Type: string Specifications: A maximum of 64 characters There is no default value. Valid characters: letters, digits, hyphens (-), periods (.) and underscores (_)	Yes

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Server: OBS
x-obs-request-id: request id
x-obs-id-2: id
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE /test?inventory&id=id1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Tue, 08 Jan 2019 13:18:35 +0000
Authorization: OBS UDSIAMSTUBTEST000001:UT9F2YUgaFu9uFGMmxFj2CBgQHs=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 000001682D993B666808E265A3F6361D
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSyB46jGSQsu06m1nyIeKxTuJ+H27ooC
Date: Tue, 08 Jan 2019 13:14:03 GMT
```

5.3 Static Website Hosting

5.3.1 Configuring Static Website Hosting for a Bucket

Functions

OBS allows you to store static web page resources such as HTML web pages, flash files, videos, and audios in a bucket. When a client accesses these resources from the website endpoint of the bucket, the browser can directly resolve and present the resources to the client. This operation is applicable to:

- Redirecting all requests to a website endpoint.
- Adding routing rules that redirect specific requests.

You can perform this operation to create or update the website configuration of a bucket.

To perform this operation, you must have the **PutBucketWebsite** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

NOTE

Avoid using periods (.) in the destination bucket name. Otherwise, failures in client authentication certificate may occur when users use HTTPS for access.

The maximum size of a network configuration request for a bucket is 10 KB.

Request Syntax

```
PUT /?website HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
Authorization: authorization
<WebsiteConfiguration>
   <RedirectAllRequestsTo>
      <HostName>hostName</HostName>
   </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request contains elements to specify the website configuration in XML format.

- To redirect all website requests sent to the bucket's website endpoint, add the elements as described in **Table 5-41**.

  Table 5-41 Elements for redirecting all website requests

Element	Description	Mandatory
WebsiteConfiguration	Root node configured on the website Type: element Ancestor: none	Yes
RedirectAllRequestsTo	Describes the redirection behavior for every request to this bucket's website endpoint. If this element is present, no other siblings are allowed. Type: element Ancestor: WebsiteConfiguration	Yes
HostName	Name of the host where requests will be redirected Type: string Ancestor: RedirectAllRequestsTo	Yes

Element	Description	Mandatory
Protocol	The HTTP or HTTPS protocol used in redirecting requests. The default protocol is HTTP. Type: string Ancestor: RedirectAllRequestsTo	No

- To configure redirection rules, add the elements as described in **Table 5-42**.

  Table 5-42 Elements for adding rules that redirect requests

Element	Description	Mandatory
WebsiteConfiguration	Root element for the website configuration Type: element Ancestor: none	Yes
IndexDocument	***Suff*** element Type: element Ancestor: WebsiteConfiguration	Yes
Suffix	*Suffix* that is appended to a request initiated for a directory on the website endpoint. For example, if the *suffix* is **index.html** and you request for **samplebucket/images/**, the data that is returned will be for the object with the key name **images/index.html** in the **samplebucket** bucket. *Suffix* cannot be empty or contain slashes (/). Type: string Ancestor: IndexDocument	Yes
ErrorDocument	*Key* element Type: element Ancestor: WebsiteConfiguration	No

Element	Description	Mandatory
Key	Object key that is used when a 4XX error occurs. This element identifies the page that is returned when a 4XX error occurs. Type: string Ancestor: ErrorDocument Condition: Required when **ErrorDocument** is specified.	No
RoutingRules	***Routing*** element Type: element Ancestor: WebsiteConfiguration	No
RoutingRule	Element of a redirection rule. A redirection rule contains a **Condition** and a **Redirect**. When the **Condition** is matched, **Redirect** takes effect. Type: element Ancestor: RoutingRules At least the *RoutingRule* element is required.	Yes
Condition	Element for describing a condition that must be met for the specified redirection to apply. Type: element Ancestor: RoutingRule	No
KeyPrefixEquals	Object key name prefix when the redirection is applied. Example: • To redirect the request for object **ExamplePage.html**, the **KeyPrefixEquals** is set to **ExamplePage.html**. Type: string Ancestor: Condition Condition: Required when the ancestor element **Condition** is specified and sibling **HttpErrorCodeReturnedEquals** is not specified. If two conditions are specified, both conditions must be true for the **Redirect** to be applied.	No

Element	Description	Mandatory
HttpErrorCodeReturnedEquals	HTTP error code returned after the **Redirect** has taken effect. The specified **Redirect** is applied only when the error code returned equals this value. Example: • If you want to redirect requests to **NotFound.html** when HTTP error code 404 is returned, set HttpErrorCodeReturnedEquals to 404 in **Condition**, and set **ReplaceKeyWith** to **NotFound.html** in **Redirect**. Type: string Ancestor: Condition Condition: Required when ancestor element **Condition** is specified and sibling **KeyPrefixEquals** is not specified. If multiple conditions are specified, the **Redirect** takes effect only after all conditions are met.	No
Redirect	Element for redirection information. You can redirect requests to another host, to another web page, or with another protocol. You can specify an error code to be returned after an error. Type: element Ancestor: RoutingRule	Yes
Protocol	Protocol used in the redirection request Type: string Ancestor: Redirect Value options: http, https Condition: Not required if one of the siblings is present.	No
HostName	Host name used in the redirection request. Type: string Ancestor: Redirect Condition: Not required if one of the siblings is present.	No

Element	Description	Mandatory
ReplaceKeyPrefixWith	Object key prefix used in the redirection request. Example: • To redirect all requests for (objects under) docs to (objects under) documents, set KeyPrefixEquals to docs in **Condition** and ReplaceKeyPrefixWith to documents in **Redirect**. Type: string Ancestor: Redirect Condition: Not required if one of the siblings is present. Can be present only if ReplaceKeyWith is not provided.	No
ReplaceKeyWith	Object key used in the redirection request. For example, redirect requests to **error.html**. Type: string Ancestor: Redirect Condition: Not required if one of the siblings is present. Can be present only if **ReplaceKeyPrefixWith** is not provided.	No
HttpRedirectCode	HTTP status code returned after the redirection request Type: string Ancestor: Redirect Condition: Not required if one of the siblings is present.	No

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains no element.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /?website HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:40:29 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:pUK7Yp0yebnq4P6gqzVjoS7whoM=
Content-Length: 194

<WebsiteConfiguration xmlns="http://obs.example.com/doc/2015-06-30/">
  <RedirectAllRequestsTo>
    <HostName>www.example.com</HostName>
  </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164360D144670B9D02AABC6
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSItqMZ/AoFUX97l1xx8s67V3cCQtXWk
Date: WED, 01 Jul 2015 03:40:29 GMT
Content-Length: 0
```

5.3.2 Obtaining the Static Website Hosting Configuration of a Bucket

Functions

You can perform this operation to get the static website hosting configuration of a bucket.

To perform this operation, you must have the **GetBucketWebsite** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

Request Syntax

```
GET /?website HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WebsiteConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <RedirectAllRequestsTo>
      <HostName>hostName</HostName>
   </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements the same as those used by the PutBucketWebsite request. For details, see **Request Elements**.

Error Responses

Table 5-43 describes possible special errors in this request.

Table 5-43 Special error

Error Code	Description	HTTP Status Code
NoSuchWebsiteConfiguration	The website configuration does not exist.	404 Not Found

For other errors, see **Table 7-3**.

Sample Request

```
GET /?website HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:41:54 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:Yxt1Ru+feHE0S94R7dcBp+hfLnI=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164363442EC03A8CA3DD7F5
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSFbGOmlN0BVp1kbwN3HWr8jbVvtKEKN
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:41:54 GMT
Content-Length: 250
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<WebsiteConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <RedirectAllRequestsTo>
    <HostName>www.example.com</HostName>
  </RedirectAllRequestsTo>
</WebsiteConfiguration>
```

5.3.3 Deleting the Static Website Hosting Configuration of a Bucket

Functions

You can perform this operation to delete the website configuration of a bucket.

To perform this operation, you must have the **DeleteBucketWebsite** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

Request Syntax

```
DELETE /?website HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: type
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE /?website HTTP/1.1
User-Agent: curl/7.29.0
Host: bucketname.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:44:37 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:AZ1b0N5eLknxNOe/c0BISV1bEqc=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF2600000164363786230E2001DC0807
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSFUG4fEyDRgzUiEY2i71bJndBCy+wUZ
Date: WED, 01 Jul 2015 03:44:37 GMT
```

5.3.4 Configuring Bucket CORS

Functions

Cross-origin resource sharing (CORS) is a standard mechanism proposed by World Wide Web Consortium (W3C) and allows cross-origin requests from clients. For standard web page requests, the scripts and contents at one website cannot interact with those at another website due to the existence of the Same Origin Policy (SOP).

OBS allows buckets to store static web resources. The buckets of OBS can serve as website resources if the buckets are properly used (for details, see **Configuring Static Website Hosting for a Bucket**). A website in OBS can respond to requests of another websites only after CORS is properly configured.

Typical application scenarios are as follows:

- With the support of CORS, you can use JavaScript and HTML5 to construct web applications and directly access the resources in OBS without the need to use proxy servers for transfer.
- You can enable the dragging function of HTML 5 to directly upload files to the OBS (with the upload progress displayed) or update the OBS contents using web applications.
- Hosts external web pages, style sheets, and HTML 5 applications in different origins. Web fonts or pictures on OBS can be shared by multiple websites.

To perform this operation, you must have the **PutBucketCORS** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

Request Syntax

```
PUT /?cors HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Length: length
Date: date
```

```
Authorization: authorization
Content-MD5: MD5
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration>
   <CORSRule>
      <ID>id</ID>
      <AllowedMethod>method</AllowedMethod>
      <AllowedOrigin>origin</AllowedOrigin>
      <AllowedHeader>header</AllowedHeader>
      <MaxAgeSeconds>seconds</MaxAgeSeconds>
      <ExposeHeader>header</ExposeHeader>
   </CORSRule>
</CORSConfiguration>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers and CORS request headers. For details, see **Table 3-3** and **Table 5-44**.

Table 5-44 CORS request header

Header	Description	Mandatory
Content-MD5	Base64-encoded 128-bit MD5 digest of the message according to RFC 1864 Type: string Example: n58IG6hfM7vqI4K0vnWpog==	Yes

Request Elements

In this request, you must configure the CORS of buckets in the request body. The lifecycle configuration is specified as XML with elements described in **Table 5-45**.

Table 5-45 CORS configuration elements

Element	Description	Mandatory
CORSConfiguration	Root node of **CORSRule** and its capacity cannot exceed 64 KB. Type: Container Ancestor: none	Yes
CORSRule	CORS rules. **CORSConfiguration** can contain a maximum of 100 rules. Type: Container Ancestor: CORSConfiguration	Yes

Element	Description	Mandatory
ID	Unique identifier of a rule. The value can contain a maximum of 255 characters. Type: string Ancestor: CORSRule	No
AllowedMethod	Method allowed by a CORS rule Type: string Possible values are GET, PUT, HEAD, POST, and DELETE. Ancestor: CORSRule	Yes
AllowedOrigin	An origin that is allowed by a CORS rule. It is a character string and can contain a wildcard (*), and allows one wildcard character (*) at most. Type: string Ancestor: CORSRule	Yes
AllowedHeader	Headers that are allowed in a PutBucketCORS request via the **Access-Control-Request-Headers** header. If a request contains **Access-Control-Request- Headers**, only a CORS request that matches the configuration of AllowedHeader is considered as a valid request. Each AllowedHeader can contain at most one wildcard (*) and cannot contain spaces. Type: string Ancestor: CORSRule	No
MaxAgeSeconds	Indicates the response time of CORS that can be cached by a client. It is expressed in seconds. Each CORSRule can contain only one MaxAgeSeconds. It can be set to a negative value. Type: Integer Ancestor: CORSRule	No
ExposeHeader	An additional header in CORS responses. The header provides additional information for clients. It cannot contain spaces. Type: string Ancestor: CORSRule	No

Response Syntax

```
HTTP/1.1 status_code

Date: date
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains no element.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /?cors HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:51:52 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:lq7BGoqE9yyhdEwE6KojJ7ysVxU=
Content-MD5: NGLzvw81f/A2C9PiGO0aZQ==
Content-Length: 617

<?xml version="1.0" encoding="utf-8"?>
<CORSConfiguration>
  <CORSRule>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedOrigin>www.example.com</AllowedOrigin>
    <AllowedHeader>AllowedHeader_1</AllowedHeader>
    <AllowedHeader>AllowedHeader_2</AllowedHeader>
    <MaxAgeSeconds>100</MaxAgeSeconds>
    <ExposeHeader>ExposeHeader_1</ExposeHeader>
    <ExposeHeader>ExposeHeader_2</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
```

Sample Response

```
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643627112BD03512FC94A4
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSYi6wLC4bkrvuS9sqnlRjxK2a5Fe3ry
Date: WED, 01 Jul 2015 03:51:52 GMT
Content-Length: 0
```

5.3.5 Obtaining the CORS Configuration of a Bucket

Functions

You can perform this operation to obtain CORS configuration information about a specified bucket.

To perform this operation, you must have the **GetBucketCORS** permission. By default, only the bucket owner can perform this operation. The bucket owner can grant the permission to other users by configuring the bucket policy or user policy.

Request Syntax

```
GET /?cors HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type:  application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CORSConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <CORSRule>
      ...
   </CORSRule>
</CORSConfiguration>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to detail the configuration. **Table 5-46** describes the elements.

Table 5-46 CORS configuration elements

Element	Description
CORSConfiguration	Root node of **CORSRules** and its capacity cannot exceed 64 KB. Type: Container Ancestor: none
CORSRule	CORS rule. CORSConfiguration can contain a maximum of 100 rules. Type: Container Ancestor: CORSConfiguration

Element	Description
ID	Unique identifier of a rule. The value can contain a maximum of 255 characters. Type: string Ancestor: CORSRule
AllowedMethod	Method allowed by a CORS rule. Type: string Possible values are GET, PUT, HEAD, POST, and DELETE. Ancestor: CORSRule
AllowedOrigin	Indicates an origin that is allowed by a CORS rule. It is a character string and can contain a wildcard (*), and allows one wildcard character (*) at most. Type: string Ancestor: CORSRule
AllowedHeader	Indicates which headers are allowed in a PUT Bucket CORS request via the **Access-Control-Request-Headers** header. If a request contains **Access-Control-Request-Headers**, only a CORS request that matches the configuration of AllowedHeader is considered as a valid request. Each AllowedHeader can contain at most one wildcard (*) and cannot contain spaces. Type: string Ancestor: CORSRule
MaxAgeSeconds	Response time of CORS that can be cached by a client. It is expressed in seconds. Each CORSRule can contain only one MaxAgeSeconds. It can be set to a negative value. Type: Integer Ancestor: CORSRule
ExposeHeader	Indicates a supplemented header in CORS responses. The header provides additional information for clients. It cannot contain spaces. Type: string Ancestor: CORSRule

Error Responses

Table 5-47 describes possible special errors in the request.

Table 5-47 Special error

Error Code	Description	HTTP Status Code
NoSuchCORSConfiguration	Indicates that the CORS configuration of buckets does not exist.	404 Not Found

For other errors, see **Table 7-3**.

Sample Request

```
GET /?cors HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:54:36 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:WJGghTrPQQXRuCx5go1fHyE+Wwg=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164363593F10738B80CACBE
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSpngvwC5TskcLGh7Fz5KRmCFIayuY8p
Content-Type: application/xml
Date: WED, 01 Jul 2015 03:54:36 GMT
Content-Length: 825

<?xml version="1.0" encoding="utf-8"?>
<CORSConfiguration xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <CORSRule>
    <ID>783fc6652cf246c096ea836694f71855</ID>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>

    <AllowedOrigin>obs.example.com</AllowedOrigin>
    <AllowedOrigin>www.example.com</AllowedOrigin>
    <AllowedHeader>AllowedHeader_1</AllowedHeader>
    <AllowedHeader>AllowedHeader_2</AllowedHeader>
    <MaxAgeSeconds>100</MaxAgeSeconds>
    <ExposeHeader>ExposeHeader_1</ExposeHeader>
    <ExposeHeader>ExposeHeader_2</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
```

5.3.6 Deleting the CORS Configuration of a Bucket

Functions

This operation is used to delete the CORS configuration of a bucket. After the CORS configuration is deleted, the bucket and objects in it cannot be accessed by requests from other websites.

To perform this operation, you must have the **PutBucketCORS** permission.

Request Syntax

```
DELETE /?cors HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type:  application/xml
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE /?cors HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 03:56:41 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mKUs/uIPb8BP0ZhvMd4wEy+EbiI=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: BF26000001643639F290185BB27F793A
x-obs-id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCSLWMRFJfckapW+ktT/+1AnAz7XlNU0b
Date: WED, 01 Jul 2015 03:56:41 GMT
```

5.3.7 OPTIONS Bucket

Functions

OPTIONS refers to pre-requests that are sent to servers by clients. Generally, the requests are used to check whether clients have permissions to perform operations on servers. Only after a pre-request is returned successfully, clients start to execute the follow-up requests.

The OBS allows buckets to store static web resources. OBS buckets can serve as website resources if the buckets are properly used. In this scenario, buckets in the OBS serve as servers to process OPTIONS pre-requests from clients.

OBS can process OPTIONS pre-requests only after CORS is configured for buckets in OBS. For details about CORS, see **Configuring Bucket CORS**.

Differences Between OPTIONS Bucket and OPTIONS Object

With the OPTIONS Object, you need to specify an object name in the URL, but an object name is not required with the OPTIONS Bucket, which uses the bucket domain name as the URL. The request lines of the two methods are as follows:

```
OPTIONS /object HTTP/1.1
OPTIONS / HTTP/1.1
```

Request Syntax

```
OPTIONS / HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Origin: origin
Access-Control-Request-Method: method
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses the headers described in **Table 5-48**.

Table 5-48 OPTIONS request headers

Header	Description	Mandatory
Origin	Origin of the cross-domain request specified by the pre-request. Generally, it is a domain name set in CORS. Type: string	Yes

Header	Description	Mandatory
Access-Control-Request-Method	An HTTP method that can be used by a request. The request can use multiple method headers. Type: string Possible values are GET, PUT, HEAD, POST, and DELETE.	Yes
Access-Control-Request-Headers	HTTP headers of a request. The request can use multiple HTTP headers. Type: string	No

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: application/xml
Access-Control-Allow-Origin: origin
Access-Control-Allow-Methods: method
Access-Control-Allow-Header: header
Access-Control-Max-Age: time
Access-Control-Expose-Headers: header
Date: date
Content-Length: length
```

Response Headers

The response uses the following headers as described in **Table 5-49**.

Table 5-49 CORS response headers

Header	Description
Access-Control-Allow-Origin	If the origin of a request meets server CORS configuration requirements, the response contains the origin. Type: string
Access-Control-Allow-Headers	If the headers of a request meet server CORS configuration requirements, the response contains the headers. Type: string
Access-Control-Max-Age	Value of MaxAgeSeconds in the CORS configuration of a server Type: integer

Header	Description
Access-Control-Allow-Methods	If the Access-Control-Request-Method of a request meets server CORS configuration requirements, the response contains the methods in the rule. Type: string Possible values are GET, PUT, HEAD, POST, and DELETE.
Access-Control-Expose-Headers	Value of ExposeHeader in the CORS configuration of a server Type: string

Response Elements

This response involves no elements.

Error Responses

Table 5-50 describes possible special errors in the request.

Table 5-50 Special error

Error Code	Description	HTTP Status Code
Bad Request	Invalid Access-Control-Request-Method: null When CORS and OPTIONS are configured for a bucket, no method header is added.	400 BadRequest
Bad Request	Insufficient information. Origin request header needed. When CORS and OPTIONS are configured for a bucket, no origin header is added.	400 BadRequest
AccessForbidden	CORSResponse: This CORS request is not allowed. This is usually because the evaluation of Origin, request method / Access-Control-Request-Method or Access-Control-Request-Headers are not whitelisted by the resource's CORS specification. When CORS and OPTIONS are configured for a bucket, origin, method, and headers do not match any rule.	403 Forbidden

For other errors, see **Table 7-3**.

Sample Request

```
OPTIONS / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:02:15 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:7RqP1vjemo6U+Adv9/Y6eGzWrzA=
Origin: www.example.com
Access-Control-Request-Method: PUT
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF260000016436314E8FF936946DBC9C
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT,DELETE
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: ExposeHeader_1,ExposeHeader_2
Access-Control-Allow-Credentials: true
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTlYimJvOyJncCLNm5y/iz6MAGLNxTuS
Date: WED, 01 Jul 2015 04:02:15 GMT
Content-Length: 0
```

5.3.8 OPTIONS Object

Functions

For details, see **OPTIONS Bucket**.

Differences Between OPTIONS Bucket and OPTIONS Object

With the OPTIONS Object, you need to specify an object name in the URL, but an object name is not required with the OPTIONS Bucket, which uses the bucket domain name as the URL. The request lines of the two methods are as follows:

```
OPTIONS /object HTTP/1.1
OPTIONS / HTTP/1.1
```

Request Syntax

```
OPTIONS /object HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Origin: origin
Access-Control-Request-Method: method
```

Request Parameters

This request contains no parameter.

Request Headers

Table 5-51 describes headers used by this request.

Table 5-51 OPTIONS request headers

Header	Description	Mandatory
Origin	Origin of the cross-domain request specified by the pre-request. Generally, it is a domain name set in CORS. Type: string	Yes
Access-Control-Request-Method	Indicates an HTTP method that can be used by a request. The request can use multiple method headers. Type: string Valid values: **GET**, **PUT**, **HEAD**, **POST**, and **DELETE**.	Yes
Access-Control-Request-Headers	Indicates the HTTP headers of a request. The request can use multiple HTTP headers. Type: string	No

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: type
Access-Control-Allow-Origin: origin
Access-Control-Allow-Methods: method
Access-Control-Allow-Header: header
Access-Control-Max-Age: time
Access-Control-Expose-Headers: header
Date: date
Content-Length: length
```

Response Headers

The request uses the headers described in **Table 5-52**.

Table 5-52 CORS request headers

Header	Description
Access-Control-Allow-Origin	If the origin of a request meets server CORS configuration requirements, the response contains the origin. Type: string

Header	Description
Access-Control-Allow-Headers	If the headers of a request meet server CORS configuration requirements, the response contains the headers. Type: string
Access-Control-Max-Age	Value of MaxAgeSeconds in the CORS configuration of a server. Type: integer
Access-Control-Allow-Methods	If the Access-Control-Request-Method of a request meets server CORS configuration requirements, the response contains the methods in the rule. Type: string Possible values are GET, PUT, HEAD, POST, and DELETE.
Access-Control-Expose-Headers	Indicates ExposeHeader in the CORS configuration of a server. Type: string

Response Elements

This response involves no elements.

Error Responses

Table 5-53 describes possible special errors in the request.

Table 5-53 Special error

Error Code	Description	HTTP Status Code
Bad Request	Invalid Access-Control-Request-Method: null When CORS and OPTIONS are configured for a bucket, no method header is added.	400 BadRequest
Bad Request	Insufficient information. Origin request header needed. When CORS and OPTIONS are configured for a bucket, no origin header is added.	400 BadRequest

Error Code	Description	HTTP Status Code
AccessForbidden	CORSResponse: This CORS request is not allowed. This is usually because the evaluation of Origin, request method/Access-Control-Request-Method or Access-Control-Request-Headers are not whitelisted by the resource's CORS spec. When CORS and OPTIONS are configured for a bucket, origin, method, and headers do not match any rule.	403 Forbidden

For other errors, see **Table 7-3**.

Sample Request

```
OPTIONS /object_1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:02:19 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:bQZG9c2aokAJsHOOkuVBK6cHZZQ=
Origin: www.example.com
Access-Control-Request-Method: PUT
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF26000001643632D12EFCE1C1294555
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT,DELETE
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: ExposeHeader_1,ExposeHeader_2
Access-Control-Allow-Credentials: true
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCS+DXV4zZetbTqFehhEcuXywTa/mi3T3
Date: WED, 01 Jul 2015 04:02:19 GMT
Content-Length: 0
```

5.4 Operations on Objects

5.4.1 Uploading Objects - PUT

Functions

After bucket creation in OBS, you can use this operation to upload an object to the bucket. Uploading an object adds it to a bucket. This requires users to have the write operation.

NOTE

The name of each object in a bucket must be unique.

With versioning not enabled, if an object to be uploaded has the same name as an existing object in the bucket, the newly uploaded object will overwrite the existing one. To protect data from being corrupted during transmission, you can add the **Content-MD5** parameter in the request header. After receiving the request, OBS will perform an MD5 consistency check. If the two MD5 values are inconsistent, the system returns an error message.

You can also specify the value of the **x-obs-acl** parameter to configure an access control policy for the object. If the **x-obs-acl** parameter is not specified when an anonymous user uploads an object, the object can be accessed by all OBS users by default.

For a single upload, the size of the object to be uploaded ranges [0, 5 GB]. To upload a file greater than 5 GB, see **Operations on Multipart Upload**.

OBS does not have real folders. To facilitate data management, OBS provides a method to simulate a folder by adding a slash (/) to the object name, for example, **test/123.jpg**. You can simulate **test** as a folder and **123.jpg** as the name of a file under the **test** folder. However, the object key remains **test/123.jpg**. Objects named in this format appear as folders on the console.

Differences Between PUT and POST Methods

Parameters are passed through the request header if the PUT method is used to upload objects; if the POST method is used to upload objects, parameters are passed through the form field in the message body.

With the PUT method, you need to specify the object name in the URL, but object name is not required with the POST method, which uses the bucket domain name as the URL. The request lines of the two methods are as follows:

```
PUT /ObjectName HTTP/1.1
POST / HTTP/1.1
```

For details about POST upload, see **Uploading Objects - POST**.

Versioning

If versioning is enabled for a bucket, the system automatically generates a unique version ID for the requested object in this bucket and returns the version ID in response header **x-obs-version-id**. If versioning is suspended for the bucket, the object version is **null**. For details about the versioning statuses of a bucket, see **Configuring Versioning for a Bucket**.

Request Syntax

```
PUT /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Content-Type: application/xml
Content-Length: length
Authorization: authorization
Date: date
<Optional Additional Header>
<object Content>
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**. The request can use additional headers, as listed in **Table 5-54**.

NOTE

OBS supports the six HTTP request headers: Cache-Control, Expires, Content-Encoding, Content-Disposition, Content-Type, and Content-Language. If these headers are carried in an object upload request, their values are saved. You can also call the metadata modification API, provided by OBS, to change the values of the six headers. When the object is downloaded or queried, the saved values are set for corresponding HTTP headers and returned to the client.

Table 5-54 Request headers

Header	Description	Mandatory
Content-MD5	Base64-encoded 128-bit MD5 digest of the message according to RFC 1864. Type: string Example: n58IG6hfM7vqI4K0vnWpog==	No
x-obs-acl	This header can be added to set access control policies for objects when creating the objects. The access control policies are the predefined common policies, including **private**, **public-read**, **public-read-write**. Type: string Note: This header is a predefined policy expressed in a character string. Example: **x-obs-acl: public-read**	No
x-obs-grant-read	When creating an object, you can use this header to authorize all users in an account the permission to read objects and obtain object metadata. Type: string Example: **x-obs-grant-read: id=domainID** If multiple accounts are authorized, separate them with commas (,).	No

Header	Description	Mandatory
x-obs-grant-read-acp	When creating an object, you can use this header to authorize all users in an account the permission to obtain the object ACL. Type: string Example: **x-obs-grant-read-acp: id=domainID** If multiple accounts are authorized, separate them with commas (,).	No
x-obs-grant-write-acp	When creating an object, you can use this header to authorize all users in an account the permission to write the object ACL. Type: string Example: **x-obs-grant-write-acp: id=domainID** If multiple accounts are authorized, separate them with commas (,).	No
x-obs-grant-full-control	When creating an object, you can use this header to authorize all users in an account the permission to read the object, obtain the object metadata, obtain the object ACL, and write the object ACL. Type: string Example: **x-obs-grant-full-control: id=domainID** If multiple accounts are authorized, separate them with commas (,).	No
x-obs-meta-*	When creating an object, you can use a header starting with **x-obs-meta-** to define object metadata in an HTTP request. User-defined metadata will be returned in the response header when you retrieve or query the metadata of the object. Type: string Example: **x-obs-meta-test: test metadata**	No

Header	Description	Mandatory
x-obs-website-redirect-location	If a bucket is configured with the static website hosting function, it will redirect requests for the object to another object in the same bucket or to an external URL. OBS stores the value of this header in the object metadata. In the following example, the request header sets the redirection to an object (**anotherPage.html**) in the same bucket: x-obs-website-redirect-location:/anotherPage.html In the following example, the request header sets the object redirection to an external URL: x-obs-website-redirect-location:http://www.example.com/ Type: string There is no default value. Constraint: The value must be prefixed by a slash (/), **http://**, or **https://**. The length of the value cannot exceed 2 KB.	No
success-action-redirect	Indicates the address (URL) to which a successfully responded request is redirected. ● If the value is valid and the request is successful, OBS returns status code 303. **Location** contains **success_action_redirect** as well as the bucket name, object name, and object ETag. ● If this parameter is invalid, OBS ignores this parameter. The response code is 204, and the **Location** is the object address. Type: string	No
x-obs-expires	Indicates the expiration time of an object, in days. An object will be automatically deleted once it expires (calculated from the last modification time of the object). This field can be configured only when an object is uploaded and cannot be modified through the metadata modification API. Type: integer Example: x-obs-expires:3	No

Request Elements

This request contains no element. Its body contains only the content of the requested object.

Response Syntax

```
HTTP/1.1 status_code
Content-Length: length
Content-Type: type
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-55**.

Table 5-55 Additional response header parameters

Header	Description
x-obs-version-id	Object version ID. If versioning is enabled for the bucket, the object version number will be returned. Type: string

Response Elements

This response contains no element.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request 1

Upload an object.

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:11:15 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:gYqplLq30dEX7GMi2qFWyjdFsyw=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BF2600000164364C10805D385E1E3C67
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-id-2: 32AAAWJAMAABAAAQAAEAABAAAQAAEAABCTzu4Jp2lquWuXsjnLyPPiT3cfGhqPoY
Date: WED, 01 Jul 2015 04:11:15 GMT
Content-Length: 0
```

Sample Request 2

Set the ACL when uploading an object.

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:13:55 GMT
x-obs-grant-read:id=52f24s3593as5730ea4f722483579ai7,id=a93fcas852f24s3596ea8366794f7224
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:gYqplLq30dEX7GMi2qFWyjdFsyw=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB7800000164845759E4F3B39ABEE55E
ETag: "d41d8cd98f00b204e9800998ecf8427e"
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSReVRNuas0knI+Y96iXrZA7BLUgj06Z
Date: WED, 01 Jul 2015 04:13:55 GMT
Content-Length: 0
```

Sample Request 3

Upload objects when versioning is enabled for the bucket.

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:17:12 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

Sample Response 3

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001439A51DB2B2577
ETag: "d41d8cd98f00b204e9800998ecf8427e"
X-OBS-ID-2: GcVgfeOJHx8JZHTHrRqkPsbKdB583fYbr3RBbHT6mMrBstReVILBZbMAdLiBYy1l
Date: WED, 01 Jul 2015 04:17:12 GMT
x-obs-version-id: AAABQ4q2M9_c0vycq3gAAAAAVURTRkha
Content-Length: 0
```

Sample Request 4

MD5 is carried when an object is uploaded.

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:17:50 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10
Content-MD5: 6Afx/PgtEy+bsBjKZzihnw==
Expect: 100-continue
```

```
1234567890
```

Sample Response 4

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: BB7800000164B165971F91D82217D105
X-OBS-ID-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCSEKhBpS4BB3dSMNqMtuNxQDD9XvOw5h
ETag: "1072e1b96b47d7ec859710068aa70d57"
Date: WED, 01 Jul 2015 04:17:50 GMT
Content-Length: 0
```

Sample Request 5

The website hosting function is configured for the bucket. Configure redirection for the object download when uploading the object.

```
PUT /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:17:12 GMT
x-obs-website-redirect-location: http://www.example.com/
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:uFVJhp/dJqj/CJIVLrSZ0gpw3ng=
Content-Length: 10240
Expect: 100-continue

[1024 Byte data content]
```

Sample Response 5

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001439A51DB2B2577
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTmxB5ufMj/7/GzP8TFwTbp33u0xhn2Z
ETag: "1072e1b96b47d7ec859710068aa70d57"
Date: WED, 01 Jul 2015 04:17:12 GMT
x-obs-version-id: AAABQ4q2M9_c0vycq3gAAAAAVURTRkha
Content-Length: 0
```

Sample Request 6

Upload an object and carry the signature in the URL.

```
PUT /object02?
AccessKeyId=H4IPJX0TQTHTHEBQQCEC&Expires=1532688887&Signature=EQmDuOhWLUrzrzRNZxwS72CXeXM%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Content-Length: 1024

[1024 Byte data content]
```

Sample Response 6

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001439A51DB2B2577
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTmxB5ufMj/7/GzP8TFwTbp33u0xhn2Z
ETag: "1072e1b96b47d7ec859710068aa70d57"
Date: Fri, 27 Jul 2018 10:52:31 GMT
x-obs-version-id: AAABQ4q2M9_c0vycq3gAAAAAVURTRkha
Content-Length: 0
```

5.4.2 Uploading Objects - POST

Functions

Uploading an object means to add an object to a bucket. To perform this operation, you must have the write permission for the bucket.

NOTE

The name of each object in a bucket must be unique.

With versioning not enabled, if an object to be uploaded has the same name as an existing object in the bucket, the newly uploaded object will overwrite the existing one. To protect data from being corrupted during transmission, you can add the **Content-MD5** parameter in the form field. After receiving the request, OBS will perform an MD5 consistency check. If the two MD5 values are inconsistent, the system returns an error message. You can also specify the value of the **x-obs-acl** parameter to configure an access control policy for the object.

You can also upload an object using the POST method.

For a single upload, the size of the object to be uploaded ranges [0, 5 GB]. To upload a file greater than 5 GB, see **Operations on Multipart Upload**.

Differences Between PUT and POST Methods

Parameters are passed through the request header if the PUT method is used to upload objects; if the POST method is used to upload objects, parameters are passed through the form field in the message body.

With the PUT method, you need to specify the object name in the URL, but object name is not required with the POST method, which uses the bucket domain name as the URL. The request lines of the two methods are as follows:

```
PUT /ObjectName HTTP/1.1
POST / HTTP/1.1
```

For details about PUT upload, see **Uploading Objects - PUT**.

Versioning

If versioning is enabled for a bucket, the system automatically generates a unique version ID for the requested object in this bucket and returns the version ID in response header **x-obs-version-id**. If versioning is suspended for a bucket, the version ID of the requested object in this bucket is **null**. For details about the versioning statuses of a bucket, see **Configuring Versioning for a Bucket**.

Request Syntax

```
POST / HTTP/1.1
Host: bucketname.obs.region.example.com
User-Agent: browser_data
Accept: file_types
Accept-Language: Regions
Accept-Encoding: encoding
Accept-Charset: character_set
Keep-Alive: 300
Connection: keep-alive
```

```
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: length


--9431149156168
Content-Disposition: form-data; name="key"

acl
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

success_redirect
--9431149156168
Content-Disposition: form-data; name="content-Type"

content_type
--9431149156168
Content-Disposition: form-data; name="x-obs-meta-uuid"

uuid
--9431149156168
Content-Disposition: form-data; name="x-obs-meta-tag"

metadata
--9431149156168
Content-Disposition: form-data; name="AccessKeyId"

access-key-id
--9431149156168
Content-Disposition: form-data; name="policy"

encoded_policy
--9431149156168
Content-Disposition: form-data; name="signature"

signature=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename"
Content-Type: image/jpeg

file_content
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to OBS
--9431149156168--
```

Request Parameters

This request contains no parameter.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

If you want to get CORS configuration information, you must use the headers in **Table 5-56**.

Table 5-56 Request headers for obtaining CORS configuration

Header	Description	Mandatory
Origin	Origin of the cross-domain request specified by the pre-request. Generally, it is a domain name. Type: string	Yes
Access-Control-Request-Headers	Indicates the HTTP headers of a request. The request can use multiple HTTP headers. Type: string	No

Request Elements

This request uses form elements. **Table 5-57** describes the form elements.

Table 5-57 Form elements

Parameter	Description	Mandatory
file	Indicates the content of the object to be uploaded. Type: binary content or text Constraint: This parameter must be the last parameter in a form. Otherwise, parameters after this parameter will be all discarded. Additionally, each request contains only one file parameter.	Yes
key	Indicates the name of the object to be created. Type: string	Yes
AccessKeyId	Indicates the access key (AK) of the requester. Type: string Constraint: This parameter is mandatory if there is security policy parameter **policy** or **signature** in the request.	Yes when the constraint is met.

Parameter	Description	Mandatory
policy	Indicates the security policy in the request. For details about the policy format, see the policy format in **Authentication of Signature Carried in the Table Uploaded Through a Browser**. Type: string Constraint: This parameter is mandatory if the bucket provides the **AccessKeyId** (or **signature**).	Yes when the constraint is met.
signature	Indicates a signature string calculated based on StringToSign. Type: string Constraint: This parameter is mandatory if the bucket provides the **AccessKeyId** (or **policy**).	Yes when the constraint is met.
token	Specifies the AK, signature, and security policy of the request initiator. The priority of a token is higher than that of a specified AK, the request signature, and the security policy of the request initiator. Type: string Example: HTML: <input type= "text" name="token" value="ak:signature:policy" />	No
x-obs-acl	When creating an object, you can add this message header to set the permission control policy for the object. The predefined common policies are as follows: **private**, **public-read**, **public-read-write**, **public-read-delivered**, and **public-read-write-delivered**. Type: string An example is provided as follows: Policy: {"acl": "public-read" } HTML: <input type="text" name="acl" value="public-read" />	No

Parameter	Description	Mandatory
x-obs-grant-read	When creating an object, you can use this header to authorize all users in an account the permission to read objects and obtain object metadata. Type: string An example is provided as follows: In POLICY: {'grant-read': 'id=domainId1' }, In HTML: <input type="text" name="grant-read" value="id=domainId1" />	No
x-obs-grant-read-acp	When creating an object, you can use this header to authorize all users in an account the permission to obtain the object ACL. Type: string An example is provided as follows: In POLICY: {"grant-read-acp": "id=domainId1" }, In HTML: <input type="text" name="grant-read-acp" value="id=domainId1" />	No
x-obs-grant-write-acp	When creating an object, you can use this header to authorize all users in an account the permission to write the object ACL. Type: string An example is provided as follows: In POLICY: {"grant-write-acp": "id=domainId1" }, In HTML: <input type="text" name="grant-write-acp" value="id=domainId1" />	No
x-obs-grant-full-control	When creating an object, you can use this header to authorize all users in an account the permission to read the object, obtain the object metadata, obtain the object ACL, and write the object ACL. Type: string An example is provided as follows: In POLICY: {"grant-full-control": "id=domainId1" }, In HTML: <input type="text" name="grant-full-control" value="id=domainId1" />	No

Parameter	Description	Mandatory
Cache-Control, Content-Type, Content-Disposition, Content-Encoding Expires	Standard HTTP headers. OBS records those headers. If you download the object or send the HEAD Object request, those parameter values are returned. Type: string An example is provided as follows: In POLICY: ["starts-with", "$Content-Type", "text/"], In HTML: <input type="text" name="content-type" value="text/plain" />	No
success_action_redirect	Indicates the address (URL) to which a successfully responded request is redirected. • If the value is valid and the request is successful, OBS returns status code 303. **Location** contains **success_action_redirect** as well as the bucket name, object name, and object ETag. • If this parameter is invalid, OBS ignores this parameter. The response code is 204, and the **Location** is the object address. Type: string An example is provided as follows: In POLICY: {"success_action_redirect": "http://123458.com"}, In HTML: <input type="text" name="success_action_redirect" value="http://123458.com" />	No
x-obs-meta-*	Indicates user-defined metadata. When creating an object, you can use this header or a header starting with **x-obs-meta-** to define object metadata in an HTTP request. User-defined metadata will be returned in the response header when you retrieve or query the metadata of the object. Type: string An example is provided as follows: In POLICY: {" x-obs-meta-test ": " test metadata " }, In HTML: <input type="text" name=" x-obs-meta-test " value=" test metadata " />	No

Parameter	Description	Mandatory
success_action_status	Indicates the status code returned after the request is successfully received. Possible values are **200**, **201**, and **204**. • If this parameter is set to **200** or **204**, the body in the OBS response message is empty. • If this parameter is set to **201**, the OBS response message contains an XML document that describes the response to the request. • If the value is not set or if it is set to an invalid value, the OBS returns an empty document with a 204 status code. Type: string An example is provided as follows: In POLICY: ["starts-with", "$success_action_status", ""], In HTML: <input type="text" name="success_action_status" value="200" />	No
x-obs-website-redirect-location	If a bucket is configured with the static website hosting function, it will redirect requests for this object to another object in the same bucket or to an external URL. OBS stores the value of this header in the object metadata. There is no default value. Constraint: The value must be prefixed by a slash (/), **http://**, or **https://**. The length of the value cannot exceed 2 KB.	No
x-obs-expires	Indicates the expiration time of an object, in days. An object will be automatically deleted once it expires (calculated from the last modification time of the object). Type: integer Example: x-obs-expires:3	No

Response Syntax

```
HTTP/1.1 status_code
Content-Type: application/xml
Location: location
Date: date
ETag: etag
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-58**.

Table 5-58 Additional response header parameters

Header	Description
x-obs-version-id	Object version ID. If versioning is enabled for the bucket, the object version number will be returned. A string **null** will be returned if the bucket housing the object has versioning suspended. Type: string
Access-Control-Allow-Origin	Indicates that the origin is included in the response if the origin in the request meets the CORS configuration requirements when CORS is configured for buckets. Type: string
Access-Control-Allow-Headers	Indicates that the headers are included in the response if headers in the request meet the CORS configuration requirements when CORS is configured for buckets. Type: string
Access-Control-Max-Age	Indicates MaxAgeSeconds in the CORS configuration of the server when CORS is configured for buckets. Type: integer
Access-Control-Allow-Methods	Indicates that methods in the rule are included in the response if Access-Control-Request-Method in the request meets the CORS configuration requirements when CORS is configured for buckets. Type: string Possible values are GET, PUT, HEAD, POST, and DELETE.
Access-Control-Expose-Headers	Indicates **ExposeHeader** in the CORS configuration of a server when CORS is configured for buckets. Type: string

Header	Description
x-obs-server-side-encryption	This header is included in a response if SSE-KMS is used. Type: string Example: x-obs-server-side-encryption:kms

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request 1

Common POST upload

```
POST / HTTP/1.1
Date: WED, 01 Jul 2015 04:15:23 GMT
Host: examplebucket.obs.region.example.com
Content-Type: multipart/form-data; boundary=7db143f50da2
Content-Length: 2424
Origin: www.example.com
Access-Control-Request-Headers:acc_header_1

--7db143f50da2
Content-Disposition: form-data; name="key"

object01
--7db143f50da2
Content-Disposition: form-data; name="acl"

public-read
--7db143f50da2
Content-Disposition: form-data; name="content-type"

text/plain
--7db143f50da2
Content-Disposition: form-data; name="expires"

WED, 01 Jul 2015 04:16:15 GMT
--7db143f50da2
Content-Disposition: form-data; name="AccessKeyId"

14RZT432N80TGDF2Y2G2
--7db143f50da2
Content-Disposition: form-data; name="policy"

ew0KICAiZXhwaXJhdGlvbiI6ICIyMDE1LTA3LTAxVDEyOjAwOjAwLjAwMFoiLA0KICAiY29uZGl0aW9ucyI6IFsNCi
AgICB7ImJ1Y2tldCI6ICJleG1hcGxlYnVja2V0IiB9LA0KICAgIHsiYWNsIjogInB1YmxpYy1yZWFkIiB9LA0KICAgIHsiR
XhwaXJlcyI6ICIxMDAwIiB9LA0KICAgIFsiZXEiLCAiJGtleSIsICJvYmplY3QwMSJdLA0KICAgIFsic3RhcnRzLXdpdGgiL
CAiJENvbnRlbnQtVHlwZSIsICJ0ZXh0LyJdLA0KICBdDQp9DQo=
--7db143f50da2
Content-Disposition: form-data; name="signature"

Vk6rwO0Nq09BLhvNSIYwSJTRQ+k=
--7db143f50da2
Content-Disposition: form-data; name="file"; filename="C:\Testtools\UpLoadFiles\object\1024Bytes.txt"
```

```
Content-Type: text/plain

01234567890
--7db143f50da2
Content-Disposition: form-data; name="submit"

Upload
--7db143f50da2--
```

Sample Response 1

After CORS is configured for a bucket, the response contains the **Access-Control-*** information.

```
HTTP/1.1 204 No Content
x-obs-request-id: 90E2BA00C26C00000133B442A90063FD
x-obs-id-2: OTBFMkJBMDBDMjZDMDAwMDAxMzNCNDQyQTkwMDYzRkRBQUFBQUFBQWJiYmJiYmJi
Access-Control-Allow-Origin: www.example.com
Access-Control-Allow-Methods: POST,GET,HEAD,PUT
Access-Control-Allow-Headers: acc_header_01
Access-Control-Max-Age: 100
Access-Control-Expose-Headers: exp_header_01
Content-Type: text/xml
Location: http://examplebucket.obs.region.example.com/object01
Date: WED, 01 Jul 2015 04:15:23 GMT
ETag: "ab7abb0da4bca5323ab6119bb5dcd296"
```

5.4.3 Copying Objects

Functions

You can perform this operation to create a copy of an existing object in OBS.

Users can determine whether to copy the metadata of the source object to the target object (by default) or replace the metadata of the target object with the metadata contained in the request. The ACL of the source object is not copied to the target object. By default, the ACL of the target object is private. You can set an ACL for the target object by sending an API request.

The request for copying an object needs to carry the information about the bucket and object to be copied in the header field. The message body cannot be carried.

The target object size range is [0, 5 GB]. If the source object size exceeds 5 GB, you can only copy some objects using the Range API.

Versioning

By default, **x-obs-copy-source** specifies the latest version of the source object. If the latest version of the source object has a deletion marker, the object is considered to have been deleted. You can add **versionId** to request header **x-obs-copy-source** to copy an object with the specified version ID.

If a bucket has versioning enabled, the system automatically generates a unique version ID for the requested object in this bucket and returns the version ID in response header **x-obs-version-id**. If versioning is suspended for the bucket, the object version number is **null**.

NOTICE

When the bucket versioning status is disabled, if you make a copy of object_A and save it as object_B, and an object named as object_B already exists, the new object_B will overwrite the existing one. After the copying is executed successfully, only new object_B can be downloaded because old object_B has been deleted. Therefore, before copying an object, ensure that there is no object with the same name as the object copy to prevent data from being deleted mistakenly. During the copying, object_A has no changes.

You cannot determine whether a request is executed successfully only using **status_code** in the header returned by HTTP. If 200 in **status_code** is returned, the server has received the request and starts to process the request. The body in the response shows whether the request is executed successfully. The request is executed successfully only when the body contains Etag; otherwise, the request fails to be executed.

Request Syntax

```
PUT /destinationObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
x-obs-copy-source: /sourceBucket/sourceObject
x-obs-metadata-directive: metadata_directive
x-obs-copy-source-if-match: etag
x-obs-copy-source-if-none-match: etag
x-obs-copy-source-if-unmodified-since: time_stamp
x-obs-copy-source-if-modified-since: time_stamp
Authorization: signature
Date: date
```

Request Parameters

This request contains no parameter.

Request Headers

You can add optional headers to specify the object to be copied. **Table 3-3** describes the optional headers.

Table 5-59 Request headers

Header	Description	Mandatory
x-obs-acl	This header can be added to set access control policies for objects when copying the objects. The access control policies are the predefined common policies, including **private**, **public-read**, **public-read-write**. Type: string Example: x-obs-acl: acl	No

Header	Description	Mandatory
x-obs-grant-read	When creating an object, you can use this header to authorize all users in an account the permission to read objects and obtain object metadata. Type: string	No
x-obs-grant-read-acp	When creating an object, you can use this header to authorize all users in an account the permission to obtain the object ACL. Type: string	No
x-obs-grant-write-acp	When creating an object, you can use this header to authorize all users in an account the permission to write the object ACL. Type: string	No
x-obs-grant-full-control	When creating an object, you can use this header to authorize all users in an account the permission to read the object, obtain the object metadata, obtain the object ACL, and write the object ACL. Type: string	No
x-obs-copy-source	Indicates names of the source bucket and the source object. If the source object has multiple versions, the versionId parameter can be used to specify the desired version. Type: string Constraint: URL encoding is required for handling Chinese characters. Example: x-obs-copy-source: /source_bucket/sourceObject	Yes

Header	Description	Mandatory
x-obs-metadata-directive	Indicates whether the metadata of the target object is copied from the source object or replaced with the metadata contained in the request. Type: string Valid values: COPY and REPLACE Default value: COPY Example: x-obs-metadata-directive: metadata_directive Constraints: Values other than **COPY** or **REPLACE** result in an immediate 400-based error response. If you need to modify the metadata (the same for both the source and target objects), this parameter must be set to **REPLACE**, otherwise, the request is invalid and the server returns a 400 HTTP status code error. This parameter cannot be used to change an encrypted object to a non-encrypted object (the same for both the source and target objects). If you use this parameter to change the encrypted object, the system returns 400.	No
x-obs-copy-source-if-match	Copies the source object only if its ETag matches the one specified by this header. Otherwise, a 412 HTTP status code error (failed precondition) is returned. Type: string Example: x-obs-copy-source-if-match: etag Constraint: This parameter can be used with **x-obs-copy-source-if-unmodified-since** but not other conditional copy parameters.	No
x-obs-copy-source-if-none-match	Copies the object if its entity tag (ETag) matches the specified tag. Otherwise, the request returns a 412 HTTP status code error (failed precondition). Type: string Example: x-obs-copy-source-if-none-match: etag Constraint: This parameter can be used with **x-obs-copy-source-if-unmodified-since** but not other conditional copy parameters.	No

Header	Description	Mandatory
x-obs-copy-source-if-unmodified-since	Copies the source object only if it has not been modified since the time specified by this header. Otherwise, a 412 HTTP status code error (failed precondition) is returned. This header can be used with **x-obs-copy-source-if-match**, but cannot be used with other conditional copy headers Type: HTTP time character string complying with the format specified at **http://www.ietf.org/rfc/rfc2616.txt** Example: x-obs-copy-source-if-unmodified -since: time-stamp	No
x-obs-copy-source-if-modified-since	Copies the source object only if it has not been modified since the time specified by this header. Otherwise, a 412 HTTP status code error (failed precondition) is returned. This header can be used with **x-obs-copy-source-if-none-match**, but cannot be used with other conditional copy headers Type: HTTP time character string complying with the format specified at **http://www.ietf.org/rfc/rfc2616.txt** Example: x-obs-copy-source-if-modified-since: time-stamp	No
x-obs-website-redirect-location	If a bucket is configured with the static website hosting function, it will redirect requests for this object to another object in the same bucket or to an external URL. OBS stores the value of this header in the object metadata. Type: string There is no default value. Constraint: The value must be prefixed by a slash (/), **http://**, or **https://**. The length of the value cannot exceed 2 KB.	No

Header	Description	Mandatory
success_action_redirect	Indicates the address (URL) to which a successfully responded request is redirected. • If the value is valid and the request is successful, OBS returns status code 303. **Location** contains **success_action_redirect** as well as the bucket name, object name, and object ETag. • If this parameter is invalid, OBS ignores this parameter. The response code is 204, and the **Location** is the object address. Type: string	No

For details about other headers, see **Table 3-3**.

Request Elements

This request contains no element.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: application/xml
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CopyObjectResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <LastModified>modifiedDate</LastModified>
   <ETag>etagValue</ETag>
</CopyObjectResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-60**.

Table 5-60 Additional response header parameters

Header	Description
x-obs-copy-source-version-id	Version ID of the source object Type: string
x-obs-version-id	Version ID of the target object Type: string

Response Elements

This response contains elements of a copy result. **Table 5-61** describes the elements.

Table 5-61 Response elements

Element	Description
CopyObjectResult	Container for the copy result Type: XML
LastModified	Latest time when the object was modified Type: string
ETag	128-bit MD5 digest of the Base64 code of a new object. ETag is the unique identifier of the object content. It can be used to identify whether the object content is changed. For example, if ETag value is **A** when an object is uploaded and the ETag value has changed to **B** when the object is downloaded, it indicates that the object content is changed. Type: string

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request 1

Copy the object **srcobject** in bucket **bucket** to the **destobject** object in bucket **examplebucket**.

```
PUT /destobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:19:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:2rZR+iaH8xUewvUKuicLhLHpNoU=
x-obs-copy-source: /bucket/srcobject
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 001B21A61C6C00000134031BE8005293
x-obs-id-2: MDAxQjIxQTYxQzZDMDAwMDAxMzQwMzFCRTgwMDUyOTNBQUFBQUFBQWJiYmJiYmJi
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 249

<?xml version="1.0" encoding="utf-8"?>
<CopyObjectResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <LastModified>2015-07-01T00:48:07.706Z</LastModified>
  <ETag>"507e3fff69b69bf57d303e807448560b"</ETag>
</CopyObjectResult>
```

Sample Request 2

Copy a multi-version object and copy the object **srcobject** whose version number is **AAABQ4uBLdLc0vycq3gAAAAEVURTRkha** in bucket **bucket** to the **destobject** object in bucket **examplebucket**.

```
PUT /destobject HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:20:29 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:4BLYv+1UxfRSHBMvrhVLDszxvcY=
x-obs-copy-source: /bucket/srcobject?versionId=AAABQ4uBLdLc0vycq3gAAAAEVURTRkha
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: DCD2FC9CAB78000001438B8A9C898B79
x-obs-id-2: DB/qBZmbN6AIoX9mrrSNYdLxwvbO0tLR/l6/XKTT4NmZspzhWrwp5Z74ybAYVOgr
Content-Type: application/xml
x-obs-version-id: AAABQ4uKnOrc0vycq3gAAAAFVURTRkha
x-obs-copy-source-version-id: AAABQ4uBLdLc0vycq3gAAAAEVURTRkha
Date: WED, 01 Jul 2015 04:20:29 GMT
Transfer-Encoding: chunked

<?xml version="1.0" encoding="utf-8"?>
<CopyObjectResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <LastModified>2015-07-01T01:48:07.706Z</LastModified>
  <ETag>"507e3fff69b69bf57d303e807448560b"</ETag>
</CopyObjectResult>
```

5.4.4 Downloading Objects

Functions

This operation downloads objects from OBS. Before using this GET operation, check that you have the read permission for the target object. If the object owner has granted anonymous users the read permission for the object, anonymous users can access this object without using the authentication header field.

Versioning

By default, the GET operation returns the current version of an object. If the current version of the object is a deletion marker, OBS returns a code meaning non-existence of the object. To obtain an object of a specified version, the **versionId** parameter can be used to specify the desired version.

Request Syntax

```
GET /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Range:bytes=byte_range
<Optional Additional Header>
```

NOTE

The field is optional. If it does not exist, you can obtain the whole content.

Request Parameters

In a **GET** request, you can override values for a set of message headers using the request parameters. Message headers that you can override are **Content-Type**, **Content-Language**, **Expires**, **Cache-Control**, **Content-Disposition**, and **Content-Encoding**. If the target object has multiple versions, use the **versionId** parameter to specify the version to be downloaded. For details, see **Table 5-62**.

NOTE

OBS does not process Accept-Encoding carried in a request or compress or decompress the uploaded data. The client determines whether to compress or decompress the data. Some HTTP clients may decompress data based on the Content-Encoding returned by the server. The client program needs to determine whether to decompress and how to decompress the data. To decompress the data, it can modify Content-Encoding (the object metadata stored in OBS) or rewrite Content-Encoding the object is downloaded. If an object download request specifies the rewrite header, the standard HTTP message header returned by OBS is subject to the rewrite content specified in the request.

Table 5-62 Request parameters

Parameter	Description	Mandatory
response-content-type	Rewrites the **Content-Type** header in the response. Type: string	No
response-content-language	Rewrites the **Content-Language** header in the response. Type: string	No
response-expires	Rewrites the **Expires** header in the response. Type: string	No
response-cache-control	Rewrites the **Cache-Control** header in the response. Type: string	No
response-content-disposition	Rewrites the **Content-Disposition** header in the response. Type: string Example: response-content-disposition=attachment; filename*=utf-8''name1 In this example, the downloaded object is renamed **name1**. If the new name contains Chinese characters, it must be URL-encoded.	No

Parameter	Description	Mandatory
response-content-encoding	Rewrites the **Content-Encoding** header in the response. Type: string	No
versionId	Indicates the version ID of the object whose content is obtained. Type: string	No
attname	Rewrites the **Content-Disposition** header in the response. Type: string Example: attname=name1 Rename the downloaded object as **name1**.	No

Request Headers

This request uses common headers. In addition, you can add additional headers to this request. **Table 5-63** describes the additional headers.

Table 5-63 Request headers

Header	Description	Mandatory
Range	Obtains the object content within the scope defined by **Range**. If the parameter value is invalid, the entire object is obtained. **Range** value starts from 0, and the maximum value equals the object length minus 1. The start value of **Range** is mandatory. If only the start value is specified, the system obtains the object content from the start value to default maximum value. After the **Range** header field is carried, the value of ETag in the response message is the ETag of the object instead of the ETag of the object in the **Range** field. Type: string bytes=byte_range Example 1: **bytes=0-4** Example 2: **bytes=1024** Example 3: **bytes=10-20, 30-40** (multiple ranges)	No
If-Modified-Since	Returns the object only if it has been modified since the time specified by this header. Otherwise, **304 Not Modified** is returned. Type: HTTP time character string complying with the format specified at **http://www.ietf.org/rfc/rfc2616.txt**	No
If-Unmodified-Since	Returns the object only if it has not been modified since the time specified by this header. Otherwise, **412 Precondition Failed** is returned. Type: HTTP time character string complying with the format specified at **http://www.ietf.org/rfc/rfc2616.txt**	No
If-Match	Returns the object only if its ETag is the same as the one specified by this header. Otherwise, **412 Precondition Failed** is returned. Type: string (Example: 0f64741bf7cb1089e988e4585d0d3434)	No

Header	Description	Mandatory
If-None-Match	Returns the object only if its ETag is different from the one specified by this header. Otherwise, **304 Not Modified** is returned. Type: string (Example: 0f64741bf7cb1089e988e4585d0d3434)	No

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: type
Date: date
Content-Length: length
Etag: etag
Last-Modified: time

<Object Content>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-64**.

Table 5-64 Additional response header parameters

Header	Description
x-obs-expiration	When an object has its lifecycle rule, the object expiration time is subject to its lifecycle rule. This header field is use **expiry-date** to describe the object expiration date. If the lifecycle rule is configured only for the entire bucket not individual objects, the object expiration time is subject to the bucket lifecycle rule. This header field uses the **expiry-date** and **rule-id** to describe the detailed expiration information of objects. If no lifecycle rule is configured, this header field is not contained in the response. Type: string

Header	Description
x-obs-website-redirect-location	Indicates the redirected-to location. If the bucket is configured with website information, this parameter can be set for the object metadata so that the website endpoint will evaluate the request for the object as a 301 redirect to another object in the same bucket or an external URL. Type: string
x-obs-delete-marker	Indicates whether an object is a deletion marker. If the object is not marked as deleted, the response does not contain this header. Type: boolean Valid values: **true** or **false** The default value is **false**.
x-obs-version-id	Object version ID. If the object has no version number specified, the response does not contain this header. Valid value: character string There is no default value.
x-obs-object-type	If the object is not a normal one, this header field is returned. The value can be **Appendable**. Type: string
x-obs-next-append-position	This header field is returned when the object is an appendable object. Type: integer

Response Elements

This response contains no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request 1

Download the entire object.

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
Accept-Ranges: bytes
ETag: "3b46eaf02d3b6b1206078bb86a7b7013"
Last-Modified: WED, 01 Jul 2015 01:20:29 GMT
Content-Type: binary/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQwxJ2I1VvxD/Xgwuw2G2RQax30gdXU
Date: WED, 01 Jul 2015 04:24:33 GMT
Content-Length: 4572

[4572 Bytes object content]
```

Sample Request 2

Download the specified range of an object (download a range of an object).

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Sep 2020 09:59:04 GMT
Range:bytes=20-30
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:mNPLWQMDWg30PTkAWiqJaLl3ALg=
```

Download the specified range of an object (download multiple ranges of an object).

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Mon, 14 Sep 2020 10:02:43 GMT
Range:bytes=20-30,40-50
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:ZwM7Vk2d7sD9o8zRsRKehgKQDkk=
```

Sample Response 2

Download the specified range of an object (download a range of an object).

```
HTTP/1.1 206 Partial Content
Server: OBS
x-obs-request-id: 000001748C0DBC35802E360C9E869F31
Accept-Ranges: bytes
ETag: "2200446c2082f27ed2a569601ca4e360"
Last-Modified: Mon, 14 Sep 2020 01:16:20 GMT
Content-Range: bytes 20-30/4583
Content-Type: binary/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSn2JHu4okx9NBRNZAvBGaws3lt3g31g
Date: Mon, 14 Sep 2020 09:59:04 GMT
Content-Length: 11

[ 11 Bytes object content]
```

Download the specified range of an object (download multiple ranges of an object).

```
HTTP/1.1 206 Partial Content
Server: OBS
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
Accept-Ranges: bytes
ETag: "2200446c2082f27ed2a569601ca4e360"
Last-Modified: Mon, 14 Sep 2020 01:16:20 GMT
Content-Type: multipart/byteranges;boundary=35bcf444-e65f-4c76-9430-7e4a68dd3d26
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSIBWFOVW8eeWujkqSnoIANC2mNR1cdF
```

```
Date: Mon, 14 Sep 2020 10:02:43 GMT
Content-Length: 288

--35bcf444-e65f-4c76-9430-7e4a68dd3d26
Content-type: binary/octet-stream
Content-range: bytes 20-30/4583
[ 11 Bytes object content]
--35bcf444-e65f-4c76-9430-7e4a68dd3d26
Content-type: binary/octet-stream
Content-range: bytes 40-50/4583
[ 11 Bytes object content]
--35bcf444-e65f-4c76-9430-7e4a68dd3d26
```

Sample Request 4

Download an object if its Etag value matches.

```
GET /object01 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
If-Match: 682e760adb130c60c120da3e333a8b09
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

Sample Response 4-1 (Non-Matched Etag)

If the Etag value of the stored object is not **682e760adb130c60c120da3e333a8b09**, the system displays a message indicating that the download fails.

```
HTTP/1.1 412 Precondition Failed
Server: OBS
x-obs-request-id: 8DF400000163D3F2A89604C49ABEE55E
Content-Type: application/xml
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSQwxJ2I1VvxD/Xgwuw2G2RQax30gdXU
Date: WED, 01 Jul 2015 04:20:51 GMT

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Error>
  <Code>PreconditionFailed</Code>
  <Message>At least one of the pre-conditions you specified did not hold</Message>
  <RequestId>5DEB00000164A214CEC54C30A3A769E3</RequestId>
  <HostId>Hw0ZGaSKVm+uLOrCXXtx4Qn1aLzvoeblctVXRAqA7pty10mzUUW/yOzFue04lBqu</HostId>
  <Condition>If-Match</Condition>
</Error>
```

Sample Response 4-2 (Successful Download)

If the Etag value of the stored object is **682e760adb130c60c120da3e333a8b09**, the download is successful.

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 5DEB00000164A21E1FC826C58F6BA001
Accept-Ranges: bytes
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSbkdml1sLSvKnoHaRcOwRI+6+ustDwk
Date: Mon, 16 Jul 2015 08:04:00 GMT
Content-Length: 8

[ 8 Bytes object content]
```

Sample Request 5

Carry the signature in the URL when downloading an object.

```
GET /object02?
AccessKeyId=H4IPJX0TQTHTHEBQQCEC&Expires=1532688887&Signature=EQmDuOhWLUrzrzRNZxwS72CXeXM%3D HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: Fri, 27 Jul 2018 10:52:31 GMT
```

Sample Response 5

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Date: Fri, 27 Jul 2018 10:52:31 GMT
Content-Length: 8

[ 8 Bytes object content]
```

Sample Request 6

Use the response-content-disposition parameter to download and rename an object.

```
GET /object01?response-content-disposition=attachment; filename*=utf-8''name1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

Sample Response 6

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Date: Fri, 27 Jul 2018 10:52:31 GMT
Content-Length: 8
Content-Disposition: attachment; filename*=utf-8''name1

[ 8 Bytes object content]
```

Sample Request 7

Use the attname parameter to download and rename an object.

```
GET /object01?attname=name1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:24:33 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:NxtSMS0jaVxlLnxlO9awaMTn47s=
```

Sample Response 7

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 804F00000164DB5E5B7FB908D3BA8E00
ETag: "682e760adb130c60c120da3e333a8b09"
Last-Modified: Mon, 16 Jul 2015 08:03:34 GMT
Content-Type: application/octet-stream
x-obs-id-2: 32AAAUJAIAABAAAQAAEAABAAAQAAEAABCTlpxILjhVK/heKOWIP8Wn2IWmQoerfw
Date: Fri, 27 Jul 2018 10:52:31 GMT
Content-Length: 8
Content-Disposition: attachment; filename*=utf-8''name1

[ 8 Bytes object content]
```

5.4.5 Querying Object Metadata

Functions

Users with the read permission on objects can perform the HeadObject operation to obtain metadata of objects. The object metadata is included in the response.

Versioning

By default, this operation returns the latest metadata of an object. If the object has a deletion marker, status code 404 is returned. To obtain the object metadata of a specified version, the **versionId** parameter can be used to specify the desired version.

Request Syntax

```
HEAD /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

Table 5-65 describes the request parameters.

Table 5-65 Request parameters

Parameter	Description	Mandatory
versionId	Object version ID Type: string	No

Request Headers

This request uses common headers. For details, see **Table 3-3**.

In addition, the request can use additional headers, as shown in **Table 5-66**.

Table 5-66 Request headers

Header	Description	Mandatory
Origin	Origin of the cross-domain request specified by the pre-request. Generally, it is a domain name. Type: string	Yes
Access-Control-Request-Headers	Indicates the HTTP headers of a request. The request can use multiple HTTP headers. Type: string	No

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Content-Type: type
Date: date
Content-Length: length
Etag: etag
Last-Modified: time
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-67**.

Table 5-67 Additional response header parameters

Header	Description
x-obs-expiration	When an object has its lifecycle rule, the object expiration time is subject to its lifecycle rule. This header field is use **expiry-date** to describe the object expiration date. If the lifecycle rule is configured only for the entire bucket not individual objects, the object expiration time is subject to the bucket lifecycle rule. This header field uses the **expiry-date** and **rule-id** to describe the detailed expiration information of objects. If no lifecycle rule is configured, this header field is not contained in the response. Type: string

Header	Description
x-obs-website-redirect-location	Indicates the redirected-to location. If the bucket is configured with website information, this parameter can be set for the object metadata so that the website endpoint will evaluate the request for the object as a 301 redirect to another object in the same bucket or an external URL. Type: string
x-obs-version-id	Object version ID. If the object has no version number specified, the response does not contain this header. Type: string There is no default value.
Access-Control-Allow-Origin	Indicates that the origin is included in the response if the origin in the request meets the CORS configuration requirements when CORS is configured for buckets. Type: string
Access-Control-Allow-Headers	Indicates that the headers are included in the response if headers in the request meet the CORS configuration requirements when CORS is configured for buckets. Type: string
Access-Control-Max-Age	Value of **MaxAgeSeconds** in the CORS configuration of the server when CORS is configured for buckets. Type: integer
Access-Control-Allow-Methods	Indicates that methods in the rule are included in the response if Access-Control-Request-Method in the request meets the CORS configuration requirements when CORS is configured for buckets. Type: string Possible values are GET, PUT, HEAD, POST, and DELETE.
Access-Control-Expose-Headers	Value of **ExposeHeader** in the CORS configuration of a server when CORS is configured for buckets. Type: string

Header	Description
x-obs-object-type	If the object is not a normal one, this header field is returned. The value can be **Appendable** Type: string
x-obs-next-append-position	This header field is returned when the object is an appendable object. Type: integer
x-obs-uploadId	This header is returned if the object is a combination of multiple parts. The header value indicates the ID of the corresponding multipart upload task. Type: string

Response Elements

This response contains no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
HEAD /object1 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:19:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:/cARjk81l2iExMfQqn6iT3qEZ74=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E4BB5905C41B6E65B6
Accept-Ranges: bytes
ETag: "3b46eaf02d3b6b1206078bb86a7b7013"
Last-Modified: WED, 01 Jul 2015 01:19:21 GMT
Content-Type: binary/octet-stream
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSD3nAiTaBoeyt9oHp9vTYtXnLDmwV6D
Date: WED, 01 Jul 2015 04:19:21 GMT
Content-Length: 4572
```

5.4.6 Deleting an Object

Functions

You can perform this operation to delete an object. If you try to delete an object that does not exist, OBS will return a success message.

Versioning

When versioning is enabled for a bucket, a deletion marker with a unique version number is generated when an object is deleted without specifying the version. However, the object is not actually deleted. If versioning is suspended for a bucket and no version is specified when you delete an object, the object whose version number is **null** is deleted, and a deletion marker with version number **null** is generated.

To delete an object of a specified version, the **versionId** parameter can be used to specify the desired version.

Request Syntax

```
DELETE /ObjectName HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

Table 5-68 describes the request parameters.

NOTICE

For deleting an object, only parameters listed in **Table 5-68** are supported. If the request contains parameters that cannot be identified by OBS, the server returns the 400 error code.

Table 5-68 Request parameters

Parameter	Description	Mandatory
versionId	Object version ID Type: string	No

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-69**.

Table 5-69 Additional response header parameters

Header	Description
x-obs-delete-marker	Indicates whether an object is deleted. If the object is not marked as deleted, the response does not contain this header. Type: boolean Valid values: **true** or **false** The default value is **false**.
x-obs-version-id	Object version ID. If the object has no version number specified, the response does not contain this header. Valid value: character string There is no default value.

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
DELETE /object2 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:19:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:MfK9JCnSFHCrJmjv7iRkRrrce2s=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 8DF400000163D3F51DEA05AC9CA066F1
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCSgkM4Dij80gAeFY8pAZIwx72QhDeBZ5
Date: WED, 01 Jul 2015 04:19:21 GMT
```

5.4.7 Deleting Objects

Functions

This operation can be used to batch delete some objects in a bucket. The deletion cannot be undone. After the operation is implemented, the returned information contains the implementation result of each object in the specified bucket. OBS deletes the objects synchronously. The deletion result of each object is returned to the request user.

Objects in batches can be deleted in **verbose** or **quiet** mode. With **verbose** mode, OBS returns results of successful and failed deletion in an XML response; with **quiet** mode, OBS only returns results of failed deletion in an XML response. OBS uses the **verbose** mode by default and you can specify the **quiet** mode in the request body.

For batch deletion, the request header must contain **Content-MD5** and **Content-Length**, so that the message body can be identified if network transmission error is detected at the server side.

Request Syntax

```
POST /?delete HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
Content-MD5: MD5
Content-Length: length

<?xml version="1.0" encoding="UTF-8"?>
<Delete>
   <Quiet>true</Quiet>
   <Object>
      <Key>Key</Key>
      <VersionId>VersionId</VersionId>
   </Object>
   <Object>
      <Key>Key</Key>
   </Object>
</Delete>
```

Request Parameters

This request involves no parameters.

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request uses elements to specify the list of objects to be deleted in a batch. **Table 5-70** describes the elements.

Table 5-70 Request Elements

Element	Description	Mandatory
Quiet	Specifies the **quiet** mode. With the **quiet** mode, OBS only returns the list of objects that failed to be deleted. This element is valid when set to **true**. Otherwise, OBS ignores it. Type: boolean	No
Delete	List of objects to be deleted Type: XML	Yes
Object	Names of objects to be deleted Type: XML	Yes
Key	Key of the object to be deleted Type: string	Yes
VersionId	Version ID of the object to be deleted Type: string	No

A maximum of 1000 objects can be deleted at a time. If you send a request for deleting more than 1000 objects, OBS returns an error message.

After concurrent tasks are assigned, OBS may encounter an internal error during cyclic deletion of multiple objects. In that case, the metadata still exists when the object index data is deleted, which means data inconsistency.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Type: application/xml
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DeleteResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
<Deleted>
   <Key>Key</Key>
</Deleted>
<Error>
   <Key>Key</Key>
   <Code>ErrorCode</Code>
   <Message>messagee</Message>
</Error>
</DeleteResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response uses elements to return results of deleted objects in a batch. **Table 5-71** describes the elements.

Table 5-71 Response elements

Element	Description
DeleteResult	Root node of batch deletion responses Type: container
Deleted	Container for results of successful deletion Type: container
Error	Container for results of failed deletion Type: container
Key	Object names in a deletion result Type: string
Code	Error code of a deletion failure Type: string
Message	Error message of a deletion failure Type: string
VersionId	Version IDs of objects to be deleted Type: string
DeleteMarker	If this element is specified, **true** will be returned when you create or delete a deletion marker in the requested bucket with versioning enabled. Type: boolean
DeleteMarkerVersionId	Indicates the version ID of the deletion marker deleted or created by the request. If the request either creates or deletes a deletion marker, OBS returns this element in response with the version ID of the deletion marker. This element will be returned in either of the following cases: ● You send a versionless request, that is, you specify only the object name but not the version ID. In this case, the UDS creates a deletion marker and returns its version ID in the response. ● You send a request with versions, that is, you specify object keys and version IDs that identify deletion markers. In this case, OBS deletes the deletion marker and returns its version ID in the response. Type: boolean

Error Responses

1. If the resolution result of an XML request contains more than 1000 objects, OBS returns **400 Bad Request**.

2. If the object key in an XML request is invalid (for example, containing more than 1024 characters), OBS returns **400 Bad Request**.

3. If the request header does not contain Content-MD5, OBS returns **400 Bad Request**.

For other errors, see **Table 7-3**.

Sample Request

```
POST /test333?delete HTTP/1.1
User-Agent: curl/7.29.0
Host: 127.0.0.1
Accept: */*
Date: WED, 01 Jul 2015 04:34:21 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:8sjZWJlWmYmYnK5JqXaFFQ+vHEg=
Content-MD5: ZPzz8L+hdRJ6qCqYbU/pCw==
Content-Length: 188

<?xml version="1.0" encoding="utf-8"?>
<Delete>
  <Quiet>true</Quiet>
  <Object>
    <Key>obja02</Key>
  </Object>
  <Object>
    <Key>obja02</Key>
  </Object>
</Delete>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3FE4CE80340D30B0542
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCRhY0FBWRm6qjOE1ACBZwS+0KYlPBq0f
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:34:21 GMT
Content-Length: 120

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<DeleteResult xmlns="http://obs.example.com/doc/2015-06-30/"/>
```

5.4.8 Configuring an Object ACL

Functions

OBS supports the control of access permission for objects. By default, only the object creator has the read and write permissions for the object. However, the creator can set a public access policy to assign the read permission to all other users.

You can set an access control policy when uploading an object or make a call of an API operation to modify or obtain the object ACL. An object ACL supports a maximum of 100 grants.

This section explains how to modify an object ACL and change access permission on an object.

Versioning

By default, this operation modifies the ACL of the latest version of an object. To specify a specified version, the request can carry the **versionId** parameter.

Request Syntax

```
PUT /ObjectName?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization

<AccessControlPolicy>
   <Owner>
      <ID>ID</ID>
   </Owner>
   <Delivered>true</Delivered>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>ID</ID>
         </Grantee>
         <Permission>permission</Permission>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

Request Parameters

Table 5-72 describes the request parameters.

Table 5-72 Request parameters

Parameter	Description	Mandatory
versionId	Object version ID. Object ACL of a specified version is to be changed. Type: string	No

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

The request message carries the ACL information of the object by using message elements. For the meanings of the elements, see **Table 5-73**.

Table 5-73 Request elements

Element	Description	Mandatory
Owner	Bucket owner information, including the ID Type: XML	Yes
ID	Domain ID of a user. Type: string	Yes
Grant	Container for the grantee and the granted permissions. A single object ACL can contain no more than 100 grants. Type: XML	No
Grantee	Container for the details about the grantee. Type: XML	No
Canned	Grants permissions to all users. Value range: Everyone Type: enumeration	No
Delivered	Indicates whether an object ACL inherits the ACL of a bucket. Type: boolean Default value: true	No
Permission	Authorized permission. Type: enumeration	No
AccessControlList	Indicates an ACL, which consists of three elements: **Grant**, **Grantee**, and **Permission**. Type: XML	Yes

Response Syntax

```
HTTP/1.1 status_code
Content-Length: length
Content-Type: application/xml
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-74**.

Table 5-74 Additional response header parameters

Header	Description
x-obs-version-id	Version number of the object whose ACL is to be modified. Type: string

Response Elements

This response involves no elements.

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
PUT /obj2?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:42:34 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:8xAODun1ofjkwHm8YhtN0QEcy9M=
Content-Length: 727

<AccessControlPolicy xmlns="http://obs.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <Delivered>false</Delivered>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <Canned>Everyone</Canned>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3F0FD2A03D2D30B0542
x-obs-id-2: 32AAAUgAIAABAAAQAAEAABAAAQAAEAABCTjCqTmsA1XRpIrmrJdvcEWvZyjbztdd
Date: WED, 01 Jul 2015 04:42:34 GMT
Content-Length: 0
```

5.4.9 Obtaining Object ACL Configuration

Functions

The implementation of this operation returns the ACL configuration of an object. You can perform this operation to view the ACL of an object, as long as you have the read permission for the object ACL.

Versioning

By default, this operation obtains the ACL of the latest version of an object. If the object has a delete marker, status code 404 is returned. To obtain the ACL of a specified version, the **versionId** parameter can be used to specify the desired version.

Request Syntax

```
GET /ObjectName?acl HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

The request parameter specifies the object ACL to be obtained. For details about the parameters, see **Table 5-75**.

Table 5-75 Request parameters

Parameter	Description	Mandatory
acl	Indicates that the request is to obtain the object ACL. Type: string	Yes
versionId	Version number of an object. Type: string	No

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length
Content-Type: application/xml
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AccessControlPolicy xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Owner>
      <ID>id</ID>
   </Owner>
   <Delivered>true</Delivered>
   <AccessControlList>
      <Grant>
         <Grantee>
            <ID>id</ID>
         </Grantee>
         <Permission>permission</Permission>
      </Grant>
   </AccessControlList>
</AccessControlPolicy>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-76**.

Table 5-76 Additional response header

Header	Description
x-obs-version-id	Version number of an object. Valid value: string There is no default value.

Response Elements

The response message of the request returns the ACL information of the object. **Table 5-77** describes the elements.

Table 5-77 Response elements

Element	Description
ID	User account ID Type: string
AccessControlList	List of users and their permissions for the bucket. Type: XML
Grant	Identifies the grantee and the permissions of the grantee. Type: XML
Grantee	Container for the details about the grantee. Type: XML

Element	Description
Delivered	Indicates whether an object ACL inherits the ACL of a bucket. Type: boolean
Permission	Permissions of a specified user for the bucket. Type: string

Error Responses

No special error responses are returned. For details about error responses, see **Table 7-3**.

Sample Request

```
GET /object011?acl HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:45:55 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:YcmvNQxItGjFeeC1K2HeUEp8MMM=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D3E650F3065C2295674C
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCS+wsHqRuA2Tx+mXUpNtBbWLPMle9CIx
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:45:55 GMT
Content-Length: 769

<?xml version="1.0" encoding="utf-8"?>
<AccessControlPolicy xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Owner>
    <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
  </Owner>
  <Delivered>false</Delivered>
  <AccessControlList>
    <Grant>
      <Grantee>
        <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee>
        <ID>783fc6652cf246c096ea836694f71855</ID>
      </Grantee>
      <Permission>READ</Permission>
      </Grant>
    <Grant>
      <Grantee>
        <Canned>Everyone</Canned>
      </Grantee>
      <Permission>READ_ACP</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

5.5 Operations on Multipart Upload

5.5.1 Listing Initialized Multipart Tasks in a Bucket

Functions

This operation queries all the multipart upload tasks that are initialized but have not been merged or canceled in a bucket.

Request Syntax

```
GET /?uploads&max-uploads=max HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: authorization
```

Request Parameters

This request uses parameters to specify the query range for multipart uploads. **Table 5-78** describes the parameters.

Table 5-78 Request parameters

Parameter	Description	Mandatory
delimiter	For a multipart upload that contains delimiters, the string between the first character and the first delimiter in the object name (excluding the prefix specified in the request, if any) are returned as **CommonPrefix**. Multipart uploads with objects that contain **CommonPrefix** are considered as a group and returned as one record. The record contains no information about the tasks, only informing the user that the group involves multipart uploads. Type: string	No
prefix	If a prefix is specified, the response only contains tasks whose names start with the prefix value. Type: string	No
max-uploads	Maximum number of multipart upload tasks returned. The value ranges from 1 to 1000. If the value has exceeded this range, 1000 tasks are returned by default. Type: integer	No

Parameter	Description	Mandatory
key-marker	Lists multipart uploads that follow the value of **key-marker**. Type: string	No
upload-id-marker	Lists multipart tasks that follow the value of **upload-id-marker** in **key-marker**. This parameter only functions together with **key-marker**. Type: string	No

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListMultipartUploadsResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Bucket>bucketname</Bucket>
   <KeyMarker/>
   <UploadIdMarker/>
   <NextKeyMarker>nextMarker</NextKeyMarker>
   <NextUploadIdMarker>idMarker</NextUploadIdMarker>
   <MaxUploads>maxUploads</MaxUploads>
   <IsTruncated>true</IsTruncated>
   <Upload>
      <Key>key</Key>
      <UploadId>uploadID</UploadId>
      <Initiator>
         <ID>domainID/domainID:userID/userID</ID>
      </Initiator>
      <Owner>
         <ID>ownerID</ID
      </Owner>

      <Initiated>initiatedDate</Initiated>
   </Upload>
</ListMultipartUploadsResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements of information about the multipart uploads. **Table 5-79** describes the elements.

Table 5-79 Response elements

Element	Description
ListMultipartUploadsResult	Container for responses of requests. Type: container Children: Bucket, KeyMarker, UploadIdMarker, NextKeyMarker, NextUploadIdMarker, MaxUploads, Delimiter, Prefix, Upload, CommonPrefixes, IsTruncated Ancestor: none
Bucket	Name of the bucket to which the multipart upload was initiated Type: string Ancestor: ListMultipartUploadsResult
KeyMarker	Object keys at or after which the multipart upload listing begins Type: string Ancestor: ListMultipartUploadsResult
UploadIdMarker	Upload ID after which the multipart upload listing begins Type: string Ancestor: ListMultipartUploadsResult
NextKeyMarker	Value of **KeyMarker** in a subsequent request after a multipart upload list is truncated Type: string Ancestor: **ListMultipartUploadsResult**
NextUploadIdMarker	Value of UploadMarker in a subsequent request after a multipart upload list is truncated Type: string Ancestor: ListMultipartUploadsResult
MaxUploads	Maximum of multipart uploads to be returned in the response Type: integer Ancestor: ListMultipartUploadsResult
IsTruncated	Indicates whether the returned list of multipart uploads is truncated. **true**: Not all results are returned. **false**: All results have been returned. Type: boolean Ancestor: ListMultipartUploadsResult

Element	Description
Upload	Container for elements related to a specific multipart upload Type: container Children: Key, UploadId, InitiatorOwner, Initiated Ancestor: ListMultipartUploadsResult
Key	Indicates the name of the object for which a multipart upload is initiated. Type: string Ancestor: Upload
UploadId	ID of the multipart upload Type: string Ancestor: Upload
Initiator	Container element that identifies who initiated the multipart upload ID of the children nodes. Type: container Ancestor: Upload
ID	ID of the account to which the owner belongs. Type: string Ancestor: Initiator, Owner
Owner	Owner of the part. Type: Container Indicates the subnode ID. Ancestor: **Upload**
Initiated	Date and time when the multipart upload was initiated Type: date Ancestor: Upload
ListMultipartUploadsResult.Prefix	Specified prefix in a request. Type: string Ancestor: ListMultipartUploadsResult
Delimiter	Delimiter in a request. Type: string Ancestor: ListMultipartUploadsResult

Element	Description
CommonPrefixes	Indicates group information. If you specify a delimiter in the request, the response contains group information in **CommonPrefixes**. Type: container Ancestor: ListMultipartUploadsResult
CommonPrefixes. Prefix	Indicates a different prefix in the group information in **CommonPrefixes**. Type: string Ancestor: **CommonPrefixes**

Error Responses

If the value of **maxUploads** is a non-integer or smaller than 0, OBS returns **400 Bad Request**.

For other errors, see **Table 7-3**.

Sample Request 1

List the initialized multipart tasks without any parameter.

```
GET /?uploads HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:51:21 GMT
Authorization: OBS UDSIAMSTUBTEST000008:XdmZgYQ+ZVy1rjxJ9/KpKq+wrU0=
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D405534D046A2295674C
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSDaHP+a+Bp0RI6Mm9XvCOrf7q3qvBQW
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:51:21 GMT
Content-Length: 681

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListMultipartUploadsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Bucket>examplebucket</Bucket>
  <KeyMarker/>
  <UploadIdMarker/>
  <Delimiter/>
  <Prefix/>
  <MaxUploads>1000</MaxUploads>
  <IsTruncated>false</IsTruncated>
  <Upload>
    <Key>obj2</Key>
    <UploadId>00000163D40171ED8DF4050919BD02B8</UploadId>
    <Initiator>
      <ID>domainID/b4bf1b36d9ca43d984fbcb9491b6fce9:userID/71f390117351534r88115ea2c26d1999</ID>
    </Initiator>
    <Owner>
      <ID>b4bf1b36d9ca43d984fbcb9491b6fce9</ID>
    </Owner>
```

```
  <Initiated>2015-07-01T02:30:54.582Z</Initiated>
 </Upload>
</ListMultipartUploadsResult>
```

Sample Request 2

List the initialized multipart tasks with the prefix and delimiter.

The following example describes how to list initialized multipart tasks when there are two multipart tasks in the bucket **examplebucket**, and their object names are **multipart-object001** and **part2-key02**. Set **prefix** to **multipart** and set **delimiter** to **object001**.

```
GET /?uploads&delimiter=object001&prefix=multipart HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 04:51:21 GMT
Authorization: OBS UDSIAMSTUBTEST000008:XdmZgYQ+ZVy1rjxJ9/KpKq+wrU0=
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 5DEB00000164A27A1610B8250790D703
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSq3ls2ZtLDD6pQLcJq1yGITXgspSvBR
Content-Type: application/xml
Date: WED, 01 Jul 2015 04:51:21 GMT
Content-Length: 681
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListMultipartUploadsResult xmlns="http://obs.example.com/doc/2015-06-30/">
 <Bucket>newbucket0001</Bucket>
 <KeyMarker></KeyMarker>
 <UploadIdMarker>
 </UploadIdMarker>
 <Delimiter>object</Delimiter>
 <Prefix>multipart</Prefix>
 <MaxUploads>1000</MaxUploads>
 <IsTruncated>false</IsTruncated>
 <CommonPrefixes>
  <Prefix>multipart-object001</Prefix>
 </CommonPrefixes>
</ListMultipartUploadsResult>
```

5.5.2 Initializing a Multipart Task

Functions

Before using this operation, make an API operation call to create a multipart upload task. The system will return a globally unique upload ID as the multipart upload identifier. This identifier can be used in subsequent requests including UploadPart, CompleteMultipartUpload, and ListParts. Create a multipart upload task does not affect the object that has the same name as object to be uploaded in multiple parts. You can create more than one multipart upload tasks for an object. This operation request can contain headers **x-obs-acl**, **x-obs-meta-***, **Content-Type**, and **Content-Encoding**. The headers are recorded in the multipart upload metadata.

Request Syntax

```
POST /ObjectName?uploads  HTTP/1.1
Host: bucketname.obs.region.example.com
```

```
Date: date
Authorization: authorization
```

Request Parameters

This request uses parameters to specify a multipart upload. **Table 5-80** describes the parameters.

Table 5-80 Request parameters

Parameter	Description	Mandatory
uploads	Indicates a multipart upload. Type: string	Yes

Request Headers

The request can use additional headers, as shown in **Table 5-81**.

Table 5-81 Request headers

Header	Description	Mandatory
x-obs-acl	When initializing a multipart upload task, you can add this message header to set the permission control policy for the object. The predefined common policies are as follows: **private**, **public-read**, and **public-read-write**. Type: string Note: This header is a predefined policy expressed in a character string. Example: **x-obs-acl: public-read-write**	No
x-obs-grant-read	When initializing a multipart upload task, you can use this header to authorize all users in an account to read the object and obtain the object metadata. Type: string Example: **x-obs-grant-read: ID=domainID** If multiple accounts are authorized, separate them with commas (,).	No
x-obs-grant-read-acp	When initializing a multipart upload task, you can use this header to authorize all users in an account the permission to obtain the object ACL. Type: string Example: **x-obs-grant-read-acp: ID=domainID** If multiple accounts are authorized, separate them with commas (,).	No

Header	Description	Mandatory
x-obs-grant-write-acp	When initializing a multipart upload task, you can use this header to authorize all users in an account the permission to write the object ACL. Type: string Example: **x-obs-grant-write-acp: ID=domainID** If multiple accounts are authorized, separate them with commas (,).	No
x-obs-grant-full-control	When initializing a multipart upload task, you can use this header to authorize all users in an account the permission to read the object, obtain the object metadata, obtain the object ACL, and write the object ACL. Type: string Example: **x-obs-grant-full-control: ID=domainID** If multiple accounts are authorized, separate them with commas (,).	No
x-obs-website-redirect-location	If a bucket is configured with the static website hosting function, it will redirect requests for this object to another object in the same bucket or to an external URL. OBS stores the value of this header in the object metadata. Type: string There is no default value. Constraint: The value must be prefixed by a slash (/), **http://**, or **https://**. The length of the value cannot exceed 2 KB.	No
x-obs-expires	Indicates the expiration time of an object, in days. An object will be automatically deleted once it expires (calculated from the last modification time of the object). Type: integer Example: x-obs-expires:3	No

For details about other common message headers, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
```

```
Content-Length: length
Connection: status

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Bucket>BucketName</Bucket>
   <Key>ObjectName</Key>
   <UploadId>uploadID</UploadId>
</InitiateMultipartUploadResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements to indicate the upload ID and the key (name) of the object (bucket) for which the multipart upload was initiated. The returned information is used in the subsequent operations. **Table 5-82** describes the elements.

Table 5-82 Response elements

Element	Description
InitiateMultipartU-ploadResult	Container of a multipart upload task. Type: XML
Bucket	Indicates the name of the bucket to which the multipart upload was initiated. Type: string
Key	Indicates the object key in a multipart upload. Type: string
UploadId	Indicates the ID for the initiated multipart upload. This ID is used for the subsequent operation. Type: string

Error Responses

1. If the AK or signature is invalid, OBS returns **403 Forbidden** and the error code is **AccessDenied**.

2. If the bucket does not exist, OBS returns **404 Not Found** and the error code is **NoSuchBucket**.

3. Check whether the user has the write permission for the specified bucket. If no, OBS returns **403 Forbidden** and the error code is **AccessDenied**.

Other errors are included in **Table 7-3**.

Sample Request 1

Initialize a multipart task.

```
POST /objectkey?uploads  HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 05:14:52 GMT
Authorization: OBS AKIAIOSFODNN7EXAMPLE:VGhpcyBtZXNzYWdlIHNpZ25lZGGieSRlbHZpbmc=
```

Sample Response 1

```
HTTP/1.1 200 OK
Server: OBS
x-obs-id-2: Weag1LuByRx9e6j5Onimru9pO4ZVKnJ2Qz7/C1NPcfTWAtRPfTaOFg==
x-obs-request-id: 996c76696e6727732072657175657374
Date: WED, 01 Jul 2015 05:14:52 GMT
Content-Length: 303

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Bucket>bucketname</Bucket>
  <Key>objectkey</Key>
  <UploadId>DCD2FC98B4F70000013DF578ACA318E7</UploadId>
</InitiateMultipartUploadResult>
```

Sample Request 2

The ACL is carried when the multipart task is initialized.

```
POST /objectkey?uploads  HTTP/1.1
Host: examplebucket.obs.region.example.com
Date: WED, 01 Jul 2015 05:15:43 GMT
x-obs-grant-write-acp:ID=52f24s3593as5730ea4f722483579ai7,ID=a93fcas852f24s3596ea8366794f7224
Authorization: OBS AKIAIOSFODNN7EXAMPLE:VGhpcyBtZXNzYWdlIHNpZ25lZGGieSRlbHZpbmc=
```

Sample Response 2

```
HTTP/1.1 200 OK
Server: OBS
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTnv+dtB51p+IVhAvWN7s5rSKhcWqDFs
x-obs-request-id: BB78000001648457112DF37FDFADD7AD
Date: WED, 01 Jul 2015 05:15:43 GMT
Content-Length: 303

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
  <Bucket>bucketname</Bucket>
  <Key>objectkey</Key>
  <UploadId>000001648453845DBB78F2340DD460D8</UploadId>
</InitiateMultipartUploadResult>
```

5.5.3 Multipart Upload

Functions

After initiating a multipart upload, you can use this operation to upload parts for the multipart upload using its task ID. When parts are uploaded in a multipart upload of an object, the upload sequence does not affect part merging, namely, multiple parts can be uploaded concurrently.

Part sizes range from 100 KB to 5 GB. However, when parts are being merged, the size of the last uploaded part ranges from 0 to 5 GB. The upload part ID ranges from 1 to 10,000.

NOTICE

The value of **partNumber** in a multipart task is unique. If you upload a part of the same **partNumber** repeatedly, the last part uploaded will overwrite the previous one. When multiple concurrent uploading of the same **partNumber** part of the same object is performed, the Last Write Win policy is applied. The time of **Last Write** is defined as the time when the metadata of the part is created. To ensure data accuracy, the client must be locked to ensure concurrent upload of the same part of the same object. Concurrent upload of different parts of the same object does not need to be locked.

Request Syntax

```
PUT /ObjectName?partNumber=partNum&uploadId=uploadID  HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Length: length
Authorization: authorization
Content-MD5:md5
<object Content>
```

Request Parameters

This request uses parameters to specify the upload task ID and part number. **Table 5-83** describes the parameters.

Table 5-83 Request parameters

Parameter	Description	Mandatory
partNumber	Indicates the ID of a part to be uploaded. The value is an integer from 1 to 10000. Type: integer	Yes
uploadId	Indicates a multipart upload ID. Type: string	Yes

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
ETag: etag
Content-Length: length
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

1. If a part number is not within the range from 1 to 10000, OBS returns **400 Bad Request**.
2. If a part size has exceeded 5 GB, the error code **400 Bad Request** is returned.
3. If the AK or signature is invalid, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
4. Check whether the bucket exists. If the bucket does not exist, OBS returns **404 Not Found** and the error code is **NoSuchBucket**.
5. View the bucket ACL to check whether the user has the read permission for the requested bucket. If the user does not have the read permission, OBS returns **403 AccessDenied**.
6. Check whether the multipart upload task exists. If the task does not exist, OBS returns **404 Not Found** and the error code is **NoSuchUpload**.
7. Check whether the request user is the initiator of the multipart upload task. If not, OBS returns **403 Forbidden** and the error code is **AccessDenied**.

Other errors are included in **Table 7-3**.

Sample Request

```
PUT /object02?partNumber=1&uploadId=00000163D40171ED8DF4050919BD02B8 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:15:55 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:ZB0hFwaHubi1aKHv7dSZjJts40g=
Content-Length: 102015348

[102015348 Byte part content]
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D40956A703289CA066F1
ETag: "b026324c6904b2a9cb4b88d6d61c81d1"
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCUQu/EOEVSMa04GXVwy0z9WI+BsDKvfh
Date: WED, 01 Jul 2015 05:15:55 GMT
Content-Length: 0
```

5.5.4 Uploading a Part of an Object - Copy

Functions

After creating a multipart upload job, you can specify its upload ID and upload a part to the job in OBS. Alternatively, you can make an API call to add a part (part of an object or the whole object).

NOTICE

You cannot determine whether a request is executed successfully only using **status_code** in the returned HTTP header. If 200 in **status_code** is returned, the server has received the request and starts to process the request. The body in the response shows whether the request is executed successfully. The request is executed successfully only when the body contains Etag; otherwise, the request fails to be executed.

Copy the source object and save it as **part1**. If a **part1** already exists before the copying, the original **part1** will be overwritten by the newly copied **part1**. After the copy is successful, only the latest **part1** is displayed. The old **part1** data will be deleted. Therefore, ensure that the target part does not exist or has no value when using the part copy operation. Otherwise, data may be deleted by mistake. The source object in the copy process does not change.

Request Syntax

```
PUT /ObjectName?partNumber=partNum&uploadId=UploadID HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
x-obs-copy-source: sourceobject
x-obs-copy-source-range:bytes=start-end
Authorization: authorization
Content-Length: length
```

Request Parameters

To copy a part, you need to specify the part number of the target part and the multipart upload task number. **Table 5-84** describes the parameters.

Table 5-84 Request parameters

Parameter	Description	Mandatory
partNumber	Indicates the ID of a part to be uploaded. Type: integer	Yes
uploadId	Indicates a multipart upload ID. Type: string	Yes

Request Headers

In addition the common message headers, the request uses two extended headers. **Table 3-3** describes the common message header.

Table 5-85 Request headers

Header	Description	Mandatory
x-obs-copy-source	Indicates the source object to be copied. Type: string	Yes
x-obs-copy-source-range	Indicates the range of bytes (start - end) to be copied from the source object. **start** indicates the start byte of the part to be copied and **end** indicates the end byte. Type: integer	No

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CopyPartResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <LastModified>modifiedDate</LastModified>
   <ETag>etag</ETag>
</CopyPartResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response contains elements of a part copy result. **Table 5-86** describes the elements.

Table 5-86 Response elements

Element	Description
LastModified	Indicates the latest time an object was modified. Type: string
ETag	ETag value of the target part. It is the unique identifier of the part content and is used to verify data consistency when merging parts. Type: string

Error Responses

1. If the AK or signature is invalid, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
2. Check whether the source bucket or destination bucket exists. If the source bucket or destination bucket does not exist, OBS returns **404 Not Found** and the error code is **NoSuchBucket**.
3. If the source object does not exist, OBS returns **404 Not Found** and the error code is **NoSuchKey**.
4. If the user does not have the read permission for the specified object, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
5. If the user does not have the write permission for the destination bucket, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
6. If the specified task does not exist, OBS returns **404 Not Found** and the error code is **NoSuchUpload**.
7. If the user is not the initiator of the multipart upload task, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
8. When the size of a copied part has exceeded 5 GB, OBS returns **400 Bad Request**.
9. If the part sequence number is not within the range from 1 to 10000, OBS returns error code **400 Bad Request**.

Other errors are included in **Table 7-3**.

Sample Request

```
PUT /tobject02?partNumber=2&uploadId=00000163D40171ED8DF4050919BD02B8 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:16:32 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:dSnpnNpawDSsLg/xXxaqFzrAmMw=
x-obs-copy-source: /destbucket/object01
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D40ABBD20405D30B0542
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTIJpD2efLy5o8sTTComwBb2He0j11Ne
Content-Type: application/xml
Date: WED, 01 Jul 2015 05:16:32 GMT
Transfer-Encoding: chunked

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CopyPartResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <LastModified>2015-07-01T05:16:32.344Z</LastModified>
  <ETag>"3b46eaf02d3b6b1206078bb86a7b7013"</ETag>
</CopyPartResult>
```

5.5.5 Listing Uploaded Parts of an Object

Functions

You can perform this operation to query all parts associated to a multipart upload. The size of each part listed by this API is the same as the size of the part uploaded.

Request Syntax

```
GET /ObjectName?uploadId=uploadid&max-parts=max&part-number-marker=marker HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: auth
```

Request Parameters

This request uses parameters to specify which parts in a multipart upload will be listed. **Table 5-87** describes the parameters.

Table 5-87 Request parameters

Parameter	Description	Mandatory
uploadId	Indicates a multipart upload ID. Type: string Default value: none	Yes
max-parts	Specifies the maximum number of parts to be listed. Type: string Default value: 1000.	No
part-number-marker	Indicates the part after which the part listing begins. OBS lists only parts with greater numbers than that specified by this parameter. Type: string Default value: none	No

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
Content-Length: length

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Bucket>BucketName</Bucket>
   <Key>object</Key>
   <UploadId>uploadid</UploadId>
   <Initiator>
      <ID>id</ID>
   </Initiator>
   <Owner>
```

```
    <ID>ownerid</ID>
  </Owner>
  <PartNumberMarker>partNmebermarker</PartNumberMarker>
  <NextPartNumberMarker>nextPartnumberMarker</NextPartNumberMarker>
  <MaxParts>maxParts</MaxParts>
  <IsTruncated>true</IsTruncated>
  <Part>
    <PartNumber>partNumber</PartNumber>
    <LastModified>modifiedDate</LastModified>
    <ETag>etag</ETag>
    <Size>size</Size>
  </Part>
</ListPartsResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response uses elements to return information about uploaded parts. **Table 5-88** describes the elements.

Table 5-88 Response elements

Element	Description
ListPartsResult	Indicates the container for responses to List Parts requests. Type: container Children: Bucket, Key, UploadId, PartNumberMarker, NextPartNumberMarker, MaxParts, IsTruncated, Part Ancestor: none
Bucket	Indicates a bucket name. Type: string Ancestor: ListPartsResult
Key	Indicates an object name. Type: string Ancestor: ListPartsResult
UploadId	Indicates the ID of a multipart upload. Type: string Ancestor: ListPartsResult
Initiator	Indicates the initiator of a multipart upload. Type: container Indicates the subnode ID. Ancestor: ListPartsResult

Element	Description
Owner	The value of this parameter is the same as that of **Initiator**. Type: container Children: ID Ancestor: ListPartsResult
ID	ID of the domain to which the owner belongs Type: string Ancestor: Initiator or Owner
PartNumberMarker	Part number after which listing parts begins. Type: integer Ancestor: ListPartsResult
NextPartNumberMarker	Indicates the value of **PartNumberMarker** in the next request when the returned result is incomplete. Type: integer Ancestor: ListPartsResult
MaxParts	Maximum number of parts returned in a response Type: integer Ancestor: ListPartsResult
IsTruncated	Indicates whether the returned part list is truncated. The value **true** indicates that the list was truncated and **false** indicates that the list was not truncated. Type: boolean Ancestor: **ListPartsResult**
Part	Indicates the container for elements related to a particular part. Type: string Children: PartNumber, LastModified, ETag, Size Ancestor: ListPartsResult **PartNumber** identifies a part.
PartNumber	Number of an uploaded part Type: integer Ancestor: ListPartsResult.Part
LastModified	Indicates the date and time a part was uploaded. Type: date Ancestor: ListPartsResult.Part

Element	Description
ETag	ETag value of the uploaded parts. It is the unique identifier of the part content and is used to verify data consistency during the combination of parts. Type: string Ancestor: ListPartsResult.Part
Size	Indicates the size of an uploaded part. Type: integer Ancestor: ListPartsResult.Part

Error Responses

1. If the AK or signature is invalid, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
2. If the requested bucket does not exist, OBS returns **404 Forbidden** and the error code is **NoSuchBucket**.
3. If the requested multipart upload task does not exist, OBS returns **404 Not Found** and the error code is **NoSuchUpload**.
4. OBS determines whether the use's domain ID has the read permission for the specified bucket. If the user does not have the permission, OBS returns **403 Forbidden** and the error code is **AccessDenied**.

For other errors, see **Table 7-3**.

Sample Request

```
GET /object02?uploadId=00000163D40171ED8DF4050919BD02B8 HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:20:35 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:xkABdSrBPrz5yqzuZdJnK5oL/yU=
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D40C099A04EF4DD1BDD9
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSK71fr+hDnzB0JBvQC1B9+S12AWxC41
Content-Type: application/xml
Date: WED, 01 Jul 2015 05:20:35 GMT
Content-Length: 888

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Bucket>test333</Bucket>
  <Key>obj2</Key>
  <UploadId>00000163D40171ED8DF4050919BD02B8</UploadId>
  <Initiator>
    <ID>domainID/domainiddomainiddomainiddo000008:userID/useriduseriduseriduseridus000008</ID>
  </Initiator>
  <Owner>
    <ID>domainiddomainiddomainiddo000008</ID>
  </Owner>
```

```
  <PartNumberMarker>0</PartNumberMarker>
  <NextPartNumberMarker>2</NextPartNumberMarker>
  <MaxParts>1000</MaxParts>
  <IsTruncated>false</IsTruncated>
  <Part>
    <PartNumber>1</PartNumber>
    <LastModified>2018-06-06T07:39:32.522Z</LastModified>
    <ETag>"b026324c6904b2a9cb4b88d6d61c81d1"</ETag>
    <Size>2058462721</Size>
  </Part>
  <Part>
    <PartNumber>2</PartNumber>
    <LastModified>2018-06-06T07:41:03.344Z</LastModified>
    <ETag>"3b46eaf02d3b6b1206078bb86a7b7013"</ETag>
    <Size>4572</Size>
  </Part>
</ListPartsResult>
```

5.5.6 Merging Parts into a Complete Object

Functions

After uploading all parts for a multipart upload, you can use this operation to complete the multipart upload. Before performing this operation, you cannot download the uploaded data. When merging parts, you need to copy the additional message header information recorded during the initialization of the multipart upload task to the object metadata. The processing process is the same as that of the common upload object with these message headers. In the case of merging parts concurrently, the Last Write Win policy must be followed but the time for initiating Last Write is specified as the time when a part multipart upload is initiated.

If a multipart upload has not been aborted, the uploaded parts occupy your storage quota. After all parts in the multipart upload are merged to an object, only the object occupies your storage quota. If a part uploaded in a multipart upload is not used in any merging parts multipart uploads, the part will be deleted to release storage quota.

You can send a request for downloading all or some data of the generated multipart by specifying a range.

You can send a request for deleting all parts uploaded in a multipart upload. Deleted data cannot be restored.

The Etag of an object with merged parts is not the real MD5 value of the object. The object ETag is calculated as follows: $MD5(M_1M_2...M_N)$-N, where M_n is the MD5 value of part n (N is the total number of parts). As described in the example below, there are three parts and each part has an MD5. The MD5 values of the three parts are combined and recalculated to obtain a new MD5 value. Then -N, as ETag of the combined part, is added to the right of the MD5 value. (N indicates the number of parts and is 3 in the example.)

If the response to an object merge request times out and error 500 or 503 is returned, you can first obtain the object metadata of the multipart upload task. Then, check whether the value of header **x-obs-uploadId** in the response is the same as the ID of this multipart upload task. If they are the same, object parts have been successfully merged on the server and you do not need to try again. For details, see **Consistency of Concurrent Operations**.

Versioning

If a bucket has versioning enabled, a unique version ID is generated for an object created from a multipart upload in this bucket and the version ID is returned in response header **x-obs-version-id**. If versioning is suspended for a bucket, the object version obtained after the merge is **null**. For details about the versioning statuses of a bucket, see **Configuring Versioning for a Bucket**.

NOTICE

If 10 parts are uploaded but only nine parts are selected for merge, the parts that are not merged will be automatically deleted by the system. The parts that are not merged cannot be restored after being deleted. Before combining the parts, adopt the interface used to list the parts that have been uploaded to check all parts to ensure that no part is missed.

Request Syntax

```
POST /ObjectName?uploadId=uploadID HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Content-Length: length
Authorization: authorization
<CompleteMultipartUpload>
   <Part>
      <PartNumber>partNum</PartNumber>
      <ETag>etag</ETag>
   </Part>
   <Part>
      <PartNumber>partNum</PartNumber>
      <ETag>etag</ETag>
   </Part>
   <Part>
      <PartNumber>partNum</PartNumber>
      <ETag>etag</ETag>
   </Part>
</CompleteMultipartUpload>
```

Request Parameters

This request uses parameters to specify the ID of a multipart upload whose parts will be merged. **Table 5-89** describes the parameters.

Table 5-89 Request parameters

Parameter	Description	Mandatory
uploadId	Indicates a multipart upload. Type: string	Yes

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request uses elements to specify the list of parts to be merged. **Table 5-90** describes the elements.

Table 5-90 Request Elements

Element	Description	Mandatory
CompleteMultipartUpload	List of parts to be combined Type: XML	Yes
PartNumber	Part number Type: integer	Yes
ETag	ETag value returned upon successful upload of a part. It is the unique identifier of the part content. This parameter is used to verify data consistency when parts are merged. Type: string	Yes

Response Syntax

```
HTTP/1.1 status_code
Date: date
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUploadResult xmlns="http://obs.region.example.com/doc/2015-06-30/">
   <Location>http://example-Bucket.obs.region.example.com/example-Object</Location>
   <Bucket>bucketname</Bucket>
   <Key>ObjectName</Key>
   <ETag>ETag</ETag>
</CompleteMultipartUploadResult>
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

In addition to the common response headers, the following message headers may also be used. For details, see **Table 5-91**.

Table 5-91 Additional response header parameters

Header	Description
x-obs-version-id	Version of the object after parts being merged. Type: string

Response Elements

This response uses elements to return the result of merging parts. **Table 5-92** describes the elements.

Table 5-92 Response elements

Element	Description
Location	URL of the object after parts being merged. Type: string
Bucket	Bucket in which parts are combined Type: string
Key	Indicates the key of the generated object. Type: string
ETag	The result calculated based on the ETag of each part is the unique identifier of the object content. Type: string

Error Responses

1. If no message body exists, OBS returns **400 Bad Request**.
2. If the message body format is incorrect, OBS returns **400 Bad Request**.
3. If the part information in the message body is not sorted by part sequence number, OBS returns **400 Bad Request** and the error code is **InvalidPartOrder**.
4. If the AK or signature is invalid, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
5. If the requested bucket does not exist, OBS returns **404 Not Found** and the error code is **NoSuchBucket**.
6. If the requested multipart upload does not exist, OBS returns **404 Not Found** and error code **NoSuchUpload**.
7. If the user is not the initiator of the task, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
8. If the request part list contains a part that does not exist, OBS returns **400 Bad Request** and the error code is **InvalidPart**.
9. If the Etag of the part contained in the request part list is incorrect, OBS returns **400 Bad Request** and the error code is **InvalidPart**.
10. If the size of a part other than the last part is smaller than 100 KB, OBS returns **400 Bad Request**.
11. If the size of the object is greater than 48.8 TB after parts being merged, OBS returns status code **400 Bad Request**.

Other errors are included in **Table 7-3**.

Sample Request

```
POST /object02?uploadId=00000163D46218698DF407362295674C HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:23:46 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:dOfK9iILcKxo58tRp3fWeDoYzKA=
```

```
Content-Length: 422

<?xml version="1.0" encoding="utf-8"?>
<CompleteMultipartUpload>
  <Part>
    <PartNumber>1</PartNumber>
    <ETag>a54357aff0632cce46d942af68356b38</ETag>
  </Part>
  <Part>
    <PartNumber>2</PartNumber>
    <ETag>0c78aef83f66abc1fa1e8477f296d394</ETag>
  </Part>
  <Part>
    <PartNumber>3</PartNumber>
    <ETag>acbd18db4cc2f85cedef654fccc4a4d8</ETag>
  </Part>
</CompleteMultipartUpload>
```

Sample Response

```
HTTP/1.1 200 OK
Server: OBS
x-obs-request-id: 8DF400000163D4625BE3075019BD02B8
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCSN8D1AfQcIvyGBZ9+Ee+jU6zv1iYdO4
Content-Type: application/xml
Date: WED, 01 Jul 2015 05:23:46 GMT
Content-Length: 326

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUploadResult xmlns="http://obs.example.com/doc/2015-06-30/">
  <Location>/examplebucket/object02</Location>
  <Bucket>examplebucket</Bucket>
  <Key>object02</Key>
  <ETag>"03f814825e5a691489b947a2e120b2d3-3"</ETag>
</CompleteMultipartUploadResult>
```

5.5.7 Canceling a Multipart Upload Task

Functions

You can perform this operation to abort a multipart upload. You cannot upload or list parts after operations to merge parts or abort a multipart upload are performed.

Request Syntax

```
DELETE /ObjectName?uploadId=uplaodID HTTP/1.1
Host: bucketname.obs.region.example.com
Date: date
Authorization: auth
```

Request Parameters

This request uses message parameters to specify the multipart upload task number of the segment task. **Table 5-93** describes the parameters.

Table 5-93 Request parameters

Parameter	Description	Mandatory
uploadId	Indicates a multipart upload. Type: string	Yes

Request Headers

This request uses common headers. For details, see **Table 3-3**.

Request Elements

This request involves no elements.

Response Syntax

```
HTTP/1.1 status_code
Date: date
```

Response Headers

The response to the request uses common headers. For details, see **Table 3-20**.

Response Elements

This response involves no elements.

Error Responses

1. If the AK or signature is invalid, OBS returns **403 Forbidden** and the error code is **AccessDenied**.
2. If the requested bucket does not exist, OBS returns **404 Not Found** and the error code is **NoSuchBucket**.
3. If you are neither the initiator of a multipart upload nor the bucket owner, OBS returns **403 Forbidden**.
4. If the operation is successful, OBS returns **204 No Content** to the user.

Other errors are included in **Table 7-3**.

Sample Request

```
DELETE /object02?uploadId=00000163D46218698DF407362295674C HTTP/1.1
User-Agent: curl/7.29.0
Host: examplebucket.obs.region.example.com
Accept: */*
Date: WED, 01 Jul 2015 05:28:27 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:QmM2d1DBXZ/b8drqtEv1QJHPbM0=
```

Sample Response

```
HTTP/1.1 204 No Content
Server: OBS
x-obs-request-id: 8DF400000163D463E02A07EC2295674C
```

```
x-obs-id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTp5YDlzn0UgqG3laRfkHLGyz7RpR9ON
Date: WED, 01 Jul 2015 05:28:27 GMT
```

6 IAM Policies and Supported Actions

6.1 Introduction

This chapter describes fine-grained permissions management for your OBS using Identity and Access Management (IAM). If your account does not require individual IAM users, skip this chapter.

By default, new IAM users do not have any permissions assigned. You need to add a user to one or more groups, and attach IAM policies to these groups. The user then inherits permissions from the groups it is a member of. This process is called authorization. After authorization, the user can perform specified operations on OBS based on the IAM policies.

For details about user policies related to OBS, see the topic of "User Permissions" in the "Service Overview" section of *OBS User Guide*. For details about the syntax structure and examples of IAM policies, see the topic of "IAM Policy" in the section "Permission Control" > "Permission Control Mechanisms" in the "Console Operation Guide" of *OBS User Guide*.

There are fine-grained policies and role-based access control (RBAC) policies. An RBAC policy consists of permissions for an entire service. Users in a group with such a policy assigned are granted all of the permissions required for that service. A fine-grained policy consists of API-based permissions for operations on specific resource types. Fine-grained policies, as the name suggests, allow for more fine-grained control than RBAC policies.

NOTE

- If you want to allow or deny the access to an API, fine-grained authorization is a good choice.
- Because of the cache, it takes about 13 minutes for the RBAC policy to take effect after being granted to users and user groups. After a fine-grained OBS policy is granted, it takes about 5 minutes for the policy to take effect.

An account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. The permissions required for calling an API are determined by the actions supported by the API. Only users who have been granted permissions allowing the actions can call the API successfully.

For example, if an IAM user needs to create buckets using an API, the user must have been granted permissions that allow the **obs:bucket:CreateBucket** action.

Supported Actions

Operations supported by a fine-grained policy are specific to APIs. The following describes the headers of the actions provided in this chapter:

- Permissions: defined by actions in a custom policy.
- APIs: REST APIs that can be called by a custom policy.
- Actions: added to a custom policy to control permissions for specific operations.

OBS supports the following actions that can be defined in a custom policy:

- **Bucket-related actions** include actions supported by all OBS bucket-related APIs, such as the APIs for listing all buckets, creating and deleting buckets, setting bucket policies, setting bucket event notification, and setting bucket logging.
- **Object-related actions** include APIs for uploading, downloading, and deleting objects.

6.2 Bucket-Related Actions

Table 6-1 Bucket-related actions

Permission	API	Action	IAM Project
Listing all buckets	**Listing Buckets**	obs:bucket:ListAllMyBuckets	√
Creating a bucket	**Creating a Bucket**	obs:bucket:CreateBucket	√
Listing objects in a bucket	**Listing Objects in a Bucket**	obs:bucket:ListBucket	√
Listing object versions in a bucket	**Listing Objects in a Bucket**	obs:bucket:ListBucketVersions	√
Checking whether a bucket exists and obtaining the metadata of the bucket	**Obtaining Bucket Metadata**	obs:bucket:HeadBucket	√
Obtaining the bucket location	**Obtaining Bucket Location**	obs:bucket:GetBucketLocation	√
Deleting a bucket	**Deleting Buckets**	obs:bucket:DeleteBucket	√

Permission	API	Action	IAM Project
Configuring a bucket policy	**Configuring a Bucket Policy**	obs:bucket:PutBucketPolicy	√
Obtaining the bucket policy information	**Obtaining Bucket Policy Information**	obs:bucket:GetBucketPolicy	√
Deleting a bucket policy	**Deleting a Bucket Policy**	obs:bucket:DeleteBucketPolicy	√
Configuring a bucket ACL	**Configuring a Bucket ACL**	obs:bucket:PutBucketAcl	√
Obtaining bucket ACL information	**Obtaining Bucket ACL Information**	obs:bucket:GetBucketAcl	√
Configuring logging for a bucket	**Configuring Logging for a Bucket**	obs:bucket:PutBucketLogging	√
Obtaining a bucket logging configuration	**Obtaining a Bucket Logging Configuration**	obs:bucket:GetBucketLogging	√
Configuring and deleting the lifecycle rule for a bucket	**Configuring Bucket Lifecycle Rules** **Deleting Lifecycle Rules**	obs:bucket:PutLifecycleConfiguration	√
Obtaining the lifecycle rule of a bucket	**Obtaining Bucket Lifecycle Configuration**	obs:bucket:GetLifecycleConfiguration	√
Configuring versioning	**Configuring Versioning for a Bucket**	obs:bucket:PutBucketVersioning	√
Obtaining the versioning information of a bucket	**Obtaining Bucket Versioning Status**	obs:bucket:GetBucketVersioning	√
Setting event notification for a bucket	**Configuring Event Notification for a Bucket**	obs:bucket:PutBucketNotification	√
Obtaining the event notification configuration of a bucket	**Obtaining the Event Notification Configuration of a Bucket**	obs:bucket:GetBucketNotification	√
Configuring tags for a bucket	**Configuring Tags for a Bucket**	obs:bucket:PutBucketTagging	√

Permission	API	Action	IAM Project
Obtaining bucket tags	**Obtaining Bucket Tags**	obs:bucket:GetBucketTagging	√
Deleting tags	**Deleting Tags**	obs:bucket:DeleteBucketTagging	√
Configuring bucket storage quota	**Configuring Bucket Storage Quota**	obs:bucket:PutBucketQuota	√
Querying bucket storage quota	**Querying Bucket Storage Quota**	obs:bucket:GetBucketQuota	√
Querying information about used space in a bucket	**Querying Information About Used Space in a Bucket**	obs:bucket:GetBucketStorage	√
Configuring bucket inventories	**Configuring Bucket Inventories**	obs:bucket:PutBucketInventoryConfiguration	√
Obtaining and listing bucket inventories	**Obtaining Bucket Inventories** **Listing Bucket Inventories**	obs:bucket:GetBucketInventoryConfiguration	√
Deleting bucket inventories	**Deleting Bucket Inventories**	obs:bucket:DeleteBucketInventoryConfiguration	√
Configuring the static website hosting for a bucket	**Configuring Static Website Hosting for a Bucket**	obs:bucket:PutBucketWebsite	√
Obtaining the static website configuration information of a bucket	**Obtaining the Static Website Hosting Configuration of a Bucket**	obs:bucket:GetBucketWebsite	√
Deleting the static website hosting configuration of a bucket	**Deleting the Static Website Hosting Configuration of a Bucket**	obs:bucket:DeleteBucketWebsite	√
Configuring bucket CORS and deleting the CORS configuration	**Configuring Bucket CORS** **Deleting the CORS Configuration of a Bucket**	obs:bucket:PutBucketCORS	√

Permission	API	Action	IAM Project
Obtaining the CORS configuration of a bucket	Obtaining the CORS Configuration of a Bucket	obs:bucket:GetBucketCORS	√

6.3 Object-Related Actions

Table 6-2 Object-related actions

Permission	API	Action	IAM Project
Uploading objects using PUT method, uploading objects using POST method, copying objects, appending an object, initializing a multipart task, uploading parts, uploading a part, and merging parts	Uploading Objects - PUT Uploading Objects - POST Copying Objects Initializing a Multipart Task Multipart Upload Merging Parts into a Complete Object	obs:object:PutObject	√
Obtaining object content and metadata	Downloading Objects Querying Object Metadata	obs:object:GetObject	√
Obtaining the content and metadata of a specified object version	Downloading Objects Querying Object Metadata	obs:object:GetObjectVersion	√
Deleting an object or objects	Deleting an Object Deleting Objects	obs:object:DeleteObject	√
Deleting an object version or object versions	Deleting an Object Deleting Objects	obs:object:DeleteObjectVersion	√
Configuring object ACL	Configuring an Object ACL	obs:object:PutObjectAcl	√
Configuring the ACL for an object of a specified version	Configuring an Object ACL	obs:object:PutObjectVersionAcl	√

Permission	API	Action	IAM Project
Obtaining the object ACL information	**Obtaining Object ACL Configuration**	obs:object:GetObjectAcl	√
Obtaining the ACL for an object of a specified version	**Obtaining Object ACL Configuration**	obs:object:GetObjectVersionAcl	√
Listing uploaded parts	**Listing Uploaded Parts of an Object**	obs:object:ListMultipartUploadParts	√
Canceling a multipart upload task	**Canceling a Multipart Upload Task**	obs:object:AbortMultipartUpload	√

7 Appendixes

7.1 Status Codes

Table 7-1 lists the status codes and prompt message returned by the server to the user.

Table 7-1 Status codes

Status Code	Description
2xx	Indicates that the server has successfully returned the requested data.
4xx	Indicates that the request sent from the client is incorrect, so the server does not create or modify data.
5xx	Indicates that an error occurs on the server, and the user does not know whether the request has been successfully sent.

7.2 Error Codes

If an API calling fails, no result data is returned. You can locate the cause of the error according to the error code of each API. If an API calling fails, HTTP status code 3*xx*, 4*xx* or 5*xx* is returned. The response body contains the specific error code and information.

Error Response Syntax

When an error occurs, the response header information contains:

- Content-Type: application/xml
- HTTP error status code 3*xx*, 4*xx*, or 5*xx*

The response body also contains information about the error. The following is an error response example that shows common elements in the Representational State Transfer (REST) error response body.

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
<Code>NoSuchKey</Code>
<Message>The resource you requested does not exist</Message>
<Resource>/example-bucket/object</Resource>
<RequestId>001B21A61C6C0000013402C4616D5285</RequestId>
<HostId>RkRCRDJENDc5MzdGQkQ4OUY3MTI4NTQ3NDk2Mjg0M0FB
QUFBQUFBYmJiYmJiYmJD</HostId>
</Error>
```

Table 7-2 describes the meaning of each element.

Table 7-2 Error response elements

Element	Description
Error	Root element that describes the error in an XML response body
Code	HTTP return code that corresponds to the error in the XML response body. For details about error codes, see Table 7-3.
Message	Details the error in the XML error response body. For details about error messages, see Table 7-3.
RequestId	ID of the request whose error response is returned. The ID is used for locating the error.
HostId	ID of the server that returns an error response
Resource	Bucket or object related to an error.

NOTE

Some error responses contain more detailed information. It is recommended that all error information be logged for easier rectification of errors.

Description

If OBS encounters an error when processing a request, a response containing the error code and description is returned. Table 7-3 describes the error codes of OBS.

Table 7-3 Error codes

Status Code	Error Code	Error Message	Solution
301 Moved Permanently	PermanentRedirect	The requested bucket can be accessed only through the specified address. Send subsequent requests to the address.	Send the request to the returned redirection address.

Status Code	Error Code	Error Message	Solution
301 Moved Permanently	WebsiteRedirect	The website request lacks **bucketName**.	Put the bucket name in the request and try again.
307 Moved Temporarily	TemporaryRedirect	Temporary redirection. If the DNS is updated, the request is redirected to the bucket.	The system automatically redirects the request or sends the request to the redirection address.
400 Bad Request	BadDigest	The specified value of **Content-MD5** does not match the value received by OBS.	Check whether the MD5 value carried in the header is the same as that calculated by the message body.
400 Bad Request	BadDomainName	The domain name is invalid.	Use a valid domain name.
400 Bad Request	BadRequest	Invalid request parameters.	Modify the parameters according to the error details in the message body.
400 Bad Request	CustomDomainAreadyExist	The configured domain already exists.	It has been configured and does not need to be configured again.
400 Bad Request	CustomDomainNotExist	Delete the domain that does not exist.	It is not configured or has been deleted. You do not need to delete it.
400 Bad Request	EntityTooLarge	The size of the object uploaded using the POST method exceeds the upper limit.	Modify the conditions specified in the policy when posting the object or reduce the object size.
400 Bad Request	EntityTooSmall	The size of the object uploaded using the POST method does not reach the lower limit.	Modify the conditions specified in the policy when posting the object or increase the object size.

Status Code	Error Code	Error Message	Solution
400 Bad Request	IllegalLocationConstraintException	A request without **Location** is sent for creating a bucket in a non-default region.	Send the bucket creation request to the default region, or send the request with the **Location** of the non-default region.
400 Bad Request	IncompleteBody	No complete request body is received due to network or other problems.	Upload the object again.
400 Bad Request	IncorrectNumberOfFilesInPostRequest	Each POST request must contain one file to be uploaded.	Carry a file to be uploaded.
400 Bad Request	InvalidArgument	Invalid parameter.	Modify the parameter according to the error details in the message body.
400 Bad Request	InvalidBucket	The bucket to be accessed does not exist.	Change the bucket name.
400 Bad Request	InvalidBucketName	The bucket name specified in the request is invalid, which may have exceeded the maximum length, or contain special characters that are not allowed.	Change the bucket name.
400 Bad Request	InvalidEncryptionAlgorithmError	Incorrect encryption algorithm. The object cannot be decrypted due to incorrect encryption header carried when downloading the SSE-C encrypted object.	Carry the correct encryption header when downloading the object.
400 Bad Request	InvalidLocationConstraint	The specified **Location** in the bucket creation request is invalid or does not exist.	Correct the **Location** in the bucket creation request.

Status Code	Error Code	Error Message	Solution
400 Bad Request	InvalidPart	One or more specified parts are not found. The parts may not be uploaded or the specified entity tags (ETags) do not match the parts' ETags.	Merge the parts correctly according to the ETags.
400 Bad Request	InvalidPartOrder	Parts are not listed in ascending order by part number.	Sort the parts in ascending order and merge them again.
400 Bad Request	InvalidPolicyDocument	The content of the form does not meet the conditions specified in the policy document.	Modify the policy in the constructed form according to the error details in the message body and try again.
400 Bad Request	InvalidRedirectLocation	Invalid redirect location.	Specifies the correct IP address.
400 Bad Request	InvalidRequest	Invalid request.	Modify the parameter according to the error details in the message body.
400 Bad Request	InvalidRequestBody	The request body is invalid. The request requires a message body but no message body is uploaded.	Upload the message body in the correct format.
400 Bad Request	InvalidTargetBucketForLogging	The delivery group has no ACL permission for the target bucket.	Configure the target bucket ACL and try again.
400 Bad Request	KeyTooLongError	The provided key is too long.	Use a shorter key.
400 Bad Request	MalformedACLError	The provided XML file is in an incorrect format or does not meet format requirements.	Use the correct XML format to retry.
400 Bad Request	MalformedError	The XML format in the request is incorrect.	Use the correct XML format to retry.
400 Bad Request	MalformedLoggingStatus	The XML format of **Logging** is incorrect.	Use the correct XML format to retry.

Status Code	Error Code	Error Message	Solution
400 Bad Request	MalformedPolicy	The bucket policy does not pass.	Modify the bucket policy according to the error details returned in the message body.
400 Bad Request	MalformedQuotaError	The Quota XML format is incorrect.	Use the correct XML format to retry.
400 Bad Request	MalformedXML	An XML file of a configuration item is in incorrect format.	Use the correct XML format to retry.
400 Bad Request	MaxMessageLengthExceeded	Copying an object does not require a message body in the request.	Remove the message body and retry.
400 Bad Request	MetadataTooLarge	The size of the metadata header has exceeded the upper limit.	Reduce the size of the metadata header.
400 Bad Request	MissingRegion	No region contained in the request and no default region defined in the system.	Carry the region information in the request.
400 Bad Request	MissingRequestBodyError	This error code is returned after you send an empty XML file.	Provide the correct XML file.
400 Bad Request	MissingRequiredHeader	Required headers are missing in the request.	Provide required headers.
400 Bad Request	MissingSecurityHeader	A required header is not provided.	Provide required headers.
400 Bad Request	TooManyBuckets	You have attempted to create more buckets than allowed.	Delete some buckets and try again.
400 Bad Request	TooManyCustomDomains	Too many user accounts are configured.	Delete some user accounts and try again.
400 Bad Request	TooManyWrongSignature	The request is rejected due to high-frequency errors.	Replace the Access Key and try again.

Status Code	Error Code	Error Message	Solution
400 Bad Request	UnexpectedContent	The request requires a message body which is not carried by the client, or the request does not require a message body but the client carries the message body.	Try again according to the instruction.
400 Bad Request	UserKeyMustBeSpecified	This operation is available only to specific users.	Contact technical support.
403 Forbidden	AccessDenied	Access denied, because the request does not carry a date header or the header format is incorrect.	Provide a correct date header in the request.
403 Forbidden	AccessForbidden	Insufficient permission. No CORS configuration exists for the bucket or the CORS rule does not match.	Modify the CORS configuration of the bucket or send the matched OPTIONS request based on the CORS configuration of the bucket.
403 Forbidden	AllAccessDisabled	You have no permission to perform the operation. The bucket name is forbidden.	Change the bucket name.
403 Forbidden	InsufficientStorageSpace	Insufficient storage space.	If the quota is exceeded, increase quota or delete some objects.
403 Forbidden	InvalidAccessKeyId	The access key ID provided by the customer does not exist in the system.	Provide correct access key Id.

Status Code	Error Code	Error Message	Solution
403 Forbidden	RequestTimeTooSkewed	There was a large time offset between the OBS server time and the time when the client initiated a request. For security purposes, OBS verifies the time offset between the client and server. If the offset is longer than 15 minutes, the OBS server will reject your requests and this error message is reported.	Check whether there is a large time offset between the client time and server time. If there is, adjust the client time based on your local time (UTC) and try again.
403 Forbidden	SignatureDoesNotMatch	The provided signature does not match the signature calculated by OBS.	Check your secret access key and signature algorithm.
403 Forbidden	VirtualHostDomainRequired	Virtual hosting access domain name is not used.	Use the virtual hosting access domain name. For details, see **Constructing a Request**.
403 Forbidden	Unauthorized	The user has not been authenticated in real name.	Authenticate the user's real name and try again.
404 Not Found	NoSuchBucket	The specified bucket does not exist.	Create a bucket and perform the operation again.
404 Not Found	NoSuchBucketPolicy	No bucket policy exists.	Configure a bucket policy.
404 Not Found	NoSuchCORSConfiguration	No CORS configuration exists.	Configure CORS first.
404 Not Found	NoSuchCustomDomain	The requested user account does not exist.	Set a user account first.
404 Not Found	NoSuchKey	The specified key does not exist.	Upload the object first.
404 Not Found	NoSuchLifecycleConfiguration	The requested lifecycle rule does not exist.	Configure a lifecycle rule first.

Status Code	Error Code	Error Message	Solution
404 Not Found	NoSuchUpload	The specified multipart upload does not exist. The upload ID does not exist or the multipart upload has been terminated or completed.	Use the existing part or reinitialize the part.
404 Not Found	NoSuchVersion	The specified version ID does not match any existing version.	Use a correct version ID.
404 Not Found	NoSuchWebsiteConfiguration	The requested website does not exist.	Configure the website first.
405 Method Not Allowed	MethodNotAllowed	The specified method is not allowed against the requested resource. The message "Specified method is not supported." is returned.	The method is not allowed.
408 Request Timeout	RequestTimeout	The socket connection to the server has no read or write operations within the timeout period.	Check the network and try again, or contact technical support.
409 Conflict	BucketAlreadyExists	The requested bucket name already exists. The bucket namespace is shared by all users of OBS. Select another name and retry.	Change the bucket name.
409 Conflict	BucketAlreadyOwnedByYou	Your previous request for creating the namesake bucket succeeded and you already own it.	No more buckets need to be created.
409 Conflict	BucketNotEmpty	The bucket that you tried to delete is not empty.	Delete the objects in the bucket and then delete the bucket.
409 Conflict	InvalidBucketState	Invalid bucket status. After cross-region replication is configured, bucket versioning cannot be disabled.	Enable bucket versioning or cancel cross-region replication.

Status Code	Error Code	Error Message	Solution
409 Conflict	OperationAborted	A conflicting operation is being performed on this resource. Retry later.	Try again later.
409 Conflict	ServiceNotSupported	The request method is not supported by the server.	Not supported by the server. Contact technical support.
411 Length Required	MissingContentLength	The HTTP header Content-Length is not provided.	Provide the Content-Length header.
412 Precondition Failed	PreconditionFailed	At least one of the specified preconditions is not met.	Modify according to the condition prompt in the returned message body.
416 Client Requested Range Not Satisfiable	InvalidRange	The requested range cannot be obtained.	Retry with the correct range.
500 Internal Server Error	InternalError	An internal error occurs. Retry later.	Contact technical support.
501 Not Implemented	ServiceNotImplemented	The request method is not implemented by the server.	Not supported currently. Contact technical support.
503 Service Unavailable	ServiceUnavailable	The server is overloaded or has internal errors.	Try later or contact technical support.
503 Service Unavailable	SlowDown	Too frequent requests. Reduce your request frequency.	Too frequent requests. Reduce your request frequency.

7.3 Obtaining Access Keys (AK/SK)

When calling an API, you need to use the AK/SK to verify the signature. To obtain the AK/SK, perform the following steps:

Step 1 Log in to the console.

Step 2 Hover the cursor on the username in the upper right corner and select **My Credentials** from the drop-down list.

Step 3 On the **My Credentials** page, click **Manage Access Keys**.

Step 4 Click **Add Access Key**. The **Add Access Key** dialog box is displayed.

Step 5 Enter the password for login.

Step 6 Enter the authentication code received in your email or mobile phone.

NOTE

For users created through IAM, if no email address or mobile number is specified during user creation, only the login password needs to be authenticated.

Step 7 Click **OK** to download the access key file.

NOTE

Keep the AK/SK file properly to prevent information leakage.

----End

7.4 Obtaining the Domain ID and User ID

When an API is called, the domain ID (**DomainID**) and user ID (**UserID**) need to be specified in some requests. Therefore, you need to obtain them from the console. The procedure is as follows:

Step 1 Log in to the console.

Step 2 Click the username and select **My Credentials** from the drop-down list.

On the **My Credentials** page, view the domain ID and user ID.

----End

7.5 Consistency of Concurrent Operations

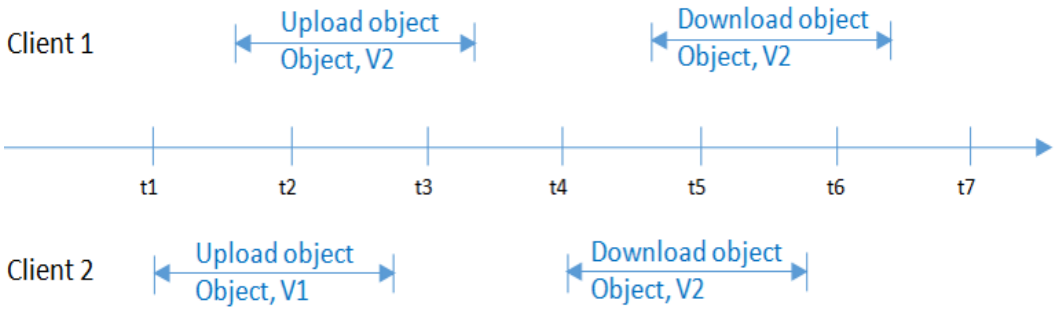
After a success message is returned in response to a client's write or deletion request, the client can obtain the latest data. If a client that initiates a write request times out in waiting for a response, or the server returns HTTP response status code **500** or **503**, the subsequent read operations may fail. If such an error occurs, query whether the data has been successfully uploaded to the server. If not, upload the data again.

If a client simultaneously uploads, queries, or deletes the same object or bucket, these operations may reach the system at different times and have different latency periods, so different results may return. For example, if multiple clients simultaneously upload the same object, the latest upload request received by the system will replace the previous one. If you want to prevent an object from being simultaneously accessed, you must add a lock mechanism for the object in upper-layer applications.

Example of Concurrent Operations

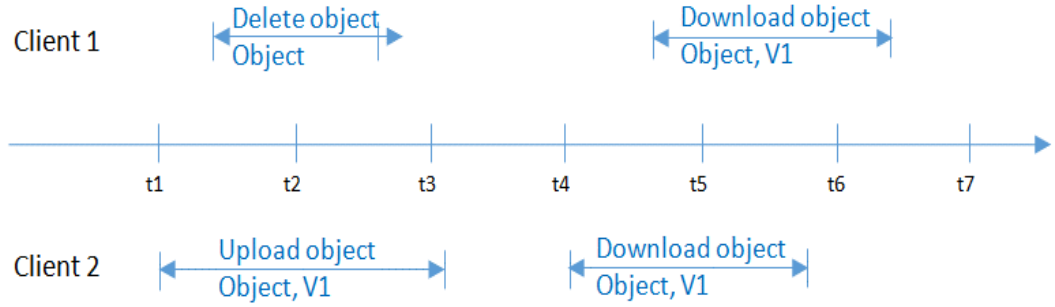
1. When client1 is uploading an object V1, client2 is uploading an object V2 with the same name. After the successful uploads, both client1 and client2 can access the latest object data V2, as shown in **Figure 7-1**.

Figure 7-1 Concurrent upload of the same object



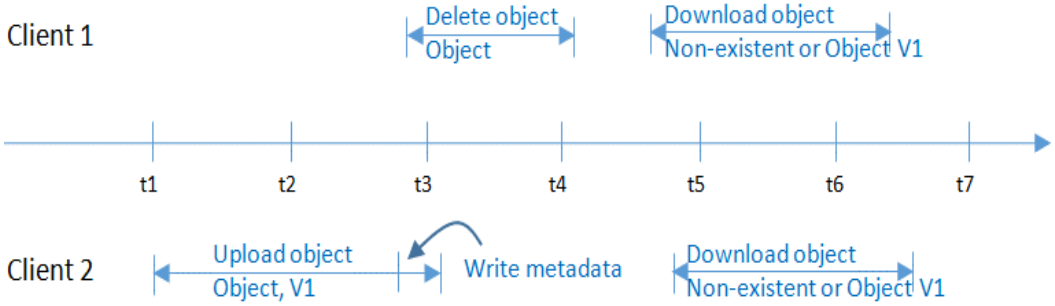
2. When client2 is uploading an object V1 and object metadata is not written yet, client1 deletes an object with the same name. In this scenario, the upload operation of client2 is still successful, and both client1 and client2 can access data object V1, as shown in **Figure 7-2**.

Figure 7-2 Concurrent upload and deletion of the same object (1)



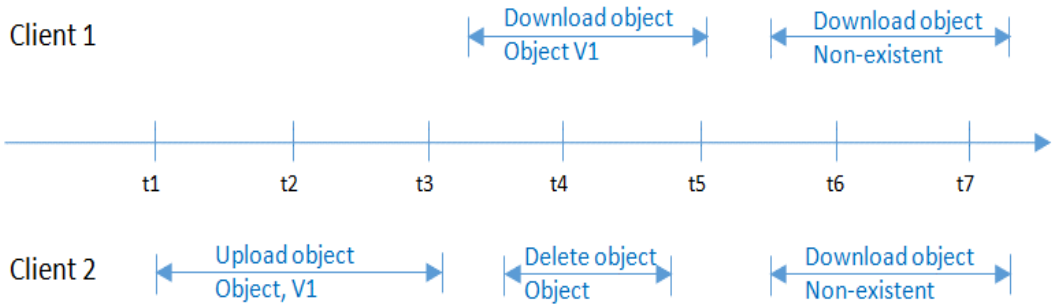
3. When client2 has successfully uploaded an object V1 and object metadata is still being written, client1 deletes an object with the same name. In this scenario, the upload operation of client2 is still successful. However, when client1 and client2 attempt to download the object, they may be able to access data object V1, or an error may be returned indicating that the object does not exist, as shown in **Figure 7-3**.

Figure 7-3 Concurrent upload and deletion of the same object (2)



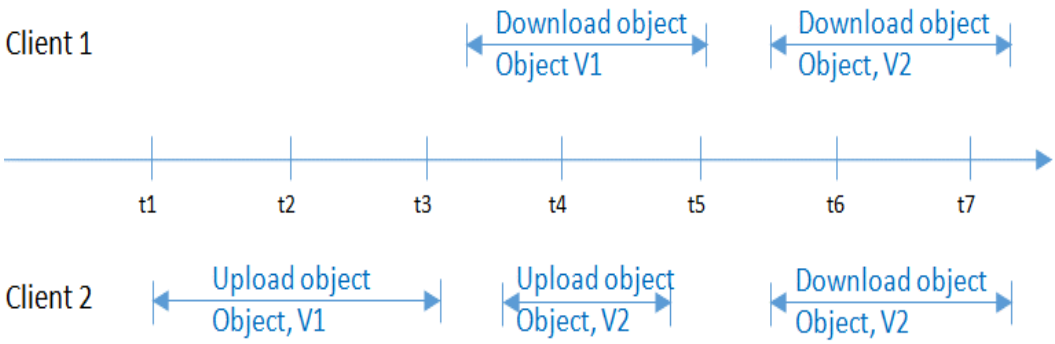
4. When client1 is downloading an object, client2 deletes an object with the same name. In this scenario, client1 may have downloaded a full copy or only part of the object data. After a deletion success message is returned to client2, an attempt to download the object will fail, and an error will be returned indicating that the object does not exist, as shown in **Figure 7-4**.

Figure 7-4 Concurrent download and deletion of the same object



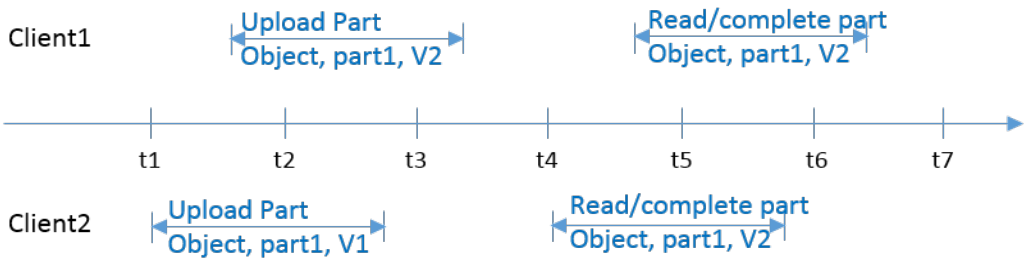
5. When client1 is downloading an object, client2 is updating an object with the same name. In this scenario, client1 may have downloaded a full copy or only part of the object data. After an update success message is returned to client 2, an attempt to download the object will succeed, and the latest data will be returned, as shown in **Figure 7-5**.

Figure 7-5 Concurrent download and update of the same object



6. When client2 is uploading part V1 of an object, client1 is uploading part V2 of the same object. After part V2 is uploaded successfully, both client1 and client2 can list the information about the multipart whose entity tag (ETag) is part V2, as shown in **Figure 7-6**.

Figure 7-6 Concurrently uploading the same part of the same object



A Change History

Date	Change History
2022-04-15	This is the second official release. This issue incorporates the following change: • Added the address for obtaining regions and endpoints.
2021-10-12	This is the first official release.