

Scalable File Service

API Reference

Issue 01
Date 2025-12-24



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2025. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: <https://www.huaweicloud.com/intl/en-us/>

Contents

1 Before You Start	1
2 API Overview	3
3 Calling APIs	5
3.1 Constructing a Request	5
3.2 Authentication	8
3.2.1 User Signature Authentication	8
3.2.2 Authentication of Signature in a Header	10
3.2.3 Signature Generators	17
3.3 Returned Values	17
4 Getting Started	19
4.1 Creating a General-Purpose File System	19
4.2 Listing General-Purpose File Systems	22
5 APIs	25
5.1 General-Purpose File System Management	25
5.1.1 Creating a General-Purpose File System	25
5.1.2 Deleting a General-Purpose File System	27
5.1.3 Listing General-Purpose File Systems	28
5.2 ACL Management	30
5.2.1 Configuring an ACL for a General-Purpose File System	30
5.2.2 Obtaining an ACL of a General-Purpose File System	33
5.2.3 Deleting an ACL from a General-Purpose File System	35
5.3 Tag Management	36
5.3.1 Batch Adding Tags to a Resource	37
5.3.2 Batch Deleting Tags from a Resource	40
5.3.3 Querying Tags of a Resource	42
5.3.4 Querying Resources by Tag	44
5.3.5 Querying the Number of Resources by Tag	50
5.3.6 Querying Tags in a Project	53
6 Permissions Policies and Supported Actions	56
6.1 Introduction	56
6.2 Supported Actions	57

7 Appendix.....	60
7.1 Status Codes.....	60
7.2 Error Codes.....	61
7.3 Obtaining Access Keys (AK/SK).....	68
7.4 Obtaining a Project ID.....	68

1 Before You Start

Overview

Welcome to *Scalable File Service API Reference*. Scalable File Service (SFS) is a network-attached storage (NAS) service that provides scalable, high-performance file storage. With SFS, you can enjoy shared file access spanning multiple Elastic Cloud Servers (ECSs), Bare Metal Servers (BMSs), and containers created on Cloud Container Engine (CCE).

This document describes how to use application programming interfaces (APIs) to perform operations on general-purpose file systems, such as creating, querying, and deleting a file system. For details about all supported operations, see [API Overview](#).

If you plan to access SFS through an API, ensure that you are familiar with SFS concepts. For details, see [Service Overview](#).

API Calling

SFS supports Representational State Transfer (REST) APIs, allowing you to call APIs using HTTP/HTTPS requests. For details about API calling, see [Calling APIs](#).

Endpoints

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. You can find the endpoint of SFS in the following table.

Table 1-1 Regions and endpoints

Region Name	Region ID	Endpoint	Protocol
CN North-Beijing4	cn-north-4	sfs3.cn-north-4.myhuaweicloud.com	HTTPS
CN East-Shanghai1	cn-east-3	sfs3.cn-east-3.myhuaweicloud.com	HTTPS

Region Name	Region ID	Endpoint	Protocol
CN South-Guangzhou	cn-south-1	sfs3.cn-south-1.myhuaweicloud.com	HTTPS
CN Southwest-Guiyang1	cn-southwest-2	sfs3.cn-southwest-2.myhuaweicloud.com	HTTPS
CN-Hong Kong	ap-southeast-1	sfs3.ap-southeast-1.myhuaweicloud.com	HTTPS

Constraints

- You can only call General-Purpose File System APIs on a private network, not a public network.
- For detailed constraints, see the constraints described in [APIs](#).

Concepts

- Account
An account is created upon successful registration. The account has full access permissions for all of its cloud services and resources. It can be used to reset user passwords and grant user permissions. The account is a payment entity, which should not be used directly to perform routine management. For security purposes, create Identity and Access Management (IAM) users and grant them permissions for routine management.
- User
An IAM user is created by an account in IAM to use cloud services. Each IAM user has its own identity credentials (password and access keys).
API authentication requires information such as the account name, username, and password.
- Region
Regions are divided based on geographical location and network latency. Public services, such as Elastic Cloud Server (ECS), Elastic Volume Service (EVS), Object Storage Service (OBS), Virtual Private Cloud (VPC), Elastic IP (EIP), and Image Management Service (IMS), are shared within the same region. Regions are classified into universal regions and dedicated regions. A universal region provides universal cloud services for common tenants. A dedicated region provides specific services for specific tenants.
For details, see [Region and AZ](#).
- AZ
An Availability Zone (AZ) comprises of one or more physical data centers equipped with independent ventilation, fire, water, and electricity facilities. Compute, network, storage, and other resources in an AZ are logically divided into multiple clusters. AZs within a region are interconnected using high-speed optical fibers to allow you to build cross-AZ, high-availability systems.

2 API Overview

These APIs allow you to use all SFS functions.

For details about whether an API supports Enterprise Project, see [Supported Actions](#).

If the description about an API in this document differs from that in the community, the description in this document is used.

File System Management APIs

Table 2-1 File system management APIs

API	Description
Creating a General-Purpose File System	Creates a general-purpose file system.
Deleting a General-Purpose File System	Deletes a general-purpose file system.
Listing General-Purpose File Systems	Lists general-purpose file systems.

ACL Management APIs

Table 2-2 ACL management APIs

API	Description
Configuring an ACL for a General-Purpose File System	Configures an ACL for a general-purpose file system.
Obtaining an ACL of a General-Purpose File System	Obtains an ACL of a general-purpose file system.
Deleting an ACL from a General-Purpose File System	Deletes an ACL from a general-purpose file system.

Tag Management APIs

Table 2-3 Tag management APIs

API	Description
Batch Adding Tags to a Resource	Batch adds tags to a general-purpose file system.
Batch Deleting Tags from a Resource	Batch deletes tags from a general-purpose file system.
Querying Tags of a Resource	Queries tags of a general-purpose file system.
Querying Resources by Tag	Queries general-purpose file systems by tag. General-purpose file systems are sorted by the time when they are created, in descending order.
Querying the Number of Resources by Tag	Queries the number of general-purpose file systems by tag.
Querying Tags in a Project	Queries tags of all general-purpose file systems owned by a tenant in a project.

3 Calling APIs

3.1 Constructing a Request

This section describes the structure of a REST API request.

Request URI

SFS uses URI to locate specific general-purpose file systems 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://[filesystem.]domain[:port]/[?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. SFS supports both HTTP and HTTPS.	Yes
filesystem	Resource path of a general-purpose file system, identifying only one general-purpose file system in SFS	No
domain	Domain name or IP address of the server for storing resources	Yes
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 port. Each transmission protocol has a default port. For example, HTTP uses port 80 and HTTPS uses port 443. In SFS, the default HTTP port is 80 and default HTTPS port is 443.	No

Parameter	Description	Mandatory
param	A specific resource contained by a general-purpose file system. Default value of this parameter indicates that the file system itself is obtained.	No

NOTICE

All API requests except those for listing file systems must contain the general-purpose file system name. To ensure DNS resolution performance and reliability, SFS requires that the general-purpose file system name must precede the domain when a request carrying a file system name is constructed to form a three-level domain name, also mentioned as virtual-hosted-style access domain name.

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 SFS

Method	Description
GET	Requests the server to return specific resources, for example, to list general-purpose file systems.
PUT	Requests the server to update specific resources, for example, to create general-purpose file systems.
POST	Requests a server to add resources or perform special operations.
DELETE	Requests the server to delete specified resources, for example, general-purpose file systems.
HEAD	Same as GET except that the server must return only the response header.
OPTIONS	Requests the server to check whether the user has the permissions to operate a resource.

Request Headers

Refers to optional and additional request fields, for example a field required by a specific URI or HTTP method. [Table 3-3](#) describes some common request header fields.

Table 3-3 Common request headers

Header	Description	Mandatory
Authorization	Signature information contained in a request message Type: string No default value.	Yes
Content-Length	The message length (excluding headers) defined in RFC 2616 Type: string No default value. Condition: 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	The time when a request is initiated, for example, Wed, 27 Jun 2018 13:39:15 +0000 . Type: string No default value. Condition: optional for requests containing header x-obs-date , required for other requests.	Conditionally required
Host	The host address, for example, filesystem.sfs3.region.myhuaweicloud.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 encoded 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 SFS resources can be accessed.
- **Coding**
You can use code to make API calls, and to assemble, send, and process request messages. It can be implemented by coding.

3.2 Authentication

3.2.1 User Signature Authentication

SFS signs a request using AK/SK. When a client is sending a request to SFS, 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 \(AK/SK\)](#).

SFS provides the signature calculation method based on the application scenario [Authentication of Signature in a Header](#).

[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 /HTTP/1.1 Host: filesystem.sfs3.region.myhuaweicloud.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913

Procedure		Example
	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 SFS.	PUT /object HTTP/1.1 Host: filesystem.sfs3.region.myhuaweicloud.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 / HTTP/1.1 Host: filesystem.sfs3.region.myhuaweicloud.com Date: Tue, 04 Jun 2019 06:54:59 GMT Content-Type: text/plain Content-Length: 5913 Authorization: OBS AccessKeyID:Signature
	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 General-Purpose File System API operations, the identity authentication can be done by carrying 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	An HTTP request method supported by the REST API. The value can be an HTTP verb such as PUT , GET , or DELETE .
Content-MD5	The base64-encoded 128-bit MD5 digest of the message according to RFC 1864. This parameter can be empty.
Content-Type	The message type, for example, text/plain . If a request does not contain this header field, this parameter will be processed as an empty string.
Date	The 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.

Parameter	Description
CanonicalizedHeaders	<p>The SFS 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-*.</p> <ol style="list-style-type: none"> 1. All characters of keywords in a request header field must be converted to lowercase letters (content values must be case sensitive, for example, x-obs-storage-class:STANDARD). 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.
CanonicalizedResource	<p>The SFS resource specified in an HTTP request. This parameter is constructed as follows: CanonicalizedResource = "/" + file system name + "?" + sub-resource</p> <ol style="list-style-type: none"> 1. General-purpose file system name, for example, /filesystem/. If there is no general-purpose file system name, use a slash (/). 2. If there is a sub-resource, add the sub-resource. Resource identifier: sfsacl 3. If there are multiple sub-resources, sort them in the alphabetical order from a to z, and use & to combine the sub-resources. <p>NOTE A sub-resource is usually unique. Do not specify multiple values for the same sub-resource (such as key=value1&key=value2). Otherwise, only the first value is used.</p>

The following table provides some examples of generating StringToSign.

Table 3-6 Obtaining the ACL of a general-purpose file system

Request Header	StringToSign
GET /?sfsacl HTTP/1.1 Host: filesystem.sfs3.region.myhuaweicloud.com Date: Sat, 12 Oct 2015 08:12:38 GMT	GET \n \n \n Sat, 12 Oct 2015 08:12:38 GMT\n /filesystem/?sfsacl

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 general-purpose file system named **newfilesystem2** in a region, the client request format is as follows:

```
PUT / HTTP/1.1
Host: newfilesystem2.sfs3.region.myhuaweicloud.com
Content-Length: length
Date: Fri, 06 Jul 2018 03:45:51 GMT
x-obs-acl:private
x-obs-storage-class:STANDARD
Authorization: OBS UDSIAMSTUBTEST000254:ydH8ffpcbS6YpeOMcEZfn0wE90c=

<CreateBucketConfiguration xmlns="http://obs.myhwclouds.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 SFS_PREFIX = "x-obs-";

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

    private static final List<String> SUB_RESOURCES = Collections.unmodifiableList(Arrays.asList(
        "CDNNotifyConfiguration", "acl", "append", "attname", "backtosource", "cors", "customdomain",
        "delete",
        "deletebucket", "directcoldaccess", "encryption", "inventory", "length", "lifecycle", "location",
        "logging",
        "metadata", "modify", "name", "notification", "orchestration", "partNumber", "policy", "position",
        "quota",
        "rename", "replication", "requestPayment", "response-cache-control", "response-content-
        disposition",
        "response-content-encoding", "response-content-language", "response-content-type", "response-
        expires",
        "restore", "select", "storageClass", "storagePolicy", "storageinfo", "tagging", "torrent", "truncate",
        "uploadId", "uploads", "versionId", "versioning", "versions", "website", "x-image-process",
        "x-image-save-bucket", "x-image-save-object", "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[] {"private"});
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 os
import hashlib
import hmac
import binascii
from datetime import datetime
IS_PYTHON2 = sys.version_info.major == 2 or sys.version < '3'
"""Hard-coded or plaintext SecretAccessKeyID is risky. For security purposes, encrypt your access key, store it in the configuration file or environment variables, and decrypt it before using it. In this example, SecretAccessKeyID is stored in the environment variables for identity authentication. Before running the code in this example, configure environment variable SECRET_ACCESS_KEY_ID."""
yourSecretAccessKeyID = os.environ.get('SECRET_ACCESS_KEY_ID')
httpMethod = "PUT"
contentType = "application/xml"
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\n"
CanonicalizedResource = "/newfilesystem2"
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=

Signature Algorithm in the C Programming Language

[Download](#) the sample code for calculating the signature in the C programming language.

1. The API for calculating the signature is contained in the **sign.h** header file.
2. The sample code for calculating the signature is contained in the **main.c** header file.

Signature Mismatch Error Handling

During an SFS API call, if the following error is reported,

Status code: 403 Forbidden

Error code: SignatureDoesNotMatch

Error message: The request signature we calculated does not match the signature you provided. Check your key and signing method.

Contact technical support.

3.2.3 Signature Generators

SFS offers visualized tools for you to easily generate signatures.

Table 3-7 Signature generators

Calculation Method	How to Obtain
Authenticating the signature in a header	Visit Generate Header .

During an SFS API call, if the following error is reported,

Status code: 403 Forbidden

Error code: SignatureDoesNotMatch

Error message: The request signature we calculated does not match the signature you provided. Check your key and signing method.

Contact technical support.

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 indicating the status of a response. It ranges from 2xx (indicating successes) to 4xx or 5xx (indicating errors). 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 3-8](#).

Table 3-8 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 time when a response is returned. Type: string Default value: none
x-obs-id-2	A special symbol that helps troubleshoot faults. Type: string Default value: none
x-obs-request-id	The unique identifier of the request. The value is generated by the SFS service and can be used for troubleshooting. 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 General-Purpose File System

Scenarios

General-purpose file systems are containers that store files in SFS. You need to create a general-purpose file system before storing data in SFS.

The following describes how to call the [Creating a General-Purpose File System](#) API in a region. For details about the calling method, see [Calling APIs](#).

Prerequisites

- You have obtained the AK and SK. For details, see [Obtaining Access Keys \(AK/SK\)](#).
- You have planned the region where you want to create a general-purpose file system and obtained the endpoint required for API calls. For details, see [Endpoints](#).

Once a region is determined, it cannot be modified after the creation is complete.

Creating a General-Purpose File System Named filesystem001 in the a1 Region

In this example, an Apache HttpClient is used.

```
package com.sfsclient;

import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.List;
import java.util.Map;

public class TestMain {
    //Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using hard
    coding may result in leakage.
    //Obtain an AK/SK pair on the management console.
    public static String accessKey = System.getenv("HUAWEICLOUD_SDK_AK");
    public static String securityKey = System.getenv("HUAWEICLOUD_SDK_SK");
    public static String region = "cn-east-3"; // The region where you plan to create the general-purpose file
```

```
system.  
  
    public static String endpoint = "sfs3.a1.region.com"; // The address of the General-Purpose File System  
    service.  
  
    public static String createSfsBody =  
        "<CreateBucketConfiguration >\n" +  
        "<Location>" + region + "</Location>\n" +  
        "</CreateBucketConfiguration>";  
  
    public static void main(String[] str) {  
  
        createFileSystem();  
  
    }  
    private static void createFileSystem() {  
        // The general-purpose file system name.  
        String fileName = "example-sfs-001";  
  
        String httpMethod = "PUT";  
        String date = DateUtils.formatDate(System.currentTimeMillis());  
        String contentType = "application/xml";  
  
        /**Calculate the signature based on the request.**/  
        String contentMD5 = "";  
        String canonicalizedHeaders = "x-obs-bucket-type:SFS";  
        String canonicalizedResource = "/" + fileName ;  
  
        // 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 stringToSign = httpMethod + "\n" +  
            contentMD5 + "\n" +  
            contentType + "\n" +  
            date + "\n" +  
            canonicalizedHeaders + "\n" + canonicalizedResource;  
  
        System.out.printf("StringToSign:\n[%s]\n\n", stringToSign);  
  
        HttpURLConnection conn = null;  
  
        try {  
            String signature = Signature.signWithHmacSha1 (securityKey, stringToSign);  
            String authorization= "OBS " + accessKey + ":" + signature;  
            System.out.printf("authorization:%s\n\n", authorization);  
  
            URL url = new URL("http://" + endpoint + "/" + fileName);  
            conn = (HttpURLConnection) url.openConnection();  
  
            // Add a signature header.  
            conn.setRequestMethod(httpMethod);  
            conn.setRequestProperty("Date", date);  
            conn.setRequestProperty("Content-Type", contentType);  
            conn.setRequestProperty("x-obs-bucket-type", "SFS");  
            conn.setRequestProperty("Authorization", authorization);  
            conn.setDoOutput(true);  
  
            // Add a body.  
            OutputStream out = conn.getOutputStream();  
            out.write(createSfsBody.getBytes());  
            out.flush();  
            out.close();  
  
            String status = conn.getHeaderField(null);  
            System.out.println(status);  
  
            // Output the response message.  
            Map<String, List<String>> headers = conn.getHeaderFields();  
            for (Map.Entry<String, List<String>> entry : headers.entrySet()) {  
                String key = entry.getKey();
```

```
        List<String> values = entry.getValue();
        if (key != null) {
            for (String value : values) {
                System.out.println(key + ": " + value);
            }
        }
    }
}
// Handle the request error.
int statusCode = conn.getResponseCode();
if (statusCode != HttpURLConnection.HTTP_OK && statusCode !=
HttpURLConnection.HTTP_NO_CONTENT) {
    InputStream errorStream = conn.getErrorStream();
    BufferedReader reader = new BufferedReader(new InputStreamReader(errorStream));
    StringBuilder responseBody = new StringBuilder();
    String line;
    while ((line = reader.readLine()) != null) {
        responseBody.append(line);
    }
    reader.close();

    System.out.println("Error: " + responseBody);
}
} catch (IOException e) {
    e.printStackTrace();
} finally {
    if (conn != null){
        conn.disconnect();
    }
}
}
```

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

```
package com.sfsclient;

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.sfsclient;

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 General-Purpose File Systems

Scenarios

If you want to view information about all general-purpose file systems created by yourself, you can call the API for listing general-purpose file systems.

The following describes how to call the API for [Listing General-Purpose File Systems](#). For details, see [Getting Started](#).

Prerequisites

- You have obtained the AK and SK. For details, see [Obtaining Access Keys \(AK/SK\)](#).
- You have planned the region where you want to view general-purpose file systems and obtained the endpoint required for API calls. For details, see [Endpoints](#).

Listing General-Purpose File Systems in the a1 Region

In this example, an Apache HttpClient is used.

```
package com.sfsclient;

import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.List;
import java.util.Map;

public class TestMain {
    //Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using hard
    coding may result in leakage.
    //Obtain an AK/SK pair on the management console.
    public static String accessKey = "ACCESS_KEY_ID";
    public static String securityKey = "SECRET_ACCESS_KEY_ID";
    public static String endpoint = "sfs3.a1.region.com"; // The address of the General-Purpose File System
    service.

    public static void main(String[] str) {
        createFileSystem();
    }
    private static void listFileSystem() {
        String httpMethod = "GET";
        String date = DateUtils.formatDate(System.currentTimeMillis());

        /**Calculate the signature based on the request.**/
        String contentMD5 = "";
        String contentType = "";
        String canonicalizedHeaders = "x-obs-bucket-type:SFS";
```

```
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 stringToSign = httpMethod + "\n" +
    contentMD5 + "\n" +
    contentType + "\n" +
    date + "\n" +
    canonicalizedHeaders + "\n" + canonicalizedResource;

System.out.printf("StringToSign:\n[%s]\n\n", stringToSign);

URLConnection conn = null;

try {
    String signature = Signature.signWithHmacSha1 (securityKey, stringToSign);
    String authorization= "OBS " + accessKey + ":" + signature;
    System.out.printf("authorization:%s\n\n", authorization);

    // Create an HTTP request.
    URL url = new URL("http://" + endpoint);
    conn = (URLConnection) url.openConnection();

    // Add a signature header.
    conn.setRequestMethod(httpMethod);
    conn.setRequestProperty("Date", date);
    conn.setRequestProperty("Content-Type", contentType);
    conn.setRequestProperty("x-obs-bucket-type", "SFS");
    conn.setRequestProperty("Authorization", authorization);
    conn.setDoOutput(true);

    String status = conn.getHeaderField(null);
    System.out.println(status);

    // Output the response message.
    Map<String, List<String>> headers = conn.getHeaderFields();
    for (Map.Entry<String, List<String>> entry : headers.entrySet()) {
        String key = entry.getKey();
        List<String> values = entry.getValue();
        if (key != null) {
            for (String value : values) {
                System.out.println(key + ": " + value);
            }
        }
    }
    // Process the returned content.
    int statusCode = conn.getResponseCode();
    if (statusCode == HttpURLConnection.HTTP_OK) {
        InputStream responseStream = conn.getInputStream();
        BufferedReader reader = new BufferedReader(new InputStreamReader(responseStream));

        StringBuilder responseBody = new StringBuilder();
        String line;
        while ((line = reader.readLine()) != null) {
            responseBody.append(line);
        }
        reader.close();

        System.out.println("responseBody: " + responseBody);
    } else {
        System.out.println("Error: " + statusCode);
    }
} catch (IOException e) {
    e.printStackTrace();
} finally {
    if (conn != null){
        conn.disconnect();
    }
}
```

```
}  
}
```

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

```
package com.sfsclient;  
  
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.sfsclient;  
  
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 General-Purpose File System Management

5.1.1 Creating a General-Purpose File System

Function

This API is used to create a general-purpose file system with a specified name.

NOTE

- The name of a general-purpose file system must be unique in SFS. If you create a file system with the same name as an existing one created by yourself in a given region, a success response is returned. In other cases, if a file system with the same name is created, an error message is returned, indicating that the file system already exists.
- By default, you can create up to 100 file systems.
- If a general-purpose file system is deleted, you can create a new file system with the same name after at least 12 hours.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

PUT /

Request Parameters

Table 5-1 Request header parameters

Parameter	Mandatory	Type	Description
Authorization	Yes	String	The signature information.
Date	Yes	String	The request time.
x-obs-az-redundancy	No	String	The AZ redundancy, single-AZ or multi-AZ.
x-obs-bucket-type	Yes	String	The header used to specify the general-purpose file system creation.
Host	Yes	String	The host address.
x-obs-epid	No	String	The enterprise project ID.

Table 5-2 Request body parameter

Parameter	Mandatory	Type	Description
Location	No	String	The region.

Response Parameters

This response uses common headers. For details, see [Table 3-8](#).

(Optional) Response Body

A response body contains information other than the response header. It is usually sent in a structured format (JSON or XML) defined by the response header parameter **Content-type**.

Example Request

- Create a general-purpose file system in the example region. The host address is Host: example-sfs-01.sfs3.example.region.com:443. The enterprise project ID is 0.

```
PUT / HTTP/1.1
Host: example-sfs-01.sfs3.example.region.com:443
Date: Wed, 07 Jun 2023 02:38:09 GMT
x-obs-bucket-type: SFS
Authorization: OBS FNEX1B77SXDIB3TFMSZZ:0Xsnu4hJVOI7VWH0wlQczVN+rbg=
Content-Length: 85
x-obs-epid: 0
<CreateBucketConfiguration>
  <Location>example</Location>
</CreateBucketConfiguration>
```

Example Response

```
HTTP/1.1 200 OK
Server: OBS
X-Obs-Request-Id: 0000018893B8058EC0470388BE6EDE88
Location: /example-sfs-01
X-Obs-Id-2: 32AAQAAEAABSAAgAAEAABAAQAAEAABCTRa4voOUvr50ncznQT/hligMxL4so2z
Date: Wed, 07 Jun 2023 02:38:11 GMT
Content-Length: 0
```

Status Codes

Status Code	Description
200	The general-purpose file system is created.

Error Codes

See [Error Codes](#).

5.1.2 Deleting a General-Purpose File System

Function

This API is used to delete a general-purpose file system.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

DELETE /

Request Parameters

Table 5-3 Request header parameters

Parameter	Mandatory	Type	Description
Authorization	Yes	String	The signature header field.
Date	Yes	String	The request time.
Host	Yes	String	The host address.

Response Parameters

This response uses common headers. For details, see [Table 3-8](#).

Example Request

```
DELETE / HTTP/1.1
User-Agent: curl/7.29.0
Host: examplefilesystem.sfs3.example.region.com
Accept: */*
Date: WED, 01 Jul 2015 02:31:25 GMT
Authorization: OBS H4lPJX0TQTHHEBQQCEC:jZiAT8Vx4azWEvPRMWi0X5BpJMA=

HTTP/1.1 204 No Content
Server: OBS
X-Obs-Request-Id: 0000018893B8081DC047305E783867DD
X-Obs-Id-2: 32AAAQAAEAABSkAgAAEAABAAAQAAEAABCT5UWgsaro3EEEnOsNEzf8w8dnydR+Eak
Date: WED, 01 Jul 2015 02:31:25 GMT
```

Status Codes

Status Code	Description
204	The general-purpose file system is deleted.

Error Codes

See [Error Codes](#).

5.1.3 Listing General-Purpose File Systems

Function

This API is used to list general-purpose file systems.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

GET /

Request Parameters

Table 5-4 Request header parameters

Parameter	Mandatory	Type	Description
x-obs-bucket-type	Yes	String	This header field is used to specify the content to be obtained. The value is SFS , which lists all general-purpose file systems.

Parameter	Mandatory	Type	Description
Authorization	Yes	String	The signature information.
Date	Yes	String	The request time.
Host	Yes	String	The host address.

Response Parameters

Status code: 200

Table 5-5 Response body parameters

Parameter	Type	Description
Owner	Owner object	The owner information of the general-purpose file system, including the tenant ID.
Buckets	Buckets object	The general-purpose file systems owned by the user.

Table 5-6 Owner

Parameter	Type	Description
ID	String	The domain ID (account ID) of a user.

Table 5-7 Buckets

Parameter	Type	Description
Bucket	Bucket object	The detailed information of the general-purpose file system.

Table 5-8 Bucket

Parameter	Type	Description
Name	String	The name of the general-purpose file system.
CreationDate	String	The time when the general-purpose file system was created.
Location	String	The location of the general-purpose file system.

Example Request

```
GET / HTTP/1.1
Host: sfs3.example.region.com
Date: date
x-obs-bucket-type: SFS
Authorization: authorization
```

Example Response

```
HTTP/1.1 200 OK
Server: OBS
X-Obs-Request-Id: 0000018893B8126DC048B06DD3816BD4
X-Obs-Id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCTMZh3Thi7lcDxuGWu9Qtp9PJbYXa7Ib
Date: Wed, 07 Jun 2023 02:38:14 GMT
Content-Type: application/xml
Content-Length: 377

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListAllMyBucketsResult xmlns="http://obs.myhwclouds.com/doc/2015-06-30/">
  <Owner>
    <ID>783fc6652cf246c096ea836694f71855</ID>
  </Owner>
  <Buckets>
    <Bucket>
      <Name>examplebucket01</Name>
      <CreationDate>2018-06-21T09:15:01.032Z</CreationDate>
      <Location>example-region-1</Location>
      <BucketType>SFS</BucketType>
    </Bucket>
    <Bucket>
      <Name>examplebucket02</Name>
      <CreationDate>2018-06-22T03:56:33.700Z</CreationDate>
      <Location>example-region-2</Location>
      <BucketType>SFS</BucketType>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

Status Codes

Status Code	Description
200	General-purpose file systems are listed.

Error Codes

See [Error Codes](#).

5.2 ACL Management

5.2.1 Configuring an ACL for a General-Purpose File System

Function

This API is used to configure an ACL for a general-purpose file system.

 NOTE

After an ACL is configured for a general-purpose file system, the configuration takes about 30 seconds to take effect.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

PUT /

Table 5-9 Query parameter

Parameter	Mandatory	Type	Description
sfsacl	Yes	String	/

Request Parameters

Table 5-10 Request header parameters

Parameter	Mandatory	Type	Description
Date	Yes	String	The request time.
Authorization	Yes	String	The signature information.
Host	Yes	String	The host address.

Table 5-11 Request body parameter

Parameter	Mandatory	Type	Description
Statement	No	Array of Statement objects	The unique identification.

Table 5-12 Statement

Parameter	Mandatory	Type	Description
Sid	No	String	The statement ID.

Parameter	Mandatory	Type	Description
Action	Yes	String	The allowed statement action. Enumerated values: <ul style="list-style-type: none"> • FullControl: read/write • Read: read-only
Effect	Yes	String	The effect specifying that the statement permission is Allow . Enumerated value: <ul style="list-style-type: none"> • Allow
Condition	Yes	Condition object	The conditions for a statement to take effect.

Table 5-13 Condition

Parameter	Mandatory	Type	Description
SourceVpc	Yes	String	A specified VPC ID.
VpcSourceIp	No	Array of strings	Specific IP addresses or IP address ranges. This parameter is currently not supported.

Response Parameters

This response uses common headers. For details, see [Table 3-8](#).

Example Request

Configuring an ACL for a general-purpose file system to grant the read/write permissions for IP addresses **127.0.0.1/24** and **192.168.1.85/24** in the VPC whose ID is **241dbf6b-dc5d-41b2-9108-ca5e56b48386**:

```
PUT /?sfsacl HTTP/1.1
Host: examplefilesystem.sfs3.example.region.com
Date: WED, 01 Jul 2015 02:32:25 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC;jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

```
{
  "Statement": [{
    "Sid": "Stmt1375240018061",
    "Action": "FullControl",
    "Effect": "Allow",
    "Condition": {
      "SourceVpc": "241dbf6b-dc5d-41b2-9108-ca5e56b48386",
      "VpcSourceIp": ["127.0.0.1/24", "192.168.1.85/24"]
    }
  }]
}
```

Example Response

```
HTTP/1.1 204 OK
Server: OBS
X-Obs-Request-Id: 0000018893B8073AC04721AA7EE3408B
X-Obs-Id-2: 32AAAQAAEAABSAAgAAEAABAAAQAAEAABCS5QDe0QLbFNz6FXoKuXHzD2wS0eJQaj
Date: Wed, 07 Jun 2023 02:38:11 GMT
```

Status Codes

Status Code	Description
204	The ACL is configured for the general-purpose file system.

Error Codes

See [Error Codes](#).

5.2.2 Obtaining an ACL of a General-Purpose File System

Function

This API is used to obtain an ACL of a general-purpose file system.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

GET /

Table 5-14 Query parameter

Parameter	Mandatory	Type	Description
sfsacl	Yes	String	/

Request Parameters

Table 5-15 Request header parameters

Parameter	Mandatory	Type	Description
Date	Yes	String	The request time.
Authorization	Yes	String	The signature information.

Parameter	Mandatory	Type	Description
Host	Yes	String	The host address.

Response Parameters

Status code: 200

Table 5-16 Response body parameter

Parameter	Type	Description
Statement	Array of Statement objects	The unique identification.

Table 5-17 Statement

Parameter	Type	Description
Sid	String	The statement ID.
Action	String	The allowed statement action. Enumerated values: <ul style="list-style-type: none"> • FullControl: read/write • Read: read-only
Effect	String	The effect specifying that the statement permission is Allow . Enumerated value: <ul style="list-style-type: none"> • Allow
Condition	Condition object	The conditions for a statement to take effect.

Table 5-18 Condition

Parameter	Type	Description
SourceVpc	String	A specified VPC ID.
VpcSourceIp	Array of strings	Specific IP addresses or IP address ranges. This parameter is currently not supported.

Example Request

```
GET /?sfsacl HTTP/1.1
Host: example-sfs-01.sfs3.example.region.com:443
```

Date: Wed, 07 Jun 2023 03:31:46 GMT
Authorization: OBS FNEX1B77SXDIB3TFMSZZ:eUqPIHnPDWGDTlgyLmsALA86wys=

Example Response

```
HTTP/1.1 200 OK
Server: OBS
Content-Type: application/json
Content-Length: 131
Date: Wed, 07 Jun 2023 03:31:59 GMT
X-Obs-Request-Id: 0000018893E94B65C046B527778F8F14
X-Obs-Id-2: 32AAAQAAEAABAAAQAAEAABAAAQAAEAABCS2lEdSHcA04319WknB1DD5BdBKuGr1
{
  "Statement": [
    {
      "Condition": {
        "SourceVpc": "f85adabc-a387-4d1d-94cf-65ef9034f752"
      },
      "Action": "FullControl",
      "Effect": "Allow",
      "Sid": ""
    }
  ]
}
```

Status Codes

Status Code	Description
200	The ACL of the general-purpose file system is obtained.

Error Codes

See [Error Codes](#).

5.2.3 Deleting an ACL from a General-Purpose File System

Function

This API is used to delete an ACL from a general-purpose file system.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

DELETE /

Table 5-19 Query parameter

Parameter	Mandatory	Type	Description
sfsacl	Yes	String	/

Request Parameters

Table 5-20 Request header parameters

Parameter	Mandatory	Type	Description
Date	Yes	String	The request time.
Authorization	Yes	String	The signature information.
Host	Yes	String	The host address.

Response Parameters

This response uses common headers. For details, see [Table 3-8](#).

Example Request

```
DELETE /?sfsacl HTTP/1.1
Host: examplefilesystem.sfs3.example.region.com
Date: WED, 01 Jul 2015 02:36:06 GMT
Authorization: OBS H4IPJX0TQTHTHEBQQCEC;jZiAT8Vx4azWEvPRMWi0X5BpJMA=
```

Example Response

```
HTTP/1.1 204 No Content
Server: OBS
X-Obs-Id-2: 32AAAQAAEAABSAAgAAEAABAAQAAEAABCSj4dxiqb1Lw50CTjVQeV3ebh3QQ6PAj
X-Obs-Request-Id: 0000018893B807D5C0472A6161D87032
Date: WED, 01 Jul 2015 02:36:06 GMT
```

Status Codes

Status Code	Description
204	The ACL of the general-purpose file system is deleted.

Error Codes

See [Error Codes](#).

5.3 Tag Management

5.3.1 Batch Adding Tags to a Resource

Function

This API is used to batch add tags to a general-purpose file system. You can add up to 20 tags to a resource.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

- POST /v3/sfs/tms/{project_id}/file-systems/{resource_id}/tags/create
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	The project ID. For details, see Obtaining a Project ID .
resource_id	Yes	String	The resource ID, which is the name of a general-purpose file system.

Request Parameters

Table 5-21 Request header parameters

Parameter	Mandatory	Type	Description
Content-type	Yes	String	The MIME type of the request body. Example: application/json
X-Auth-Token	No	String	The user token.

Table 5-22 Request body parameters

Parameter	Mandatory	Type	Description
tags	No	List<resource_tag>	<p>The tag list. For details, see Table 5-23.</p> <p>This parameter is mandatory for common tenants.</p> <p>Use either tags or sys_tags if you have the op_service permissions.</p>
sys_tags	No	List<resource_tag>	<p>The system tag list. This parameter is only available to the op_service permissions.</p> <p>Use either tags or sys_tags if you have the op_service permissions.</p> <p>Only one resource_tag structure is used in TMS calls currently.</p> <p>The key is fixed at _sys_enterprise_project_id.</p> <p>The value can be UUID or 0. 0 indicates the default enterprise project.</p> <p>System tags can only be added.</p> <p>For details, see Table 5-23.</p>

Table 5-23 resource_tag

Parameter	Mandatory	Type	Description
key	Yes	String	<p>The tag key. A tag key can contain a maximum of 128 characters. It can contain letters, digits, and spaces representable in UTF-8 and special characters (<code>_.:=+@</code>). It cannot start or end with a space and cannot be left empty. Tag keys starting with <code>_sys_</code> are system tags, and you cannot start a tag key with <code>_sys_</code>.</p>

Parameter	Mandatory	Type	Description
value	Yes	String	The tag value. A tag value can contain a maximum of 255 characters. It can contain letters, digits, and spaces representable in UTF-8 and special characters (_.:+=-@) and can be left empty. It cannot start or end with a space.

Response Parameters

None

Example Request

Batch adding tags to a general-purpose file system whose name is **bucketName** with the project ID **c80a2157ba1d46c0825265947342077c**:

POST https://{endpoint}/v3/sfs/tms/c80a2157ba1d46c0825265947342077c/file-systems/bucketName/tags/create

Request body example:

```
{
  "tags":[
    {
      "key":"key1",
      "value":"value1"
    },
    {
      "key":"key2",
      "value":"value2"
    }
  ]
}
```

Example Response

None

Status Codes

- Normal

Status Code	Description
204	Resource tags added in a batch.

- Abnormal

Status Code	Description
400	Invalid tag parameter.
401	Certification failed.
403	Authentication failed.
404	Resource not found.
500	System error.

5.3.2 Batch Deleting Tags from a Resource

Function

This API is used to batch delete tags from a resource. System tags cannot be deleted. If any tag to be deleted is not found, a successful result is returned.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

- POST /v3/sfs/tms/{project_id}/file-systems/{resource_id}/tags/delete
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	The project ID. For details, see Obtaining a Project ID .
resource_id	Yes	String	The resource ID, which is the name of a general-purpose file system.

Request Parameters

Table 5-24 Request header parameters

Parameter	Mandatory	Type	Description
Content-type	Yes	String	The MIME type of the request body. Example: application/json

Parameter	Mandatory	Type	Description
X-Auth-Token	No	String	The user token.

Table 5-25 Request body parameter

Parameter	Mandatory	Type	Description
tags	Yes	List< resource _tag >	The tag list. For details, see Table 5-23 .

Response Parameters

None

Example Request

Batch deleting tags from a general-purpose file system whose name is **bucketName** with the project ID **c80a2157ba1d46c0825265947342077c**:

POST https://{endpoint}/v3/sfs/tms/c80a2157ba1d46c0825265947342077c/file-systems/bucketName/tags/delete

Request body example:

```
{
  "tags": [
    {
      "key": "key1"
    },
    {
      "key": "key2",
      "value": "value2"
    }
  ]
}
```

Example Response

None

Status Codes

- Normal

Status Code	Description
204	Resource tags deleted in a batch.

- Abnormal

Status Code	Description
400	Invalid tag parameter.
401	Certification failed.
403	Authentication failed.
404	Resource not found.
500	System error.

5.3.3 Querying Tags of a Resource

Function

This API is used to query tags of a resource.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

- GET /v3/sfs/tms/{project_id}/file-systems/{resource_id}/tags
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	The project ID. For details, see Obtaining a Project ID .
resource_id	Yes	String	The resource ID, which is the name of a general-purpose file system.

Request Parameters

Table 5-26 Request header parameters

Parameter	Mandatory	Type	Description
Content-type	Yes	String	The MIME type of the request body. Example: application/json
X-Auth-Token	No	String	The user token.

Response Parameters

Table 5-27 Response body parameters

Parameter	Mandatory	Type	Description
tags	No	List< resource_tag >	The tag list. For details, see Table 5-23 .
sys_tags	No	List< resource_tag >	The system tag list. This parameter is only available to users with the op_service permission. It contains only one resource_tag structure currently. The key is fixed at _sys_enterprise_project_id . The value is the ID of an enterprise project. Value 0 indicates the default enterprise project. For details, see Table 5-23 .

Example Request

Querying tags of a general-purpose file system whose name is **bucketName** with the project ID **c80a2157ba1d46c0825265947342077c**:

```
GET https://{endpoint}/v3/sfs/tms/c80a2157ba1d46c0825265947342077c/file-systems/bucketName/tags
```

Example Response

```
{
  "tags": [
    {
      "key": "key1",
      "value": "value1"
    },
    {
      "key": "key2",
      "value": "value2"
    }
  ]
}
```

Status Codes

- Normal

Status Code	Description
200	Tags of a resource are queried.

- Abnormal

Status Code	Description
400	Invalid tag parameter.
401	Certification failed.
403	Authentication failed.
404	Resource not found.
500	System error.

5.3.4 Querying Resources by Tag

Function

This API is used to query resources by tag. Resources are sorted by the time when they are created, in descending order.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

- POST /v3/sfs/tms/{project_id}/file-systems/resource-instances/filter
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	The project ID. For details, see Obtaining a Project ID .
limit	No	Int	The maximum number of records that can be queried. The value ranges from 1 to 1000 , and the default value is 1000 .

Parameter	Mandatory	Type	Description
offset	No	Int	The index location. The query starts from the next data record indexed by this parameter. The default value is 0 , indicating that the query starts from the first data record. The value cannot be a negative number.

Request Parameters

Table 5-28 Request header parameters

Parameter	Mandatory	Type	Description
Content-type	Yes	String	The MIME type of the request body. Example: application/json
X-Auth-Token	No	String	The user token.

Table 5-29 Request body parameters

Parameter	Mandatory	Type	Description
without_any_tag	No	boolean	Excludes any tags. If this parameter is set to true , all resources without tags are queried. In this case, the tags field is ignored. If this parameter is set to false or not specified, it does not take effect, meaning that all resources are returned or resources are filtered by tags or matches .

Parameter	Mandatory	Type	Description
tags	No	List<tag>	<p>Includes tags. A maximum of 20 tags can be specified. Tag keys must be unique. Each tag key can have a maximum of 20 tag values. A tag value can be empty but the structure cannot be missing. Tag values of the same key must be unique.</p> <p>The response returns resources containing all tags in this list. Keys in this list are in the AND relationship and values in each key-value structure are in the OR relationship. If this parameter is not specified, all resources will be returned.</p> <p>For details, see Table 5-30.</p>
sys_tags	No	List<tag>	<p>Includes system tags. This parameter is only available to users with the op_service permission.</p> <p>It cannot be used with filtering criteria without_any_tag and tags at the same time.</p> <p>It contains only one tag structure currently.</p> <p>The key is fixed at _sys_enterprise_project_id.</p> <p>The value is the ID of an enterprise project.</p> <p>The key contains only one value. Value 0 indicates the default enterprise project.</p> <p>For details, see Table 5-30.</p>
matches	No	List<match>	<p>The fields to be matched. The key in match is a dictionary value fixed at resource_name, meaning that the prefix search is performed based on the value of the key. It will be extended later.</p> <p>For details, see Table 5-31.</p>

Table 5-30 tag

Parameter	Mandatory	Type	Description
key	Yes	String	The tag key. A tag key can contain a maximum of 128 characters. It can contain letters, digits, and spaces representable in UTF-8 and special characters (<code>_:=+@</code>). It cannot start or end with a space and cannot be left empty. Tag keys starting with <code>_sys_</code> are system tags, and you cannot start a tag key with <code>_sys_</code> .
values	Yes	List<String>	The tag value list. A value can be an empty array but cannot be left blank. If values is an empty array, any value is queried. The values are in the OR relationship.

Table 5-31 match

Parameter	Mandatory	Type	Description
key	Yes	String	The key. The key is fixed at resource_name currently. Other key values will be available later.
value	Yes	String	The value. Each tag value can contain a maximum of 255 Unicode characters. The character set is not verified.

Response Parameters

Table 5-32 Response body parameters

Parameter	Mandatory	Type	Description
resources	Yes	List< resource >	The resource list. For details, see Table 5-33 .
total_count	Yes	Integer	The total number of records.

Table 5-33 resource

Parameter	Mandatory	Type	Description
resource_id	Yes	String	The resource ID.
resource_detail	Yes	Object	The resource details. It is left blank by default. The value is a resource object used for extension.
tags	Yes	List<resource_tag>	The tag list. If there is no tag, an empty array is used by default. For details, see Table 5-23 .
sys_tags	No	List<resource_tag>	This parameter is only available to users with the op_service permission. It contains only one resource_tag structure currently. The key is fixed at _sys_enterprise_project_id . The value is the ID of the enterprise project. Value 0 indicates the default enterprise project. For details, see Table 5-23 .
resource_name	Yes	String	The resource name. This parameter is an empty string if there is no resource name.

Example Request

Listing resources (with the project ID set to c80a2157ba1d46c0825265947342077c, offset to 0, and limit to 10):

```
POST https://{endpoint}/v3/sfs/tms/c80a2157ba1d46c0825265947342077c/file-systems/resource-instances/filter?limit=10&offset=0
```

Request body example:

```
{
  "tags":[
    {
      "key":"key1",
      "values":[
        "value1",
        "value2"
      ]
    },
    {
      "key":"key2",
      "values":[]
    }
  ]
}
```

```

        "value1",
        "value2"
    ]
  }
],
"matches":[
  {
    "key":"resource_name",
    "value":"resource1"
  }
],
"without_any_tag":"false"
}

```

Example Response

```

{
  "resources":[
    {
      "resource_detail":"","
      "resource_id":"resource1",
      "resource_name":"resource1",
      "tags":[
        {
          "key":"key1",
          "value":"value1"
        }
      ],
      "sys_tags":[]
    }
  ],
  "total_count":1
}

```

Status Codes

- Normal

Status Code	Description
200	Resources are listed.

- Abnormal

Status Code	Description
400	Invalid tag parameter.
401	Certification failed.
403	Authentication failed.
404	Resource not found.
500	System error.

5.3.5 Querying the Number of Resources by Tag

Function

This API is used to query the number of resources by tag.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

- POST /v3/sfs/tms/{project_id}/file-systems/resource-instances/count
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	The project ID. For details, see Obtaining a Project ID .

Request Parameters

Table 5-34 Request header parameters

Parameter	Mandatory	Type	Description
Content-type	Yes	String	The MIME type of the request body. Example: application/json
X-Auth-Token	No	String	The user token.

Table 5-35 Request body parameters

Parameter	Mandatory	Type	Description
without_any_tag	No	boolean	Excludes any tags. If this parameter is set to true , all resources without tags are queried. In this case, the tags field is ignored. If this parameter is set to false or not specified, it does not take effect, meaning that all resources are returned or resources are filtered by tags or matches .
tags	No	List<tag>	Includes tags. A maximum of 20 tags can be specified. Each tag key can have a maximum of 20 tag values. A tag value can be empty but the structure cannot be missing. Tag values of the same key must be unique. Tag keys must be unique, and tag values of the same key must be unique. The response returns resources containing all tags in this list. Keys in this list are in the AND relationship and values in each key-value structure are in the OR relationship. If this parameter is not specified, all resources will be returned. For details, see Table 5-30 .

Parameter	Mandatory	Type	Description
sys_tags	No	List<tag>	<p>Includes system tags. This parameter is only available to users with the op_service permission.</p> <p>It cannot be used with filtering criteria without_any_tag and tags at the same time.</p> <p>It contains only one tag structure currently.</p> <p>The key is fixed at _sys_enterprise_project_id.</p> <p>The value is the ID of an enterprise project.</p> <p>The key contains only one value. Value 0 indicates the default enterprise project.</p> <p>For details, see Table 5-30.</p>
matches	No	List<match>	<p>The fields to be matched. The key in match is a dictionary value fixed at resource_name, meaning that the prefix search is performed based on the value of the key. It will be extended later.</p> <p>For details, see Table 5-31.</p>

Response Parameters

Table 5-36 Response body parameter

Parameter	Mandatory	Type	Description
total_count	Yes	Integer	The total number of records.

Example Request

Querying the number of resources using project ID **c80a2157ba1d46c0825265947342077c**:

```
POST https://{endpoint}/v3/sfs/tms/c80a2157ba1d46c0825265947342077c/file-systems/resource-instances/count
```

Request body example:

```
{
  "tags":[
    {
```

```

    "key": "key1",
    "values": [
      "value1",
      "value2"
    ]
  },
  {
    "key": "key2",
    "values": [
      "value1",
      "value2"
    ]
  }
],
"matches": [
  {
    "key": "resource_name",
    "value": "resource1"
  }
],
"without_any_tag": "true"
}

```

Example Response

```

{
  "total_count": 1
}

```

Status Codes

- Normal

Status Code	Description
200	The number of resources is queried.

- Abnormal

Status Code	Description
400	Invalid tag parameter.
401	Certification failed.
403	Authentication failed.
404	Resource not found.
500	System error.

5.3.6 Querying Tags in a Project

Function

This API is used to query tags of all resources owned by a tenant in a project.

Authorization

Each account has all of the permissions required to call all APIs, but IAM users must have the required permissions specifically assigned. For the specific permissions required, see [Permissions Policies and Supported Actions](#).

URI

- GET /v3/sfs/tms/{project_id}/file-systems/tags
- Parameter description

Parameter	Mandatory	Type	Description
project_id	Yes	String	The project ID. For details, see Obtaining a Project ID .

Request Parameters

Table 5-37 Request header parameters

Parameter	Mandatory	Type	Description
Content-type	Yes	String	The MIME type of the request body. Example: application/json
X-Auth-Token	No	String	The user token.

Response Parameters

Table 5-38 Response body parameter

Parameter	Mandatory	Type	Description
tags	Yes	List< tag >	The tag list. For details, see Table 5-30 .

Example Request

Querying tags in the project whose ID is **c80a2157ba1d46c0825265947342077c**:

```
GET https://{endpoint}/v3/sfs/tms/c80a2157ba1d46c0825265947342077c/file-systems/tags
```

Example Response

```
{
  "tags": [
    {
      "key": "key1",
```

```
    "values":[
      "value1",
      "value2"
    ],
    {
      "key":"key2",
      "values":[
        "value1",
        "value2"
      ]
    }
  ]
}
```

Status Codes

- Normal

Status Code	Description
200	Tags in a project are queried.

- Abnormal

Status Code	Description
400	Invalid tag parameter.
401	Certification failed.
403	Authentication failed.
404	Resource not found.
500	System error.

6 Permissions Policies and Supported Actions

6.1 Introduction

This section describes fine-grained permissions management for your SFS resources. If your Huawei Cloud account does not need individual IAM users, then you may skip over this section.

By default, new IAM users do not have permissions assigned. You need to add a user to one or more groups, and attach permissions policies or roles to these groups. Users inherit permissions from the groups to which they are added and can perform specified operations on cloud services based on the permissions.

You can grant users permissions by using roles and policies. Roles are a type of coarse-grained authorization mechanism that defines permissions related to user responsibilities. Policies define API-based permissions for operations on specific resources under certain conditions, allowing for more fine-grained, secure access control of cloud resources.

NOTE

You can use policies to allow or deny access to specific APIs.

Each account has all the permissions required to call all APIs, but IAM users must be assigned the required permissions. 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 wants to query ECSs using an API, the user must have been granted permissions that allow the **ecs:servers:list** action.

Supported Actions

SFS provides system-defined policies that can be directly used in IAM. You can also create custom policies and use them to supplement system-defined policies, implementing more refined access control. Actions supported by policies are specific to APIs. See the following common concepts related to policies:

- **Permissions:** Statements in a policy that allow or deny certain operations.
- **APIs:** REST APIs that can be called by a user who has been granted specific permissions.
- **Actions:** Specific operations that are allowed or denied.
- **Related actions:** Actions on which a specific action depends to take effect. When assigning permissions for the action to a user, you also need to assign permissions for the related actions.
- **IAM projects/Enterprise projects:** Authorization scope of a custom policy. A custom policy can be applied to IAM projects or enterprise projects or both. Policies that contain actions for both IAM and enterprise projects can be used and applied for both IAM and Enterprise Management. Policies that only contain actions for IAM projects can be used and applied to IAM only. For details about the differences between IAM and enterprise projects, see [What Are the Differences Between IAM and Enterprise Management?](#)

SFS supports the following actions that can be defined in custom policies:

General-Purpose File System Management: includes actions related to file system management APIs, such as the APIs used for creating a file system, listing file systems, and deleting a file system.

ACL Management: includes actions related to ACL management APIs, such as the APIs used for configuring, obtaining, and deleting a file system ACL.

Tag Management: includes actions related to tag management APIs, such as the APIs for batch adding tags to a resource, batch deleting tags from a resource, querying tags of a resource, querying resources by tag, querying the number of resources by tag, and querying tags in a project.

6.2 Supported Actions

NOTE

The check mark (√) and cross symbol (x) indicate that an action takes effect or does not take effect for the corresponding type of projects.

General-Purpose File System Management

Permission	API	Action	IAM Project (Project)	Enterprise Project (Enterprise Project)
Creating a General-Purpose File System	PUT /{file-system-name}	sfs3:fileSystem:createFileSystem	×	√
Listing General-Purpose File Systems	GET /	sfs3:fileSystem:listFileSystems	×	√

Permission	API	Action	IAM Project (Project)	Enterprise Project (Enterprise Project)
Deleting a General-Purpose File System	DELETE /{file-system-name}	sfs3:fileSystem:deleteFileS ystem	×	√

ACL Management

Permission	API	Action	IAM Project (Project)	Enterprise Project (Enterprise Project)
Configuring an ACL for a General-Purpose File System	PUT /{file-system-name}?sfsacl	sfs3:fileSystem:putACL	×	√
Obtaining an ACL for a General-Purpose File System	GET /{file_system_name}?sfsacl	sfs3:fileSystem:getACL	×	√
Deleting an ACL from a General-Purpose File System	DELETE /{file_system_name}?sfsacl	sfs3:fileSystem:deleteACL	×	√

Tag Management

Table 6-1 Tag management actions

Permission	API	Action	IAM Project (Project)	Enterprise Project (Enterprise Project)
Batch Adding Tags to a Resource	POST /v3/sfs/tms/{project_id}/file-systems/{resource_id}/tags/create	sfs3:tags:tagResource	×	√
Batch Deleting Tags from a Resource	POST /v3/sfs/tms/{project_id}/file-systems/{resource_id}/tags/delete	sfs3:tags:unTagResource	×	√
Querying Tags of a Resource	GET /v3/sfs/tms/{project_id}/file-systems/{resource_id}/tags	sfs3:tags:listTagsForResource	×	√
Querying Resources by Tag	POST /v3/sfs/tms/{project_id}/file-systems/resource-instances/filter	sfs3:tags:listResourcesByTag	×	×
Querying the Number of Resources by Tag	POST /v3/sfs/tms/{project_id}/file-systems/resource-instances/count	sfs3:tags:listResourcesByTag	×	×
Querying Tags in a Project	GET /v3/sfs/tms/{project_id}/file-systems/tags	sfs3:tags:listTags	×	×

7 Appendix

7.1 Status Codes

If an error code starting with **APIGW** is returned after you call an API, rectify the fault by referring to the instructions provided in [Error Codes](#).

- Normal

Returned Value	Description
200 OK	Specifies the normal response for the GET and PUT operations.
201 Created	Specifies the normal response for the POST operation.
202 Accepted	The request has been accepted for processing.
204 No Content	Specifies the normal response for the DELETE operation.

- Abnormal

Returned Value	Description
400 Bad Request	The server failed to process the request.
401 Unauthorized	You must enter a username and the password to access the requested page.
403 Forbidden	Access to the requested page is forbidden.
404 Not Found	The requested page was not found.
405 Method Not Allowed	You are not allowed to use the method specified in the request.

Returned Value	Description
406 Not Acceptable	The response generated by the server could not be accepted by the client.
407 Proxy Authentication Required	You must use the proxy server for authentication. Then the request can be processed.
408 Request Timeout	The request timed out.
409 Conflict	The request could not be processed due to a conflict.
500 Internal Server Error	The request is not completed because of a service error.
501 Not Implemented	The request is not completed because the server does not support the requested function.
502 Bad Gateway	The request is not completed because the server receives an invalid response from an upstream server.
503 Service Unavailable	The request is not completed because the service is unavailable.
504 Gateway Timeout	A gateway timeout error occurred.

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 `3xx`, `4xx` or `5xx` is returned. The response body contains the specific error code and information. If you are unable to identify the cause of an error, contact customer service and provide the error code so that we can help you solve the problem as soon as possible.

Error Response Syntax

When an error occurs, the response header information contains:

- Content-Type: application/xml
- HTTP error status code `3xx`, `4xx`, or `5xx`

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-file-system/object</Resource>
```

```
<RequestId>001B21A61C6C0000013402C4616D5285</RequestId>
<HostId>RkRCRDJENDc5MzdGQkQ4OUY3MTI4NTQ3NDk2Mjg0M0FB
QUFBQUFBYmJiYmJiYmJD</HostId>
</Error>
```

Table 7-1 describes the meaning of each element.

Table 7-1 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 7-2 .
Message	Error details in the XML response body. For details about error messages, see Table 7-2 .
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	File system 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 SFS encounters an error when processing a request, a response containing the error code and description will be returned. [Table 7-2](#) shows the General-Purpose File System error codes.

Table 7-2 Error codes

Status Code	Error Code	Error Message	Solution
301 Moved Permanently	PermanentRedirect	The requested file system can be accessed only through the specified address. Send subsequent requests to the address.	Send the request to the returned redirection address.
301 Moved Permanently	WebsiteRedirect	The website request lacks bucketName .	Put the file system name in the request and try again.

Status Code	Error Code	Error Message	Solution
307 Moved Temporarily	TemporaryRedirect	Temporary redirection. If the DNS is updated, the request is redirected to the file system.	The system automatically redirects the request or sends the request to the redirection address.
400 Bad Request	BadDigest	The object content MD5 value specified by the client is inconsistent with that received by the system.	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	IllegalLocationConstraintException	A request without Location is sent for creating a file system in a non-default region.	Send the file system creation request to the default region, or send the request with the Location of a non-default region.
400 Bad Request	InvalidArgument	Invalid parameter.	Modify the parameters according to the error details in the message body.
400 Bad Request	InvalidBucket	The file system to be accessed does not exist.	Try again with another file system name.
400 Bad Request	InvalidBucketName	The file system name in the request is too long or contains special characters that are not allowed.	Try again with another file system name.
400 Bad Request	InvalidLocationConstraint	The specified Location in the file system creation request is invalid or does not exist.	Correct the Location in the file system creation request.

Status Code	Error Code	Error Message	Solution
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 parameters 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	KeyTooLongError	The provided key is too long.	Use a shorter key.
400 Bad Request	MalformedError	The XML format in the request is incorrect.	Use the correct XML format to retry.
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	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.

Status Code	Error Code	Error Message	Solution
400 Bad Request	TooManyBuckets	You have attempted to create more file systems than allowed.	Delete some file systems and try again.
400 Bad Request	TooManyWrongSignatures	The request is rejected due to high-frequency errors.	Replace the Access Key and try again.
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	ContentSHA256Mismatch	The object's SHA-256 value calculated by the client is different from that calculated by the server.	Check whether the SHA-256 value calculated by the client is correct.
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	DeregisterUser	The user has been deregistered.	Top up or re-register.
403 Forbidden	InArrearOrInsufficientBalance	The subscriber owes fees or the account balance is insufficient, and the subscriber does not have the permission to perform an operation.	Top up.
403 Forbidden	InvalidAccessKeyId	The access key ID provided by the customer does not exist in the system.	Provide a correct access key ID.

Status Code	Error Code	Error Message	Solution
403 Forbidden	RequestTimeTooSkewed	<p>There was a large time offset between the OBS server time and the time when the client initiated a request.</p> <p>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.</p>	<p>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.</p>
403 Forbidden	SignatureDoesNotMatch	<p>The provided signature does not match the signature calculated by the system.</p>	<p>Check your secret access key and signature algorithm. For details, see Why Don't the Signatures Match?</p>
403 Forbidden	VirtualHostDomainRequired	<p>Virtual hosting access domain name is not used.</p>	<p>Use the virtual hosting access domain name. For details, see Constructing a Request.</p>
403 Forbidden	Unauthorized	<p>The user has not been authenticated in real name.</p>	<p>Authenticate the user's real name and try again.</p>
404 Not Found	NoSuchBucket	<p>The specified file system does not exist.</p>	<p>Create a file system and perform the operation again.</p>
404 Not Found	NoSuchLifecycleConfiguration	<p>The requested lifecycle rule does not exist.</p>	<p>Configure a lifecycle rule first.</p>
405 Method Not Allowed	MethodNotAllowed	<p>The specified method is not allowed against the requested resource.</p> <p>The message "Specified method is not supported." is returned.</p>	<p>The method is not allowed.</p>

Status Code	Error Code	Error Message	Solution
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 file system name already exists. The file system namespace is shared by all users.	Try again with another file system name.
409 Conflict	BucketAlreadyOwnedByYou	Your previous request for creating the named file system succeeded and you already own it.	No further action is required.
409 Conflict	BucketNotEmpty	The file system you tried to delete is not empty.	Delete the objects in the file system and then delete the file system.
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.
500 Internal Server Error	InternalServerError	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.


```
{
  "domain_id": "65382450e8f64ac0870cd180d14e684b",
  "is_domain": false,
  "parent_id": "65382450e8f64ac0870cd180d14e684b",
  "name": "project_name",
  "description": "",
  "links": {
    "next": null,
    "previous": null,
    "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
  },
  "id": "a4a5d4098fb4474fa22cd05f897d6b99",
  "enabled": true
},
"links": {
  "next": null,
  "previous": null,
  "self": "https://www.example.com/v3/projects"
}
}
```

Obtain the Project ID from the Console

To obtain a project ID from the console, perform the following operations:

1. Log in to the management console.
2. Click the username and select **My Credentials** from the drop-down list.

On the **API Credentials** page, view the project ID in the project list.

Figure 7-1 Viewing the project ID

