Oracle ZFS Storage Appliance Object API Guide for Amazon S3 Service Support, Release OS8.8.x

F13774-06
November 2023

Oracle ZFS Storage Appliance Object API Guide for Amazon S3 Service Support, Release OS8.8.x,

F13774-06

Primary Author: Heidi Hall

Contents

1 Getting Started with the Oracle ZFS Storage Appliance S3 Object API Service

2 Working with the Oracle ZFS Storage Appliance S3 Object API Service

3 S3 Object API Operation Command Reference

4 S3 Client Error Handling Reference

1

Getting Started with the Oracle ZFS Storage Appliance S3 Object API Service

The Oracle ZFS Storage Appliance S3 Object API Service enables Amazon S3 clients and applications to store content on an Oracle ZFS Storage Appliance filesystem. The following sections provide information to help you get started with using the Oracle ZFS Storage Appliance S3 Object API Service.

Related Information

- Oracle ZFS Storage Appliance RESTful API Guide, Release OS8.8.x
- Using Oracle Cloud Infrastructure Object Storage Classic

Preparing S3 API Compatible Clients

The Oracle ZFS Storage Appliance S3 Object API Service supports the following Amazon S3 compatible clients.

> **Note:**
>
> Other S3 compatible clients might work but have not been tested.

- s3cmd
- Boto and Boto3
- CloudBerry
- Cyberduck
- JetS3t Cockpit

s3cmd

This topic provides example information to help you install, configure, and use the `s3cmd` command-line tool. For additional information about this tool, refer to the product documentation listed under "Related Information" at the end of this section.

Installation Example: s3cmd Client

In the following example, the GitHub address is used to clone and install the `s3cmd` client:

```
git clone https://github.com/s3cmd
cd s3cmd
python setup.py install
```

Configuration Example: Create and Configure the Default Configuration File

A default configuration file eliminates the need to provide an `s3cfg` configuration file for each client session, which must be specified in the command line. To work with the Oracle ZFS Storage Appliance S3 Object API Service, the configuration file must have specific properties set. Use the following examples to create and configure the default configuration file.

Create a default configuration file in the user's home directory by entering the following command:

```
s3cmd --configure
```

For a Microsoft Windows system, see About the s3cmd configuration file for the default configuration file name and its location.

Leave all prompts empty except for the last two:

```
Test access with supplied credentials? [Y/n] n

Save settings? [y/N] y
Configuration saved to '/Users/<user>/.s3cfg'
```

Next, configure the following properties in the default configuration file:

- `access_key`

  This property points to the key required for S3 Authentication. For additional details, see Authentication for S3 API.

- `host_base`

  This property points to the network address of the Oracle ZFS Storage Appliance system and the share object store that was enabled for the S3 service.

- `host_bucket`

  This property must be empty.

- `secret_key`

  This property points to the secret key generated for S3 authentication.

- `use_https`

This property is for disabling or enabling the HTTPS service. For instance, set the `use_https` property to `False` to disable the HTTPS service, or set it to `True` to enable the HTTPS service.

Example:

```
cat .s3cfg

[default]
access_key =
             your_access_key-ID

host_base =
             hostname.example.com/s3/v1/export/S3_enabled_share

host_bucket =
secret_key =
             your_secret key
use_https = False
```

Usage Examples: s3cmd Client

The following examples show interaction with the Oracle ZFS Storage Appliance S3 Object API Service. Additional usage information about the `s3cmd` command-line tool is available at the S3 Tools website. For more usage information, see

List all buckets:

```
s3cmd -c .s3cfg ls
```

Create a new bucket named `new_bucket`:

```
s3cmd -c .s3cfg mb s3://new_bucket
```

Upload file `abc.txt` to `new_bucket`:

```
s3cmd -c .s3cfg put abc.txt s3://new_bucket
```

List all objects in `new_bucket`:

```
s3cmd -c .s3cfg ls s3://new_bucket
```

Download object `abc.txt` from `new_bucket` to a new file `abc.2.txt`:

```
s3cmd -c .s3cfg get s3://new_bucket/abc.txt abc.2.txt
```

Delete object `abc.txt` from `new_bucket`:

```
s3cmd -c .s3cfg del s3://new_bucket/abc.txt
```

Delete bucket `new_bucket`:

```
s3cmd -c .s3cfg rb s3://new_bucket
```

Run command in debug mode using option `--debug`:

```
s3cmd -c .s3cfg cmd parameters --debug
```

Related Information

- See Installation of s3cmd package on the GitHub website for `s3cmd` installation information.
- See the S3 Tools website for Amazon S3 tools and usage.
- See Amazon S3 Compatibility API in Oracle Cloud Infrastructure documentation.

Boto and Boto3

The following sections provide example information to help you install, configure, and use the Boto and Boto3 command-line tools. For additional information about these tools, refer to the product documentation listed under "Related Information" at the end of this section.

> **Note:**
>
> Boto and Boto3 are client functions in Amazon Web Services (AWS) Software Development Kit (SDK) for Python. Boto3 is the next generation of Boto and is available for general use.

Installation Example: Boto and Boto3

Install the latest version of Boto or Boto3 using `pip`, for example:

```
pip install boto

pip install boto3
```

Configuration Example: Boto and Boto3

Prior to using Boto (or Boto3), you need to set up authentication credentials. Authentication credentials can be configured in multiple ways. For instance, you can pass authentication credentials as parameter methods, environmental variables, or within a file such as a shared credentials file or an AWS configuration file. The following example defines the authentication credentials in an AWS configuration file.

```
cat ~/.aws/credentials

[default]

aws_access_key_id = YOUR_ID

aws_secret_access_key = YOUR_SECRET
```

Usage Examples: Boto and Boto3

The following Boto and Boto3 examples show interaction with the Oracle ZFS Storage Appliance S3 Object API Service.

Boto:

```
#!/usr/bin/env python

import boto
```

```
import boto.s3.connection

access_key = 'coma-04042017'
secret_key = 'd0f8c646dc4303930c547b85ef549ce80aa0709f4720436ab8f24afeaebf80b9'

conn = boto.connect_s3(
   aws_access_key_id = coma-04042017,
   aws_secret_access_key =
d0f8c646dc4303930c547b85ef549ce80aa0709f4720436ab8f24afeaebf80b9,
   host = "x4200-85.us.example.com", # Storage appliance host name.
   port = 443,                       # Set to HTTPS port number for HTTPS connection.
   is_secure=True,                   # Set to True for HTTPS connection.
   debug = 2,
   path = "/s3/v1/export/coma",      # Configured share S3 filesystem.
   calling_format=boto.s3.connection.OrdinaryCallingFormat()
        )

bucket = conn.create_bucket('new_bucket')
```

Boto3:

```
#!/usr/bin/env python

import boto3

access_key = 'open-sesame'
secret_key = 'ddc37fc17a3837dcb4757612cefa8f4cf0be9508b51b6d6a1087c24e3d7da95f'

b3_session = boto3.Session(aws_access_key_id=access_key,
                           aws_secret_access_key=secret_key,
                           region_name='lalaland')

b3_client = b3_session.client('s3', endpoint_url="https://192.0.2.0/s3/v1/export/
mystore")

bucket = b3_client.create_bucket(Bucket="newbucket")
```

Related Information

- See the GitHub boto website for the latest versions of Boto or Boto3.
- See the Boto 3 Documentation for installation, configuration, and use with Amazon S3.

CloudBerry

The following sections provide example information to help you install, configure, and use CloudBerry backup tools. For additional information about this tool, refer to the CloudBerry website.

Installation: CloudBerry

Download and install the appropriate version from the CloudBerry website.

Configuration Example: CloudBerry

The following CloudBerry Client configuration is based on Windows 10. To configure an S3 compatible account, perform the following:

1. In the **Select Cloud Storage** dialog box, choose **S3 Compatible**.
2. In the **S3 Storage Account** dialog box, do the following:

- Enter the access key, secret key, and the service point.

  Note that the service point is provided on the Oracle ZFS Storage Appliance share protocol. For example:

  `https://appliance_hostname.example.com/s3/v1/export/S3_enabled_sharename/`

  Do not omit the trailing `/`.

- In the **Bucket name** field, specify a new bucket name or an existing one.
- Click **Advanced Settings** and clear the **SSL** check box.

Usage: CloudBerry

Start backing up files by (1) selecting the S3 compatible account that you just created and (2) following the CloudBerry backup plan and restore plan wizard.

Cyberduck

The following sections provide example information to help you install, configure, and use the Cyberduck S3 compatible browser. For additional information about this S3 compatible browser, refer to the Cyberduck website.

Installation: Cyberduck

Download and install the appropriate version from the Cyberduck website.

Configuration Example: Cyberduck

The configuration is based on a Cyberduck client on Windows 10. The default Cyberduck S3 profile does *not* support the Oracle ZFS Storage Appliance S3 API. An Oracle ZFS Storage Appliance Cyberduck profile template must be created manually. See the following examples:

Oracle ZFS Storage Appliance S3 (HTTPS) Cyberduck Profile:

The Oracle ZFS Storage Appliance S3 (HTTPS) Cyberduck profile uses S3 signature v4 (recommended version).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "https://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
    <dict>
        <key>Protocol</key>
        <string>s3</string>
        <key>Vendor</key>
        <string>s3-https</string>
        <key>Scheme</key>
        <string>https</string>
        <key>Description</key>
        <string>S3 (HTTPS)</string>
        <key>Default Port</key>
        <string>443</string>
        <key>Hostname Configurable</key>
        <true/>
        <key>Port Configurable</key>
        <true/>
        <key>Context</key>
        <string>/s3/v1/export/SHARE</string>
```

```
    </dict>
</plist>
```

Oracle ZFS Storage Appliance AWS2 Signature Version (HTTPS) Cyberduck Profile:

The Oracle ZFS Storage Appliance AWS2 Signature Version (HTTPS) Cyberduck profile uses S3 signature v2.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "https://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
    <dict>
        <key>Protocol</key>
        <string>s3</string>
        <key>Vendor</key>
        <string>s3-aws2-https</string>
        <key>Scheme</key>
        <string>https</string>
        <key>Description</key>
        <string>S3 AWS2 Signature Version (HTTPS)</string>
        <key>Default Port</key>
        <string>443</string>
        <key>Hostname Configurable</key>
        <true/>
        <key>Port Configurable</key>
        <true/>
        <key>Authorization</key>
        <string>AWS2</string>
        <key>Context</key>
        <string>/s3/v1/export/SHARE</string>
    </dict>
</plist>
```

In both templates, you need to change "SHARE" in the "Context" keyword to the share that has S3 service enabled. For example, you can change it to `/s3/v1/export/` *`S3_enabled_share`* .

```
<key>Context</key>

<string>/s3/v1/export/S3_enabled_share</string>
```

Cyberduck GUI-based Configuration:

1. Double-click the profile file you just created. An S3 configuration dialog box appears.
2. In the S3 configuration dialog box, specify the server address that points to the appliance. For example, *appliance_hostname.example*.com
3. In the S3 configuration dialog box, click **Close**. An S3 bookmark for the Oracle ZFS Storage Appliance system appears.
4. Double-click the bookmark. A Login dialog box appears.
5. In the Login dialog box, specify the required **Access Key** and **Secret Keys**, then click **Login**.

   You have successfully logged in to your Cyberduck S3 account.

Usage: Cyberduck

Cyberduck works as a file explorer, enabling you to browse the Amazon S3 storage service, such as a hard disk.

JetS3t Cockpit

The following sections provide example information to help you install, configure, and use the JetS3t Cockpit application. For additional information about this application, refer to the product documentation listed under "Related Information" at the end of this section.

> **Note:**
>
> JetS3t is a free, open-source Java toolkit and application suite for Amazon S3, Amazon CloudFront, and Google Storage for Developers. JetS3t Cockpit is a graphical Java application for viewing and managing AWS S3 content.

Installation: JetS3t Cockpit

Download the zip package from the GitHub JetS3t site. Unzip the package. Download Java JRE and JDK to use appliance and library.

Configuration Example: JetS3t Cockpit

To configure JetS3t Cockpit on Windows 10, follow these steps:

1. Under `jets3t package root`, look for "`RestS3Service`, then edit `configs/jets3t` properties as follows:
   - In the `s3service.s3-endpoint=`*`property`* , specify the network name for the appliance connection.
   - Specify the shared repository that is enabled for S3 by inserting the following line immediately after the line `s3service.s3-endpoint=`:

     ```
     s3service.s3-endpoint-virtual-path=/s3/v1/export/S3_enabled_share
     ```

     Replace *S3_enabled_share* with the name of the S3 enabled share.
   - Change the property value for `buckets=` from `false` to `true`.

```
###
# RestS3Service
###

s3service.https-only=true     # if required, change from 'true' to 'false'

s3service.s3-endpoint=hostname.example.com  # Storage appliance host name.
s3service.s3-endpoint-virtual-path=/s3/v1/export/S3_enabled_share # insert
this line.
s3service.s3-endpoint-http-port=80
s3service.s3-endpoint-https-port=443
s3service.disable-dns-buckets=true   # change from 'false' to 'true'.

s3service.default-bucket-location=US

s3service.enable-storage-classes=true
s3service.default-storage-class=STANDARD
```

2. Browse to the `bin` directory and double-click `cockpit.bat`. (`/cockpit.sh` if in Linux)

The JetS3t Cockpit Login dialog box appears.

3. In the JetS3t Cockpit Login dialog box, click the **Direct Login** tab, and enter the **Access Key** and **Secret Keys**, then click **Login**.

   You have successfully logged in to the JetS3t Cockpit account.

Usage: JetS3t Cockpit

Use the JetS3t Cockpit interface like a file explorer. If you want to use the `jets3t` library, refer to the JetS3t programmer guide and code samples for more information.

Related Information

See the JetS3t site site for documentation, including code samples.

Protocol Ports Requirements for Amazon S3

The Oracle ZFS Storage Appliance S3 Object API operates on the following ports:

Table 1-1 S3 Protocol Ports

Port Number	Protocol
443	HTTPS
80	HTTP

> **Note:**
>
> HTTP(S) requests sent to Oracle ZFS Storage Appliance that start with `/s3/` are intended for the Oracle ZFS Storage Appliance S3 API.

Interoperability With Other Data Access Protocols

The following list defines interoperability limitations between the S3 protocol and other data access protocols:

> **Note:**
>
> Other data access protocols include SWIFT API, NFS, SMB, WebDAV, and so on.

- **Container and Bucket** – A Swift container will appear as a bucket through S3 and a bucket as a container through Swift.
- **S3 Access** – Any change made by another protocol to the Oracle ZFS Storage Appliance access control lists (ACLs) will have a direct effect on S3 access. The S3 Object API automatically ignores the Oracle ZFS Storage Appliance ACLs that cannot be mapped to S3 ACLs. Access to an object or bucket is dependent on the appliance ACL associations. Do *not* change the appliance ACLs on a filesystem or any of its contents via another protocol.

- **Read and Write Mode** – It is advisable to keep all other protocols in read-only mode when S3 is in read/write mode. If an object is edited by another protocol, users might see unexpected results the next time a request is made to retrieve the object. For instance, an edit from another protocol will be detected and checksums will be recalculated the next time the object is accessed using the S3 API.
- **DLO and Versioning Operations** – Complex operations like Dynamic Large Object (DLO) and versioning are *not* supported for interoperability between SWIFT and S3. These operations, which are very protocol specific, will not work across object protocols. DLO and versioned objects will appear like normal objects from the other protocol. It is advisable to maintain caution when dealing with such objects as accidental deletion or editing could break the functionality for the other protocol.
- **User-Defined Metadata** – User-defined metadata can be stored using either SWIFT or S3. User-defined metadata stored by one of the object protocols can then be retrieved by the other protocol. Note that S3 only allows setting of user-defined metadata for objects, whereas Swift allows it for containers and objects. The user metadata for containers will not be visible from S3.
- **Bucket Access** – Only the object and bucket creator (owner) is permitted access to the object and bucket from the other protocols.

S3 API Usage Guidelines

The following table describes usage guidelines for naming or mapping S3 API bucket directories and objects.

Table 1-2 Usage Guidelines: S3 API

Number	Guideline Description
1	The S3 API creates subdirectories in bucket for each `/` it encounters in the object's name. For example, when the user uploads an object called `accounting/billing.pdf`, a directory called `accounting` is created in the bucket.
2	Objects that are mapped to the Oracle ZFS Storage Appliance directories, such as an object with names that end with `/`, cannot store content for themselves. However, content can be created in them. For example: object `foo/` cannot have any data associated with it but object `foo/bar` can.
3	Any file name not permitted by the Oracle ZFS Storage Appliance filesystem is also not permitted while creating objects in the S3 API. For example, objects names with any of the following characters are not permitted: `//`, `/`, `..` Object names containing double `//` are also not permitted (`foo//bar`).
4	Object names that are relative paths are not permitted. For example, `../../../foo` is not permitted.
5	Oracle ZFS Storage Appliance limits file and directory names to 255 characters. The same character limit (255) applies to the virtual directory names and object names for the Oracle ZFS Storage Appliance S3 API.

Table 1-2 (Cont.) Usage Guidelines: S3 API

Number	Guideline Description
6	When bucket versioning is either enabled or suspended, take the following naming conventions into consideration: • When versioning is enabled for a bucket, previous versions are saved in the same directory as the current version of the object. • The current version of an object is renamed when a new object or delete marker takes its place. The new name of the current version follows the pattern: *object_name-versionId* . For example, billing.pdf-0001.

Authentication for S3 API

The following sections identify general aspects of the S3 authentication process as it relates to the Oracle ZFS Storage Appliance S3 Object API Service. For detailed S3 authentication information, refer to the following Amazon S3 documentation:

- Signing and Authenticating REST Requests
- Supported Authorization Versions
- Authenticating Requests

Supported Authorization Versions

The Oracle ZFS Storage Appliance S3 Object API Service supports both AWS S3 authentication v2 and v4. Both versions require you to create an access key and an associated secret key for proving your identity to the system.

Signature Version 2 Format:

```
Authorization: AWS AWSAccessKeyId:Signature
```

Signature Version 4 Format (recommended):

```
signature=Hex(HMAC-SHA256(SigningKey, StringtoSign)
```

Authenticating Requests

For all S3 bucket operations and object operations, Amazon uses an authorization header in all requests to provide the authentication information. When a request is made, the secret key is used to generate a signature. This signature along with the access key are sent to the server as part of the HTTP/HTTPS request. The server will retrieve the secret key using the access key and generate its own signature. When the generated signature by the system matches the signature in the request, access is granted to the user who issued the keys.

Related Information

- Authenticating Requests (AWS Signature Version 4)
- Authenticating Requests (AWS Signature Version 2)

Supported and Unsupported S3 API Operations

Refer to the following sections for supported and unsupported S3 API operations on buckets and objects:

- Supported S3 Operations on Buckets
- Supported S3 Operations on Objects
- Unsupported S3 Operations on Buckets
- Unsupported S3 Operations on Objects

Supported S3 Operations on Buckets

The following table identifies bucket operations that are supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-3 Supported S3 Bucket Operations

Operation	Amazon S3 API Documentation
GET Service `GET /`	GET Service
DELETE Bucket `DELETE /`	DELETE Bucket
DELETE Bucket tagging `DELETE /?tagging`	DELETE Bucket tagging
GET Bucket (List Objects) version 2 `GET /?list-type=2`	GET Bucket (List Objects) Version 2
GET Bucket (List Objects) version 1	GET Bucket (List Objects) Version 1
GET Bucket acl `GET /?acl`	GET Bucket acl
GET Bucket tagging `GET /?tagging`	GET Bucket tagging
GET Bucket Object versions `GET /?versions`	GET Bucket Object versions
GET Bucket versioning `GET /?versioning`	GET Bucket versioning
HEAD Bucket `HEAD /`	HEAD Bucket
PUT Bucket `PUT /`	PUT Bucket
PUT Bucket acl PUT /?acl	PUT Bucket acl
PUT Bucket tagging `PUT /?tagging`	PUT Bucket tagging

Table 1-3 (Cont.) Supported S3 Bucket Operations

Operation	Amazon S3 API Documentation
PUT Bucket versioning `PUT /?versioning`	PUT Bucket versioning
GET List Multipart Uploads `GET /?uploads&delimiter=Delimiter&encoding-type=EncodingType&key-marker=KeyMarker&max-uploads=MaxUploads&prefix=Prefix&upload-id-marker=UploadIdMarker`	List Multipart Uploads

Supported S3 Operations on Objects

The following table identifies object operations that are supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-4 Supported S3 Object Operations

Operation	Amazon S3 API Documentation
DELETE Object: `DELETE /ObjectName`	DELETE Object
Delete Multiple Objects `POST /?delete`	Delete Multiple Objects
GET Object `GET /ObjectName`	GET Object
GET Object ACL `GET /ObjectName?acl`	GET Object ACL
HEAD Object `HEAD /ObjectName`	HEAD Object
OPTIONS Object `OPTIONS /ObjectName`	OPTIONS object If CORS is not enabled on the bucket, Amazon S3 returns a 403 Forbidden response.
POST Object `POST /`	POST Object POST object is done through HTML forms.
PUT Object `PUT /ObjectName`	PUT Object
PUT Object ACL `PUT /ObjectName?acl`	PUT Object acl
PUT Object - Copy `PUT /destinationObject`	PUT Object - Copy
Multipart Upload:	

Table 1-4 (Cont.) Supported S3 Object Operations

Operation	Amazon S3 API Documentation
POST Object Uploads `POST /ObjectName?uploads`	Create Multipart Upload
PUT Object Part `PUT /ObjectName?partNumber=PartNumber&uploadId=UploadId`	Upload Part
PUT Object Part Copy `PUT /ObjectName?partNumber=PartNumber&uploadId=UploadId`	Upload Part Copy
GET List Parts `GET /ObjectName?max-parts=MaxParts&part-number-marker=PartNumberMarker&uploadId=UploadId`	List Parts
POST Complete Multipart Upload `POST /ObjectName?uploadId=UploadId`	Complete Multipart Upload
DELETE Abort Multipart Upload `DELETE /ObjectName?uploadId=UploadId`	Abort Multipart Upload

Unsupported S3 Operations on Buckets

The following table identifies bucket operations that are not supported by the Oracle ZFS Storage Appliance S3 Object API Service, including cross-origin resource sharing (CORS).

Table 1-5 Unsupported S3 Bucket Operations

Unsupported Bucket Operation	Unsupported Bucket Operation
GET Bucket location	GET Bucket accelerate PUT Bucket accelerate
GET Bucket cors PUT Bucket cors PUT Bucket cors DELETE Bucket cors	GET Bucket website PUT Bucket website DELETE Bucket website
PUT Bucket lifecycle DELETE Bucket lifecycle	GET Bucket notification PUT Bucket notification
PUT Bucket policy DELETE Bucket policy	GET Bucket requestPayment PUT Bucket requestPayment
GET Bucket replication PUT Bucket replication DELETE Bucket replication	GET Bucket logging PUT Bucket logging

In addition, feature `PutObjectLockConfiguration` is not supported.

Unsupported S3 Operations on Objects

The following table identifies object operations that are not supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-6 Unsupported S3 Object Operations

Unsupported Object Operation	Unsupported Object Operation
GET Object torrent	POST Object restore

Supported and Unsupported Header Requests

The following topics identify header request behavior for the Oracle ZFS Storage Appliance S3 Object API Service.

- Supported Common Request Headers
- Unsupported Request Headers

Supported Common Request Headers

The following table identifies the common request headers supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-7 Common Supported Request Headers

Supported Request Header	Description	Required
Authorization	The Authorization header field identifies the information required for request signature authentication. For more information, see "The Authentication Header" in Amazon Simple Storage Service Developer Guide.	Yes **Note** - For anonymous requests, this header is not required.
Content-Length	The Content-Length header field represents the length of the message (*without the headers*) according to RFC 2616.	No **Note** - This header is required for PUTs and load XML operations.
Content-MD5	The Content-MD5 header field represents the base64 encoded 128-bit MD5 digest of the message (*without the headers*) according to RFC 1864. This header is used as a message integrity check to verify that the data is the same data that was originally sent.	No

Table 1-7 (Cont.) Common Supported Request Headers

Supported Request Header	Description	Required
Date **or** x-amz-date	These header fields represents the current date and time according to the requester. When you specify the Authorization header, you must specify either the `x-amz-date` or the `Date` header. If you specify both, the value specified for the `x-amz-date` header takes precedence	Yes
Expect	The Expect header field is used only when sending a request body with the property values: *100-continue*. **Note** - When your application uses 100-continue as a property value, the request body is not sent until after it receives an acknowledgment. If the message is rejected based on the headers, the body of the message is not sent.	No
Host	The Oracle ZFS Storage Appliance Host FQDNT header field is required for HTTP 1.1 (**most toolkits add this header automatically**); optional for HTTP/1.0 requests.	Yes
x-amz-content-sha256	When using signature version 4 to authenticate a request, the x-amz-content-sha256 header field provides a hash of the request payload. For more information, see the Amazon documentation Signature Calculations for the Authorization Header: Transferring Payload in a Single Chunk (AWS Signature Version 4). When uploading an object in chunks, you can set the value to STREAMING-AWS4-HMAC-SHA256-PAYLOAD to indicate that the signature covers only headers and that there is no payload. For more information, see the Amazon documentation Signature Calculations for the Authorization Header: Transferring Payload in Multiple Chunks (Chunked Upload) (AWS Signature Version 4).	Yes
x-amz-expected-bucket-owner	The expected bucket owner's account ID. If the bucket is owned by a different account, HTTP status code `403 Forbidden` (access denied) is returned and the request fails.	No

Unsupported Request Headers

The following table lists request headers that are not supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-8 Unsupported Request Headers

Unsupported Request Header	Notes
x-amz-security-token	The Amazon S3 header for generating a temporary security credential is not supported by the Oracle ZFS Storage Appliance S3 API.
x-amz-server-side-encryption-customer-algorithm x-amz-server-side-encryption-customer-key x-amz-server-side-encryption-customer-key-MD5	These Amazon S3 headers for custom server-side encryption are not supported. Oracle ZFS Storage Appliance provides it own data encryption capability that is managed through its BUI, CLI, and REST interfaces. For more details, see NFS Authentication and Encryption Options in *Oracle ZFS Storage Appliance Security Guide, Release OS8.8.x*.

Supported and Unsupported Response Headers

The following topics identify response header behavior for the Oracle ZFS Storage Appliance S3 Object API Service.

Supported Common Response Headers

The following table identifies the response headers that are common to most Amazon S3 responses.

Table 1-9 Common Supported Response Headers

Supported Response Header	Description
Content-Length	The Content-Length header field identifies the response body length in bytes. Type: String Default: None
Content-Type	The Content-Type header field identifies the MIME type of the content. For example, Content-Type: text/html; charset=utf-8 Type: String Default: None
Date	The Date header field identifies the data and time of the S3 response. For example, Wed, 01 Mar 2018 12:00:00 GMT. Type: String Default: None
ETag	The ETag header field identifies a specific version of a resource (a hash of the object). The ETag reflects changes only to the contents of an object, not its metadata. The ETag is an MD5 digest of the object data. Type: String
Server	The Server header field identifies the name of the server that created the response. Type: String Valid Value: Apache
x-amz-delete-marker	The `x-amz-delete-marker` identifies whether the object returned was a delete marker (`true`) or not (`false`). Type: Boolean Valid Values: true | false Default: false
x-amz-request-id	The `x-amz-request-id` is a value that is created by Amazon S3 to uniquely identify a request for troubleshooting purposes. Type: String Default: None

Table 1-9 (Cont.) Common Supported Response Headers

Supported Response Header	Description
x-amz-version-id	When versioning is enabled, the `x-amz-version-id` identifies the version of the object. When versioning is suspended, the version ID is always null. Type: String Valid Values: null | string Default: null
x-amz-tagging-count	When the count is greater than zero, the `x-amz-tagging-count` returns the count of the tags associated with the object. Type: String Default: None

Unsupported Common Response Headers

The following table lists response headers that are not supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-10 Unsupported Response Headers

Unsupported Response Header	Notes
x-amz-restore	The `x-amz-restore` header is not supported by the Oracle ZFS Storage Appliance S3 API Service.
x-amz-replication-status	The `x-amz-replication-status` header is not supported by the S3 API. Oracle ZFS Storage Appliance provides its own remote replication capability, that is managed through its management interfaces (BUI, CLI, REST). For more details, see Remote Replication in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x* .
x-amz-server-side-encryption x-amz-server-side-encryption-aws-kms-key-id x-amz-server-side-encryption-customer-algorithm x-amz-server-side-encryption-customer-key-MD5	These Amazon S3 headers for custom server-side encryption are not supported. Oracle ZFS Storage Appliance provides it own data encryption capability that is managed through its BUI, CLI, and REST interfaces. For more details, see NFS Authentication and Encryption Options in *Oracle ZFS Storage Appliance Security Guide, Release OS8.8.x* .
x-amz-id-2	The `x-amz-id-2` token is not supported by the Oracle ZFS Storage Appliance S3 API Service. Use the `x-amz-request-id` response header value as an alternative solution.

Unsupported Configuration for ZFS Data Features

The following table identifies Oracle ZFS Storage Appliance data features that are not supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Table 1-11 Unsupported Oracle ZFS Storage Appliance S3 Object API Data Features

Unsupported Feature	Notes
Encryption	The configuration of the Oracle ZFS Storage Appliance encryption data feature is not supported by the Oracle ZFS Storage Appliance S3 Object API Service. However, the ZFS encryption data feature is configurable from the BUI, CLI, and REST appliance management interfaces. For details, see NFS Authentication and Encryption Options in *Oracle ZFS Storage Appliance Security Guide, Release OS8.8.x* .
Replication	The configuration of the Oracle ZFS Storage Appliance replication data feature is not supported by the Oracle ZFS Storage Appliance S3 Object API Service. However, the ZFS replication data feature is configurable from the BUI, CLI, and REST appliance management interfaces. For details, see Remote Replication in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x* .
Snapshot	The configuration of the Oracle ZFS Storage Appliance snapshot data feature is not supported by the Oracle ZFS Storage Appliance S3 Object API Service. However, the ZFS snapshot data feature is configurable from the BUI, CLI, and REST appliance management interfaces. For details, see Snapshot Space Management in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x* .

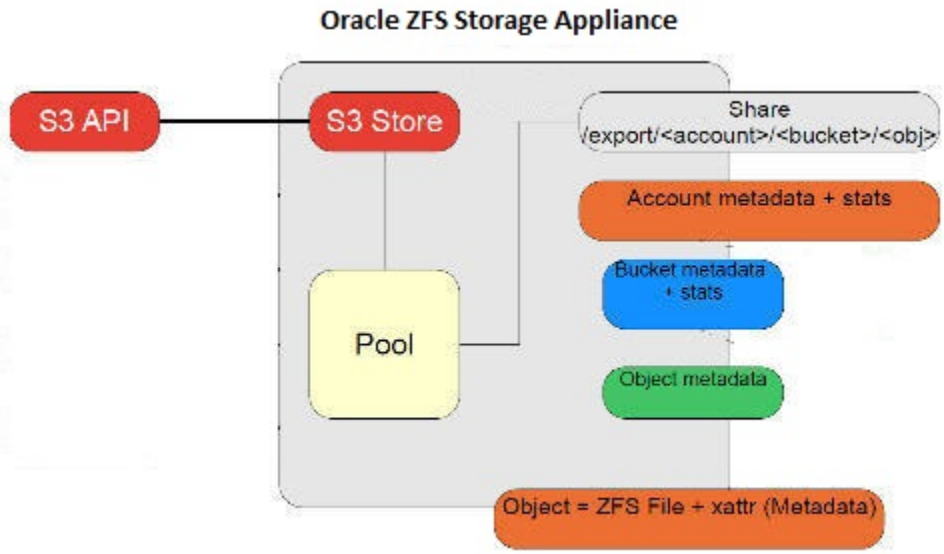
2

Working with the Oracle ZFS Storage Appliance S3 Object API Service

The following sections describe information about accessing resources, controlling access to resources, as well as supported versioning options to protect your data.

Key Concepts and Elements for Accessing Resources

The Oracle ZFS Object Storage Model is a share filesystem, where each share is mapped to a project or tenant. Each share is an account. Each account-share holds buckets and each bucket holds data objects. The S3 API uses these key elements to access the appliance resources. An illustration and description of these key elements follow.



- **Share** – A share is a collection of S3 buckets. In the Oracle ZFS Storage Appliance object store implementation, the exported share name represents the account name, for example: `/export/account_name _share_mount_point`).

> **Note:**
>
> A share is typically associated with a departmental group-type entity. Access to resources within a share are managed by user roles. For more information about appliance user roles, see Configuring Users in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x* .

- **Bucket** – A bucket is a user-definable element in the object data path. It is a container for storing S3 data objects. In the AWS S3 architecture, buckets and objects are known as resources. For additional information about the use and configuration of buckets, see Introduction to Amazon S3.
- **Object** – Objects represent the entities stored in a bucket. Each object within a bucket is uniquely identified by a key name and version ID. On Oracle ZFS Storage Appliance, each object consist of a file and a set of metadata that describes the object.

Making Requests Using the S3 Object API

To send a request message to Oracle ZFS Storage Appliance using the S3 Object API, the request message should include the following entities:

- **Request Type** – The request type states the action to be performed on a resource that is identified in the URI, for example:

  ```
  GET https://appliance:443/s3/v1/export/
                sharename/bucketname/objectname
  ```

  Where:

  – `GET` is the action to be performed on the resource identified in the URI. For a list of supported actions on buckets and objects, see Supported and Unsupported S3 API Operations.

  – The following is the request URI:

  ```
  https://appliance:443/s3/v1/export/
                      sharename/bucketname/objectname
  ```

  The request URI identifies the resources on which to perform actions.

  Where:

  * *appliance* represents the network address or DNS name of the Oracle ZFS Storage Appliance system.
  * *sharename* represents the S3 account name of the share mount point upon which the action is to be performed.
  * *bucketname* represents the name of the bucket upon which the action is to be performed.
  * *objectname* represents the name of the object upon which the action is to be performed.

- **Request-Header Fields** – The request-header fields act as request modifiers that pass additional information to the S3 appliance. The request headers appear after the request line in the message body. For a list of supported request headers, see Supported and Unsupported Header Requests.

Controlling Access to Resources Using S3 ACLs

Access to AWS S3 resources are, by default, private. Only the owner of a resource has access. Optionally, resource owners can grant resource permission to other users by specifying resource-based policy options, such as access control lists (ACLs).

> **Note:**
>
> Other AWS S3 resource-based policy options such as Bucket Policies and User Policies are not supported by the Oracle ZFS Storage Appliance S3 API Service. These AWS S3 policies are similar to the appliance roles that are granted to users. For more information about the Oracle ZFS Storage Appliance roles, see Configuring Users in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x* .

> **Note:**
>
> To support a unified view of the Oracle ZFS Storage Appliance filesystem from other appliance-supported protocols, S3 ACLs are automatically mapped to the equivalent appliance filesystem ACLs. For additional information about Oracle ZFS Storage Appliance ACLs, see Access Control Lists for Filesystems in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x* .

To better understand how to manage appliance resource permissions using AWS S3 ACLs, see the following topics.

- Supported and Unsupported Header Requests
- Setting ACL Policy Permissions in a Request
- Supported Amazon S3 Predefined User Groups
- Supported S3 ACL Permissions

For further details about managing access permissions with AWS S3 ACLs, see Who Is a Grantee?

Specifying S3 ACL Permissions

S3 ACLs enable you to manage access to buckets and objects. Each bucket and object has an ACL attached to it as a subresource. It defines which user or user groups are granted access, as well as the type of access granted. For instance, when a request is received against a resource, the S3 API checks the corresponding ACL to verify that the requester has the necessary access permissions. Each time you create a bucket or object, the S3 API creates a default ACL Policy that grants the resource owner full access control over the newly created resource as shown in the following example.

Example: Default ACL Policy for new bucket or object

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="https://s3.amazonaws.com/doc/2018-05-23/">
  <Owner>
```

```
    <ID>***Owner-Canonical-User-ID***</ID>
    <DisplayName>Bucket Owner Display Name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>***Canonical-User ID or Group URI***</ID>
        <DisplayName>Appliance Grantee's username</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Where:

- `Owner` provides the appliance canonical user ID for the owner that created the bucket.
- `Grant` provides the name of the grantee and the permission granted.

  Note that the default ACL Policy includes only one `Grant` element for the bucket owner. To grant bucket permission to others, you need to add a `Grant` element for each additional user or predefined group. Each `Grant` element must always identify the name of the grantee, as well as the permissions granted.
- `Grantee` provides the name of an individual or predefined group receiving access permission.

  The grantee can either be an authorized appliance user or an S3 predefined group. When granting access to individual appliance users, you need to specify the canonical user ID associated with the appliance user account. When granting permission to an S3 predefined group, you need to specify the predefined group URI. For a list of supported predefined groups, see Supported Amazon S3 Predefined User Groups.
- `Permission` provides the type of access permission that is being granted to the grantee. For a list of supported ACL permissions, see Supported S3 ACL Permissions.

Setting ACL Policy Permissions in a Request

Use one of the following request methods when using S3 APIs (such as `PUT`/`GET`/`DELETE`) to set access policy permissions:

- When creating resources – Set the ACL permissions in the request's HTTP header.
- When editing ACLs associated with existing resources – Set the ACL permissions either in the request's HTTP header or in the request's body.

Supported Amazon S3 Predefined User Groups

The following table describes the supported Amazon S3 predefined user groups.

Table 2-1 Amazon S3 Predefined User Groups

Predefined User Group	Description
All Users Group	The All Users Group is represented by the following URI: `http://acs.amazonaws.com/groups/global/AllUsers` Access permission to this group enables anyone to access the resource. The requests can either be signed (authenticated) or unsigned (anonymous). **Note** - Unsigned requests omit the Authentication header in the request. Anonymous users will be mapped to the user `nobody` in Oracle ZFS Storage Appliance. For security reasons, a resource owner should never grant the All Users Group any of the following permissions: WRITE, WRITE_ACP, or FULL_CONTROL.
Authenticated Users Group	The Authenticated Users Group is represented by the following URI: `http://acs.amazonaws.com/groups/global/AuthenticatedUsers` This group represents all Oracle ZFS Storage Appliance authenticated user accounts. Access permission to this group enables any authenticated user access to the resource. Therefore, when using this group, all requests must be signed (authenticated).

Supported S3 ACL Permissions

The following tables describe the supported permissions for primary and canned ACLs:

- Primary ACL: Grantee Supported Permissions
- Canned ACL: Supported Group Permissions

> **Note:**
>
> You can specify only one canned ACL in a request.

Table 2-2 Primary ACL: Grantee Supported Permissions

Permission	When Granted on Bucket	When Granted on Object
READ	Enables grantee to list the objects in the bucket.	Enables grantee to read the object data and its metadata.
WRITE	Enables grantee to create, overwrite, and delete any object in the bucket.	Not applicable.
READ_ACP	Enables grantee to read the bucket ACL.	Enables grantee to read the object ACL.
WRITE_ACP	Enables grantee to write the ACL for the applicable bucket.	Enables grantee to write the ACL for the applicable object.
FULL_CONTROL	Allows grantee the READ, WRITE, READ_ACP, and WRITE_ACP permissions on the bucket.	Enables grantee the READ, READ_ACP, and WRITE_ACP permissions on the object.

Table 2-3 Canned ACL: Supported Group Permissions

Canned ACL	Applies To	Permissions Added To ACL
private	Bucket and object	Owner gets FULL_CONTROL. No one else has access rights (default).
public-read	Bucket and object	Owner gets FULL_CONTROL. The All Users Group gets READ access.
public-read-write	Bucket and object	Owner gets FULL_CONTROL. The All Users Group gets READ and WRITE access. For security reasons, granting this canned ACL on a bucket is generally not recommended.
authenticated-read	Bucket and object	Owner gets FULL_CONTROL. The Authenticated Users Group gets READ access.
bucket-owner-read	Object	Object owner gets FULL_CONTROL. Bucket owner gets READ access. If you specify this canned ACL when creating a bucket, the appliance S3 API ignores it.
bucket-owner-full-control	Object	Both the object owner and the bucket owner get FULL_CONTROL over the object. If you specify this canned ACL when creating a bucket, the appliance S3 API ignores it.

Protecting Your Data with S3 Object Versioning

To preserve, retrieve, and restore every version of every object stored in your Amazon versioning-enabled S3 bucket, the Oracle ZFS Storage Appliance S3 Object API Service supports the use of the following AWS S3 versioning operations.

- **Versioning-Enabled Bucket Object Operations** – The Oracle ZFS Storage Appliance S3 Object API Service supports the following versioning-enabled bucket operations:
  - Adding Objects to Versioning-Enabled Buckets
  - Listing Objects in a Versioning-Enabled Bucket
  - Retrieving Object Versions
  - Deleting Object Versions
- **Versioning-Suspended Bucket Object Operations** – The Oracle ZFS Storage Appliance S3 Object API Service supports the following versioning-suspended bucket operations:
  - Adding Objects to Versioning-Suspended Buckets
  - Retrieving Object Versions from Versioning-Suspended Buckets
  - Deleting Object Versions from Versioning-Suspended Buckets

For further details about AWS S3 versioning features, see Using Versioning.

3

S3 Object API Operation Command Reference

This reference identifies the operation commands supported by the Oracle ZFS Storage Appliance S3 Object API Service.

Operations on Services

The Oracle ZFS Storage Appliance S3 Object API supports the GET Service operation on services.

GET Service

Returns a list of all buckets owned by the authenticated sender of the request. Anonymous requests cannot list buckets, and you cannot list buckets that you did not create.

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/
```

Request Parameters

The GET Service operation does not support the use of request parameters.

Request Headers

The GET Service operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Response Elements

For a list of supported elements in the XML response for the GET Service operation, see GET Service.

Normal Response Code

```
200
```

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<ListAllMyBucketsResult>
  <Owner>
    <ID>s3_user</ID>
    <DisplayName>s3_user</DisplayName>
  </Owner>
  <Buckets>
    <Bucket>
      <Name>quotes</Name>
      <CreationDate>2006-02-03T16:45:09.000Z</CreationDate>
    </Bucket>
    <Bucket>
      <Name>samples</Name>
      <CreationDate>2006-02-03T16:41:58.000Z</CreationDate>
    </Bucket>
  </Buckets>
</ListAllMyBucketsResult>
```

Operations on Buckets

The Oracle ZFS Storage Appliance S3 Object API supports the following operations on buckets:

- GET Bucket
- GET Bucket ACL
- GET Bucket Object Versioning
- GET Bucket Tagging
- GET Bucket Versioning
- HEAD Bucket
- PUT Bucket
- PUT Bucket ACL
- PUT Bucket Tagging
- PUT Bucket Versioning
- DELETE Bucket
- DELETE Bucket Tagging

GET Bucket

The GET Bucket operation returns some or all (up to 1,000) of the objects in a bucket. You can use the request parameters as selection criteria to return a subset of the objects in a bucket.

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name?list-type=2
```

Request Parameters

For a list of supported request parameters, see GET Bucket (List Objects) Version 2.

Request Headers

The GET Bucket operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Bucket operation does not support the use of request elements.

Response Headers

The GET Bucket operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported elements in the XML response for the GET Bucket operation, see GET Bucket (List Objects) Version 2.

Normal Response Code

`200`

Error Response Code

The GET Bucket operation returns special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example Response

For a list of response examples, see GET Bucket (List Objects) Version 2.

GET Bucket ACL

The GET Bucket ACL operation returns the access control list (ACL) of a bucket. To use `GET` to return the ACL of the bucket, you must have `READ_ACP` access to the bucket.

Syntax Example

`GET https://`*`appliance`*`:443/s3/v1/export/`*`share_mount_point_path/bucket_name`*`?acl`

Request Parameters

The GET Bucket ACL operation does not support the use of request parameters.

Request Headers

The GET Bucket ACL operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Bucket ACL operation does not support the use of request elements.

Response Headers

The GET Bucket ACL operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported elements in the XML response for the GET Bucket ACL operation, see GET Bucket acl.

Normal Response Code

```
200
```

Error Response Code

The GET Bucket API return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET ?acl
HTTP/1.1 200 OK
x-amz-request-id: tx318BC8BC148832E5
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2018 12:00:00 GMT
Content-Length: 124
Content-Type: text/plain
Server: Apache

<AccessControlPolicy>
  <Owner>
    <ID>john</ID>
    <DisplayName>mary</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>jane</ID>
        <DisplayName>Bob</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
```

```
  </AccessControlList>
</AccessControlPolicy>
```

GET Bucket Object Versioning

The GET Bucket Object Versioning operation lists metadata about all of the object versions in a bucket. Additionally, you can use request parameters as selection criteria to return metadata about a subset of all the object versions.

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?versions
```

Request Parameters

For a list of supported request parameters, see GET Bucket Object versions.

Request Headers

The GET Bucket Object Versioning operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Bucket Object Versioning operation does not support the use of request elements.

Response Headers

The GET Bucket Object Versioning operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported response elements, see GET Bucket Object versions.

Normal Response Code

```
200
```

Error Response Code

The GET Bucket Object Versioning API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

> **Note:**
>
> For additional request and response examples, see GET Bucket Object versions.

```
GET /?versions
<?xml version="1.0" encoding="UTF-8"?>

<ListVersionsResult>
    <Name>bucket</Name>
    <Prefix>my</Prefix>
    <KeyMarker/>
    <VersionIdMarker/>
    <MaxKeys>5</MaxKeys>
    <IsTruncated>false</IsTruncated>
    <Version>
        <Key>my-image.jpg</Key>
        <VersionId>003</VersionId>
        <IsLatest>true</IsLatest>
         <LastModified>2018-10-12T17:50:30.000Z</LastModified>
        <ETag>&quot;fba9dede5f27731c9771645a39863328&quot;</ETag>
        <Size>434234</Size>
        <StorageClass>STANDARD</StorageClass>
        <Owner>
            <ID>mary</ID>
            <DisplayName>mary</DisplayName>
        </Owner>
    </Version>
    <DeleteMarker>
        <Key>my-second-image.jpg</Key>
        <VersionId>001</VersionId>
        <IsLatest>true</IsLatest>
        <LastModified>2009-11-12T17:50:30.000Z</LastModified>
        <Owner>
            <ID>jill</ID>
            <DisplayName>jill</DisplayName>
        </Owner>
    </DeleteMarker>
    <Version>
        <Key>my-second-image.jpg</Key>
        <VersionId>002</VersionId>
        <IsLatest>false</IsLatest>
        <LastModified>2009-10-10T17:50:30.000Z</LastModified>
        <ETag>&quot;9b2cf535f27731c974343645a3985328&quot;</ETag>
        <Size>166434</Size>
        <StorageClass>STANDARD</StorageClass>
        <Owner>
            <ID>jill</ID>
            <DisplayName>jill</DisplayName>
        </Owner>
    </Version>
    <DeleteMarker>
        <Key>my-third-image.jpg</Key>
        <VersionId>002</VersionId>
        <IsLatest>true</IsLatest>
        <LastModified>2009-10-15T17:50:30.000Z</LastModified>
        <Owner>
            <ID>moe</ID>
            <DisplayName>moe</DisplayName>
        </Owner>
    </DeleteMarker>
    <Version>
        <Key>my-third-image.jpg</Key>
        <VersionId>001</VersionId>
        <IsLatest>false</IsLatest>
        <LastModified>2009-10-11T12:50:30.000Z</LastModified>
```

```
            <ETag>&amp;quot;772cf535f27731c974343645a3985328&amp;quot;</ETag>
            <Size>64</Size>
            <StorageClass>STANDARD</StorageClass>
            <Owner>
                <ID>moe</ID>
                <DisplayName>moe</DisplayName>
            </Owner>
        </Version>
</ListVersionsResult>
```

GET Bucket Tagging

The GET Bucket Tagging operation returns the tag set associated with the bucket.

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?tagging
```

Request Parameters

The GET Bucket Tagging operation does not support the use of request parameters.

Request Headers

The GET Bucket Tagging operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Bucket Tagging operation does not support the use of request elements.

Response Headers

The GET Bucket Tagging operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported response elements, see GET Bucket tagging.

Normal Response Code

```
200
```

Error Response Code

The GET Bucket Tagging operation does not return errors associated with a bucket. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET /?tagging
HTTP/1.1 200 OK
Date: Wed, 25 Nov 2018 12:00:00 GMT
Connection: close
Server: Apache

<Tagging>
  <TagSet>
     <Tag>
       <Key>Project</Key>
       <Value>Project One</Value>
     </Tag>
     <Tag>
       <Key>User</Key>
       <Value>msmith</Value>
     </Tag>
  </TagSet>
</Tagging>
```

GET Bucket Versioning

The GET Bucket Versioning operation returns the versioning state of a bucket. To retrieve the versioning state of a bucket, you must be the bucket owner. The following versioning states apply to this operation:

- When versioning is enabled on a bucket, the response is as follows:

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

- When versioning is suspended on a bucket, the response is as follows:

  ```
  <VersioningConfiguration> <Status>Suspended</Status> </
  VersioningConfiguration>
  ```

- When versioning is not enabled or suspended on a bucket, the response is as follows:

  ```
  <VersioningConfiguration/>
  ```

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?
versioning
```

Request Parameters

The GET Bucket Versioning operation does not support the use of request parameters.

Request Headers

The GET Bucket Versioning operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Bucket Versioning operation does not support the use of request elements.

Response Headers

The GET Bucket Versioning operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

The following XML response elements are supported:

- `Status`
- `VersioningConfiguration`

For a detailed description of these response elements, see GET Bucket versioning.

Normal Response Code

`200`

Error Response Code

The GET Bucket Versioning operations does not support errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET /?versioning
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2018-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

HEAD Bucket

The HEAD Bucket operation determines whether a bucket exists and if you have permission to access it. To use this operation, you must have permissions to list content in the share target.

Syntax Example

`HEAD https://`*`appliance`*`:443/s3/v1/export/`*`share_mount_point_path/bucketname`*

Request Parameters

The HEAD Bucket operation does not support the use of request parameters.

Request Headers

The HEAD Bucket operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The HEAD Bucket operation does not support the use of request elements.

Response Headers

The HEAD Bucket operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

The HEAD Bucket operation does not support the use of response elements.

Normal Response Code

```
200 OK
```

> **Note:**
>
> The HEAD Bucket operation returns the `200 OK` response code when both the bucket and access permissions to the bucket exist.

Error Response Codes

- `404 Not Found`. This response code is returned when the bucket does not exist.
- `403 Forbidden`. This response code is returned when permissions to access the bucket do not exist.

For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
HEAD /
HTTP/1.1 200 OK
x-amz-request-id: tx32FE2CEB32F5EE25
Date: Fri, 10 2018 21:34:56 GMT
Server: Apache
```

PUT Bucket

The PUT Bucket operation creates a new bucket. To create a bucket with this operation, you must have an Oracle ZFS Storage Appliance user account and a valid

Access Key assigned to your user account. Anonymous requests are never allowed to create buckets. When you create a bucket, you automatically become the bucket owner. Optionally, a bucket owner can grant permissions to other appliance users or predefined groups.

Syntax Example

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucketname
```

Request Parameters

The PUT Bucket operation does not support the use of request parameters.

Request Headers

In addition to supporting common request headers, the PUT Bucket operation also supports headers for specifying canned ACLs and specific ACL access permissions for appliance users and predefined groups. For further details about these request headers, see PUT Bucket.

Request Elements

This implementation of the PUT Bucket operation does not support the use of request elements.

Response Headers

The PUT Bucket operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the PUT Bucket operation does not return response elements.

Error Response Code

The PUT Bucket API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

> **Note:**
>
> For additional request and response examples, see PUT Bucket.

```
PUT HTTP/1.1
x-amz-date: Sat, 07 Apr 2018 00:54:40 GMT
Authorization: authorization string
x-amz-grant-write: id="scobby", id="shaggy"

HTTP/1.1 200
```

PUT Bucket ACL

The PUT Bucket ACL operation sets permissions on an existing bucket using the access control list (ACL).

> **Note:**
>
> For further details about managing access permissions using ACLs, see Controlling Access to Resources Using S3 ACLs.

Syntax Example

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?acl
```

Request Parameters

This implementation of the PUT Bucket ACL operation does not support the use of request parameters.

Request Headers

The PUT Bucket ACL operation supports the following type of request headers.

- Common request headers. For more details, see table "Common Supported Request Headers" in Supported Common Request Headers.
- Canned ACL and Grantee Permission request headers. For more details, see PUT Bucket acl.

Request Elements

The PUT Bucket ACL operation only supports the use of request elements when using a request body to specify an ACL. For a description of supported request elements, see PUT Bucket acl.

Response Headers

The PUT Bucket ACL operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the PUT Bucket ACL operation does not return response elements.

Error Response Code

The PUT Bucket ACL operation returns special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT ?acl HTTP/1.1
Content-Length: 1660
x-amz-date: Thu, 12 Apr 2018 20:04:21 GMT
Authorization: authorization string

<AccessControlPolicy>
  <Owner>
    <ID>bob</ID>
    <DisplayName>bob</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xsi:type="CanonicalUser">
        <ID>bill</ID>
        <DisplayName>bill</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI xmlns="">http://acs.amazonaws.com/groups/global/AllUsers</URI>
      </Grantee>
      <Permission xmlns="">READ</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

PUT Bucket Tagging

The PUT Bucket Tagging operation enables you to add a set of tags to an existing bucket. The bucket owner has this permission by default and can grant this permission to others.

Syntax Example

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?tagging
```

Request Parameters

This implementation of the PUT Bucket Tagging operation does not support the use of request parameters.

Request Headers

The PUT Bucket Tagging operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The PUT Bucket Tagging operation supports the use of request elements. For description of these request elements, see PUT Bucket tagging.

Response Headers

The PUT Bucket Tagging operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the PUT Bucket Tagging operation does not return response elements.

Expected HTTP Response Code

```
204 No Content
```

Error Response Code

This implementation of the PUT Bucket Tagging operation supports the use of the following response errors:

- Special Error. MalformedXMLError where the XML provided does not match the schema.
- Common Errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT ?tagging HTTP/1.1
Content-Length: 1660
x-amz-date: Thu, 12 Apr 2018 20:04:21 GMT
Authorization: authorization string

<Tagging>
  <TagSet>
    <Tag>
      <Key>Project</Key>
      <Value>Project One</Value>
    </Tag>
    <Tag>
      <Key>User</Key>
      <Value>jsmith</Value>
    </Tag>
  </TagSet>
</Tagging>
HTTP/1.1 204 No Content
```

PUT Bucket Versioning

The PUT Bucket Versioning operation enables the bucket owner to set the versioning state of an existing bucket. Supported versioning state values are as follows:

- `Enabled` - Enables versioning for the objects in the bucket. All objects added to the bucket receive a unique version ID.

- `Disabled` - Disables versioning for the objects in the bucket. All objects added to the bucket receive the version ID null.

> **Note:**
>
> If the versioning state has never been set on a bucket, it has no versioning state; a GET versioning request does not return a versioning state value.

Syntax Example

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?versioning
```

Request Parameters

This implementation of the PUT Bucket Versioning operation does not support the use of request parameters.

Request Headers

The PUT Bucket Versioning operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The PUT Bucket Versioning operation supports the use of the Status and VersioningConfiguration request elements. For more information about these request elements, see PUT Bucket versioning.

Response Headers

The PUT Bucket Versioning operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the PUT Bucket Versioning operation does not return response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The PUT Bucket Versioning operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT /?versioning
<VersioningConfiguration>
  <Status>Enabled</Status>
</VersioningConfiguration>
HTTP/1.1 200 OK
```

DELETE Bucket

The DELETE Bucket operation deletes the bucket named in the URI. Note that all objects (including all object versions and delete markers) in the bucket must be deleted before the bucket itself can be deleted.

Syntax Example

```
DELETE https://appliance:443/s3/v1/export/share_mount_point_path/bucketname
```

Request Parameters

This implementation of the DELETE Bucket operation does not support the use of request parameters.

Request Headers

The DELETE Bucket operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

This implementation of the DELETE Bucket operation does not support the use of request elements.

Response Headers

The DELETE Bucket operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the DELETE Bucket does not return response elements.

Expected HTTP Response Code

```
204 No Content
```

Error Response Code

The DELETE Bucket API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
DELETE /
HTTP/1.1 204 No Content
```

DELETE Bucket Tagging

The DELETE Bucket Tagging operation removes a tag set from the specified bucket.

Syntax Example

```
DELETE https://appliance:443/s3/v1/export/share_mount_point_path/bucketname?tagging
```

Request Parameters

This implementation of the DELETE Bucket Tagging operation does not support the use of request parameters.

Request Headers

The DELETE Bucket Tagging operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

This implementation of the DELETE Bucket Tagging operation does not support the use of request elements.

Response Headers

The DELETE Bucket Tagging operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the DELETE Bucket Tagging operation does not return response elements.

Expected HTTP Response Code

```
204 No Content
```

Error Response Code

The DELETE Bucket Tagging API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
DELETE /?tagging
HTTP/1.1 204 No Content
```

Operations on Objects

The Oracle ZFS Storage Appliance S3 Object API supports the following operations on objects:

- GET Object
- GET Object ACL
- GET Object Tagging
- HEAD Object
- OPTIONS Object
- PUT Object
- PUT Object Copy
- PUT Object ACL
- PUT Object Tagging
- POST Object
- DELETE Object
- DELETE Object Tagging
- Multipart Upload

GET Object

The GET Object retrieves S3 objects. To use this operation, you must have READ access to the object. If READ access is granted to an anonymous user, the object is returned without an authorization header. Note that the GET Object operation, by default, returns the current version of an object. To return a different version, use the `versionId` subresource. In cases where the current version of the object is a delete marker, S3 behaves as if the object was deleted and includes `x-amz-delete-marker: true` in the response.

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucketname/
objectname
```

Request Parameters

This implementation of the GET Object operation does not support the use of request parameters.

Request Headers

The GET Object operation supports the use of the following request headers:

- Request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.
- Request headers for retrieving objects. For a description of these request headers, see GET Object.

Request Elements

The GET Object operation does not support the use of request elements.

Response Headers

The GET Object operation supports the use of response headers. For a description of the response headers supported for the GET Object operation, see GET Object.

Response Elements

This implementation of the GET Object operation does not return response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

This implementation of the GET Object operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET /my-image.jpg
HTTP/1.1 200 OK
x-amz-request-id: tx318BC8BC148832E5
Date: Mon, 3 Oct 2016 22:32:00 GMT
Last-Modified: Wed, 12 Oct 2009 17:50:00 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234

[434234 bytes of object data]
```

Example Response when the Latest Object is a Delete Marker

```
HTTP/1.1 404 Not Found
x-amz-request-id: 318BC8BC148832E5
x-amz-version-id: 003
x-amz-delete-marker: true
```

```
Date: Wed, 28 Oct 2018 22:32:00 GMT
Content-Type: text/plain
Connection: close
```

Example Request Getting a Specified Version of an Object

```
GET /myObject?versionId=002 HTTP/1.1
Date: Wed, 28 Oct 2018 22:32:00 GMT
Authorization: authorization string
HTTP/1.1 200 OK
x-amz-request-id: 318BC8BC148832E5
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2018 12:00:00 GMT
x-amz-version-id: 002
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234
Content-Type: text/plain
Connection: close
[434234 bytes of object data]
```

GET Object ACL

The GET Object ACL operation returns the access control list (ACL) for the specified object. To use this operation, you must have `READ_ACP` access to the object.

Syntax Example

`GET https://`*`appliance`*`:443/s3/v1/export/`*`share_mount_point_path/bucketname/objectname`*`?acl`

Request Parameters

This implementation of the GET Object ACL does not support the use of request parameters.

Request Headers

The GET Object ACL operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Object ACL operation does not support the use of request elements.

Response Headers

The GET Object ACL operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported elements in the XML response for the GET Object ACL operation, see GET Object ACL.

Normal Response Code

```
200 OK
```

Error Response Code

The GET Object API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

Sample Response

```
GET /my-image.jpg?acl
HTTP/1.1 200 OK
x-amz-request-id: tx318BC8BC148832E5
x-amz-version-id: 009
Date: Wed, 28 Oct 2018 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2018 12:00:00 GMT
Content-Length: 124
Content-Type: text/plain
Connection: close

<AccessControlPolicy>
  <Owner>
    <ID>micky</ID>
    <DisplayName>micky</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee type="CanonicalUser">
        <ID>minny</ID>
        <DisplayName>minny</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

Sample Request Getting the ACL of the Specific Version of an Object

```
GET /my-image.jpg?versionId=0003
HTTP/1.1 200 OK
x-amz-request-id: 318BC8BC148832E5
Date: Wed, 28 Oct 2018 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2018 12:00:00 GMT
x-amz-version-id: 0004
Content-Length: 124
Content-Type: text/plain
Connection: close

<AccessControlPolicy>
  <Owner>
    <ID>micky</ID>
    <DisplayName>micky</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
```

```
      <Grantee xsi:type="CanonicalUser">
        <ID>minny</ID>
        <DisplayName>minny</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

GET Object Tagging

The GET Object Tagging operation returns the tags associated with an object by sending the GET request against the tagging subresource associated with the object. To use this operation, you must have READ permissions on the object.

Syntax Example

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucketname/
objectname?tagging
```

Request Parameters

This implementation of GET Object Tagging API does not support the use of request parameters.

Request Headers

The GET Object Tagging operation uses only request headers that are common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.

Request Elements

The GET Object Tagging operation does not support the use of request elements.

Response Headers

The GET Object Tagging operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported elements in the XML response for the GET Object Tagging operation, see GET Object tagging.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The GET Object Tagging operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET /example-object?tagging
HTTP/1.1 200 OK
Date: Thu, 22 Sep 2018 21:33:08 GMT
Connection: close
Server: Apache
<?xml version="1.0" encoding="UTF-8"?>
<Tagging xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <TagSet>
      <Tag>
         <Key>tag1</Key>
         <Value>val1</Value>
      </Tag>
      <Tag>
         <Key>tag2</Key>
         <Value>val2</Value>
      </Tag>
   </TagSet>
</Tagging>
```

HEAD Object

The HEAD Object operation retrieves metadata from an object without returning the object itself. This operation is useful if you are interested only in an object's metadata. To use this operation, you must have READ access to the object. A HEAD request has the same options as a GET operation on an object. The response is identical to the GET response except that there is no response body.

Syntax Example

```
HEAD https://appliance:443/s3/v1/export/share_mount_point_path/bucketname/objectname
```

Request Parameters

This implementation of the HEAD Object operation does not support the use of request parameters.

Request Headers

The HEAD Object operation supports the use of the following type of request headers:

- Request headers common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers
- Request headers for Head Object operations. For more information, see HEAD Object.

Request Elements

The HEAD Object operation does not support the use of request elements.

Response Headers

This implementation of the HEAD Object operation supports the use of the `x-amz-meta-*` and `x-amz-version-id` response headers. For more details about these response headers, see HEAD Object.

Response Elements

This implementation of the HEAD Object operation does not return response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The HEAD Object API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
HEAD /my-image.jpg
HTTP/1.1 200 OK
x-amz-request-id: 318BC8BC143432E5
x-amz-version-id: 0007
x=custom-made: custom-value
Date: Wed, 28 Oct 2018 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2018 12:00:00 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234
Content-Type: text/plain
Connection: close
Server: Apache
```

OPTIONS Object

The `403 Forbidden` response code is always returned for the OPTIONS Object API operation. The Oracle ZFS Storage Appliance S3 API does not support the use of a cross-origin resource sharing CORS configuration on a bucket.

Expected HTTP Response Code

```
403 Forbidden
```

PUT Object

The PUT Object operation adds an object to a bucket. To add an object to a bucket, you must have WRITE permissions on the bucket. To ensure data is not corrupted

when using the PUT Object operation, you should use the `Content-MD5` header. To configure your application to send `Request Headers` prior to sending the request body, use the `100-continue` HTTP status code.

Storage Class Options

Oracle ZFS Storage Appliance only supports the STANDARD storage class option.

Access Permissions

To grant specific permission on an object using a request header, you can either:

- Specify a canned (predefined) ACL using the `x-amz-acl` request header. For more information, see Controlling Access to Resources Using S3 ACLs.
- Specify access permissions explicitly using the `x-amz-grant-read`, `x-amz-grant-read-acp`, and `x-amz-grant-write-acp`, `x-amz-grant-full-control` headers. These headers map to the set of permissions S3 supports in an ACL. For more information, see Controlling Access to Resources Using S3 ACLs.

Syntax Example

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucketname/objectname
```

Object Versioning

If you enable versioning for a bucket, S3 automatically generates a unique version ID for the object being stored. S3 returns this ID in the response using the `x-amz-version-id` response header. If versioning is suspended, S3 always uses `null` as the version ID for the object stored. If you enable versioning for a bucket, when S3 receives multiple write requests for the same object simultaneously, it stores all of the objects as separate versions.

Request Parameters

This implementation of the PUT Object operation does not support the use of request parameters.

Request Headers

The PUT Object operation supports the use of following request headers:

- Request headers common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.
- Request headers for PUT Object operations, which include `Content-Disposition`, `Content-Encoding`, `Content-Length`, `Content-MD5`, `Content-Type`, `Expect`, `x-amz-meta-`, `x-amz-tagging`. For a description of these request headers, see PUT Object.

Request Elements

The PUT Object operation does not support the use of request elements.

Response Headers

The PUT Object operation supports the use of the following response headers:

- Response headers common to all operations. For more information, see table "Supported Response Headers" in Supported Common Response Headers.
- The `x-amz-version-id` header. This header describes the object version.

Response Elements

This implementation of the PUT Object operation does not return response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The PUT Object API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT /my-image.jpg
Date: Wed, 12 Oct 2018 17:50:00 GMT
Authorization: authorization string
Content-Type: text/plain
Content-Length: 11434
x-amz-grant-full-control: id="michael"
x-amz-meta-author: Janet
Expect: 100-continue

HTTP/1.1 100 Continue

HTTP/1.1 200 OK
x-amz-request-id: tx0A49CE4060975EAC
Date: Wed, 12 Oct 2018 17:50:00 GMT
ETag: "1b2cf535f27731c974343645a3985328"
Content-Length: 0
Connection: close
Server: Apache
```

When versioning is enabled on the bucket, the response includes the `x-amz-version-id` header:

```
HTTP/1.1 100 Continue

HTTP/1.1 200 OK
x-amz-request-id: tx0A49CE4060975EAC
x-amz-version-id: 0075
Date: Wed, 12 Oct 2018 17:50:00 GMT
ETag: "fbacf535f27731c9771645a39863328"
Content-Length: 0
Connection: close
```

PUT Object Copy

The PUT Object Copy operation creates a copy of a stored S3 object. A PUT Object Copy operation is the same as performing a GET and then a PUT. Adding the request header, `x-amz-copy-source`, makes the PUT operation copy the source object into the destination bucket. When copying an object, you can preserve most of the metadata (default) or specify new metadata. However, the ACL is not preserved and is set to private for the user making the request. All copy requests must be authenticated and cannot contain a message body. Additionally, you must have READ access to the source object and WRITE access to the destination bucket. To copy an object only under certain conditions, such as whether the ETag matches or whether the object was modified before or after a specified date, use the request headers `x-amz-copy-source-if-match`, `x-amz-copy-source-if-none-match`, `x-amz-copy-source-if-unmodified-since`, or `x-amz-copy-source-if-modified-since`.

Syntax Example

For a syntax example, see PUT Object - Copy.

Versioning

By default, `x-amz-copy-source` identifies the current version of an object to copy. However, if the current version is a delete marker, S3 behaves as if the object were deleted.

To copy a different version, use the `versionId` subresource. If you enable versioning on the target bucket, S3 generates a unique version ID for the object being copied. This version ID is different from the version ID of the source object. S3 returns the version ID of the copied object in the `x-amz-version-id` response header in the response. Note that if you do not enable versioning or suspend versioning on the target bucket, the version ID S3 generates a null.

Access Permissions

To grant specific permission on an object using a request header, you can either:

- Specify a canned (predefined) ACL using the `x-amz-acl` request header. For more information, see Controlling Access to Resources Using S3 ACLs.
- Specify access permissions explicitly using the `x-amz-grant-read`, `x-amz-grant-read-acp`, and `x-amz-grant-write-acp`, `x-amz-grant-full-control` headers. These headers map to the set of permissions S3 supports in an ACL. For more information, see Controlling Access to Resources Using S3 ACLs.

> **Note:**
>
> You can use either a canned ACL or specify access permissions explicitly. You cannot do both.

Request Parameters

This implementation of PUT Object Copy operation does not support the use of request parameters.

Request Headers

The PUT Object Copy operation supports the use of following request headers:

- Request headers common to all operations. For more information, see table "Common Supported Request Headers" in Supported Common Request Headers.
- Request headers for PUT Object operations, which include `x-amz-copy-source`, `x-amz-metadata-directive`, `x-amz-copy-source-if-match`, `x-amz-copy-source-if-none-match`, `x-amz-copy-source-if-unmodified-since`, `x-amz-copy-source-if-modified-since`, `x-amz-tagging-directive`. For a description of these request headers, see PUT Object - Copy.

Request Elements

The PUT Object Copy operation supports the following requests elements:

- `CopyObjectResult`
- `ETag`
- `LastModified`

For a description of the supported request elements, see PUT Object - Copy.

Response Headers

The PUT Object Copy operation supports the use of the following response headers:

- Response headers common to all operations. For more information, see table "Supported Response Headers" in Supported Common Response Headers.
- Response headers for PUT Object operation, which include `x-amz-version-id` and `x-amz-copy-source-version-id`. For a description of these response headers, see PUT Object - Copy.

Response

This implementation of the PUT Object Copy operation does not return response elements.

Expected HTTP Response Code

`200 OK`

Error Response Code

The PUT Object Copy API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT /my-second-image.jpg HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2018 22:32:00 GMT
x-amz-copy-source: /bucket/my-image.jpg
HTTP/1.1 200 OK
x-amz-request-id: tx318BC8BC148832E5
x-amz-copy-source-version-id: 0009
x-amz-version-id: 0099
Date: Wed, 28 Oct 2018 22:32:00 GMT
Connection: close
Server: Apache

<CopyObjectResult>
   <LastModified>2009-10-28T22:32:00</LastModified>
   <ETag>"9b2cf535f27731c974343645a3985328"</ETag>
 </CopyObjectResult>
```

Where `x-amz-version-id` returns the version ID of the object in the destination bucket, and `x-amz-copy-source-version-id` returns the version ID of the source object.

PUT Object ACL

The PUT Object ACL sets the access control list (ACL) permissions on an existing bucket object. To set ACL permissions on an existing bucket object, you must have `WRITE_ACP` permissions. You can choose to use request headers to specify the permissions, or specify the ACL in the request body.

Versioning

The ACL for an object is set at the object version level. By default, a PUT request sets the ACL for the current version of the object. To set the ACL for a different version, use the `versionId` subresource.

Syntax Example

For syntax examples, see PUT Object acl.

Request Parameters

This implementation of the PUT Object ACL operation does not support the use of request parameters.

Request Headers

The PUT Object ACL operation supports the use of the following access-control headers to set permissions:

- `x-amz-acl` header. Use this header to specify canned ACL permissions.
- `x-amz-grant-permission` header. Use this header to individually specify the permissions for a grantee.

For more information about how to specify ACL permissions, see:

- Controlling Access to Resources Using S3 ACLs
- PUT Object acl

Request Elements

The PUT Object ACL operation supports the use of request elements when not using a request body. Note that if you use a request body, you cannot use the request headers to set an ACL. For a list of supported request elements, see PUT Object acl.

Response Headers

The PUT Object ACL operation supports the use of the following response headers:

- Response headers common to all operations. For more information, see table "Supported Response Headers" in Supported Common Response Headers.
- Response headers for a PUT Object operation, which include `x-amz-version-id`. For further details about this response header, see PUT Object acl.

Response Elements

This implementation of the PUT Object ACL does not support the use of response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Codes

The PUT Object ACL operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT /my-image.jpg?acl
<AccessControlPolicy>
  <Owner>
    <ID>joe</ID>
    <DisplayName>joe/DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee type="CanonicalUser">
        <ID>jack</ID>
        <DisplayName>joe</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>

HTTP/1.1 200 OK
x-amz-request-id: tx318BC8BC148832E5
x-amz-version-id: 0055
Date: Wed, 28 Oct 2018 22:32:00 GMT
```

```
Last-Modified: Sun, 1 Jan 2018 12:00:00 GMT
Content-Length: 0
Connection: close
Server: Apache
```

Alternatively, a request can also be made to a specific version of an object, for instance:

```
PUT /my-image.jpg?acl&versionId=0099
```

PUT Object Tagging

The PUT Object Tagging operation adds a set of tags to an existing object. A tag is a key-value pair. You can associate tags with an object by sending a PUT request against the tagging subresource associated with the object. You can retrieve tags by sending a GET request.

Syntax Example

For syntax examples, see PUT Object tagging.

Request Parameters

This implementation of the PUT Object Tagging operation does not support the use of request parameters.

Request Headers

This implementation of the PUT Object Tagging operation does not support the use of request headers.

Request Elements

For a list of supported request elements, see PUT Object tagging.

Response Headers

The PUT Object Tagging operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the PUT Object Tagging operation does not return response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

`MalformedXMLError`. The XML provided does not match the schema..

Example

```
PUT object-key?tagging HTTP/1.1
Content-Length: length
Content-MD5: pUNXr/BjKK5G2UKExample==
x-amz-date: 20180923T001956Z
Authorization: authorization string
<Tagging>
   <TagSet>
      <Tag>
         <Key>tag1</Key>
         <Value>val1</Value>
      </Tag>
      <Tag>
         <Key>tag2</Key>
         <Value>val2</Value>
      </Tag>
   </TagSet>
</Tagging>
HTTP/1.1 200 OK
x-amz-request-id: tx236A8905248E5A01
Date: Fri, 23 Sep 2018 00:20:19 GMT
```

POST Object

The POST Object operation adds an object to a specified bucket using HTML forms.

> **Note:**
>
> POST is an alternate form of PUT that enables browser-based uploads as a way of putting objects in buckets. Parameters that are passed to PUT in HTTP Headers are instead passed as form fields to POST in the multipart-form-data encoded message body. WRITE access is required to add an object to a bucket. To ensure that data is not corrupted traversing the network, use the Content-MD5 form field. When you use this form field, S3 checks the object against the provided MD5 value. If they do not match, S3 returns an error. Additionally, you can calculate the MD5 value while posting an object to S3 and compare the returned ETag to the calculated MD5 value. The ETag only reflects changes to the contents of an object, not its metadata.

Syntax Example

For request syntax examples, see POST Object.

Request Parameters

This implementation of the POST Object operation does not support the use of request parameters.

Request Headers

This implementation of the POST Object operation does not support the use of request headers.

Request Elements

The request is made through an HTTP form.

Form Field Names Supported In Request

The POST Object operation supports the use of following form fields in a request.

> **Note:**
>
> Server-side encryption form fields are not supported.

Field Name	Field Name
`AWSAccessKeyId`	`policy`
`acl`	`success_action_redirect`
`Cache-Control`, `Content-Type`, `Content-Disposition`, `Content-Encoding`, `Expires`	`success_action_status`
`file`	`tagging`
`key`	`x-amz-storage-class`
`x-amz-meta-*`	

For a description of these supported form fields, see POST Object.

Response Headers

In addition to the response headers common to all responses, this implementation of the POST Object operation can include the following response headers:

- `success_action_redirect`
- `x-amz-version-id`

For a more information about these response headers, see POST Object. For a description of common response headers, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

For a list of supported elements in the XML response for the POST Object operation, see POST Object.

Expected HTTP Error Response Codes

`200 or 201 or 204`

Error Response Code

The POST Object API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
POST / HTTP/1.1
User-Agent: browser_data
Accept: file_types
Accept-Language: Regions
Accept-Encoding: encoding
Accept-Charset: character_set
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: length

--9431149156168
Content-Disposition: form-data; name="key"

acl
--9431149156168
Content-Disposition: form-data; name="tagging"

<Tagging><TagSet><Tag><Key>Tag Name</Key><Value>Tag Value</Value></Tag></
TagSet></Tagging>
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

success_redirect
--9431149156168
Content-Disposition: form-data; name="Content-Type"

content_type
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

uuid
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

metadata
--9431149156168
Content-Disposition: form-data; name="AWSAccessKeyId"

access-key-id
--9431149156168
Content-Disposition: form-data; name="Policy"

encoded_policy
--9431149156168
Content-Disposition: form-data; name="Signature"

signature=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg
```

```
file_content
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to S3
--9431149156168--

response:
HTTP/1.1 100 Continue
HTTP/1.1 200 OK
x-amz-request-id: tx0A49CE4060975EAC
x-amz-version-id: null
Date: Wed, 01 Mar  2018 12:00:00 GMT
ETag: "828ef3fdfa96f00ad9f27c383fc9ac7f"
Content-Length: 0
Connection: close
Server: Apache
```

DELETE Object

If a null version of an object exists, the DELETE operation removes the null version of the object and inserts a delete marker.

Syntax Example

For a syntax example, see DELETE Object.

Versioning

To remove a specific version, you must be the bucket owner and you must use the `versionId` subresource. Using this subresource permanently deletes the version. If the object deleted is a delete marker, S3 sets the response header, `x-amz-delete-marker`, to `true`.

Request Parameters

This implementation of the DELETE Object operation does not support the use of request parameters.

Request Headers

This implementation of the DELETE Object operation does not support the use of request headers.

Request Elements

This implementation of the DELETE Object operation does not support the use of request elements.

Response Headers

This implementation of the DELETE Object operation supports the use of the `x-amz-version-id` response header. For more details about this response header, see DELETE Object.

Response Elements

This implementation of the DELETE Object operation does not return response elements.

Expected HTTP Error Response Code

```
204 No Content
```

Error Response Code

The DELETE Object operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

The following request deletes `my-desk-image.jpg`.

```
DELETE /my-desk-image.jpg
HTTP/1.1 204 NoContent
x-amz-request-id: tx0A49CE4060975EAC
Date: Wed, 12 Oct 2018 17:50:00 GMT
Content-Length: 0
Connection: close
Server: Apache
```

The following request deletes the specified version of the object, `my-desk-image.jpg`.

```
DELETE /my-third-image.jpg?versionId=00012
HTTP/1.1 204 NoContent
x-amz-request-id: tx0A49CE4060975EAC
x-amz-version-id: 00012
Date: Wed, 12 Oct 2018 17:50:00 GMT
Content-Length: 0
Connection: close
Server: Apache
```

If the object deleted is a delete marker, the following response example appears.

```
HTTP/1.1 204 NoContent
x-amz-request-id: tx0A49CE4060975EAC
x-amz-version-id: 0011
x-amz-delete-marker: true
Date: Wed, 12 Oct 2018 17:50:00 GMT
Content-Length: 0
Connection: close
Server: Apache
```

DELETE Object Tagging

The DELETE Object Tagging operation removes the entire tag set from the specified object.

Syntax Example

For a syntax example, see DELETE Object tagging.

Versioning

To delete tags of a specific object version, add the `versionId` query parameter in the request.

Request Parameters

The DELETE Object Tagging operation does not support the use of request parameters.

Request Headers

The DELETE Object Tagging operation does not support the use of request headers.

Request Elements

The DELETE Object Tagging operation does not support the use of request elements.

Response Headers

The DELETE Object Tagging operation uses only response headers that are common to most responses. For more information, see table "Supported Response Headers" in Supported Common Response Headers.

Response Elements

The DELETE Object Tagging operation does not return response elements.

Expected HTTP Response Code

```
204 No Content
```

Error Response Code

The DELETE Object API does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
DELETE exampleobject/?tagging
HTTP/1.1 204 No Content
Date: Wed, 25 Nov 2018 12:00:00 GMT
Connection: close
Server: Apache
```

Multipart Upload

Multipart upload reduces object upload time by uploading parts of a single object in parallel. Parts can be uploaded independently, in any order. Amazon S3 uses the uploaded parts to construct the object.

Individual parts of an object can be uploaded in parallel to reduce upload time. The following object operations are supported for multipart upload:

- Create Multipart Upload
- Upload Part
- Upload Part Copy
- List Parts
- List Multipart Uploads
- Complete Multipart Upload
- Abort Multipart Upload

Create Multipart Upload

The Create Multipart Upload operation initiates an upload and returns the upload ID, which is used for uploading part requests. The ID is also used when completing or aborting the multipart upload request.

Syntax Example

Request syntax example for Create Multipart Upload:

```
POST https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name/
object_name?uploads
```

Request Parameters

This implementation of the Create Multipart Upload operation does not support request parameters.

Request Headers

This implementation of the Create Multipart Upload operation supports request headers that are common to all operations, as described in table "Common Supported Request Headers" in Supported Common Request Headers. In addition, the following request headers are supported, as described in Create Multipart Upload:

- `Content-Disposition`
- `Content-Encoding`
- `Content-Language`
- `Content-Type`
- `x-amz-storage-class` - ignore value, all `STANDARD`
- `x-amz-tagging`

- `x-amz-acl`
- `x-amz-grant-full-control`
- `x-amz-grant-read`
- `x-amz-grant-read-acp`
- `x-amz-grant-write-acp`

Request Elements

This implementation of the Create Multipart Upload operation does not support the use of request elements.

Response Headers

This implementation of the Create Multipart Upload operation supports response headers that are common to all operations, as described in table "Common Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the Create Multipart Upload operation returns the following response elements, as described in Create Multipart Upload:

- `InitiateMultipartUploadResult`
- `Bucket`
- `Key`
- `UploadId`

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The Create Multipart Upload operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
POST /example-object?uploads HTTP/1.1
x-amz-date: Thu, 12 Apr 2023 20:04:21 GMT
Authorization: authorization string

HTTP/1.1 200 OK
<InitiateMultipartUploadResult>
  <Bucket>example-bucket</Bucket>
  <Key>example-object</Key>
  <UploadId>VXBsb2FkIElEIGZvciA2aWWpbmcncyBteS1tb3ZpZS5tMnRzIHVwbG9hZA</UploadId>
</InitiateMultipartUploadResult>
```

Upload Part

The Upload Part operation uploads a part of a multipart upload. In the request, use the upload ID from the Create Multipart Upload operation.

Syntax Example

Request syntax example for Upload Part:

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name/
object_name?partNumber=PartNumber&uploadId=UploadId
```

Request Parameters

This implementation of the Upload Part operation supports the following request parameters, as described in Upload Part:

- `partNumber` - required
- `uploadId` - required

Request Headers

This implementation of the Upload Part operation supports the following common request headers, as described in table "Common Supported Request Headers" in Supported Common Request Headers:

- `Content-Length`
- `Content-MD5`
- `x-amz-expected-bucket-owner`

Request Elements

This implementation of the Upload Part operation accepts the following binary data:
Body - The content to upload.

Response Headers

This implementation of the Upload Part operation supports the use of the following response header, as described in Upload Part: `ETag`.

Response Elements

This implementation of the Upload Part operation does not support the use of response elements.

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The Upload Part operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT /example-object?partNumber=1&uploadId=a8251fc8-6e82-4b2c-af7a-74ed97dd6c5d HTTP/1.1
x-amz-date: Thu, 12 Apr 2023 20:04:21 GMT
Authorization: authorization string
Content-Length: 10485760
***part data omitted***

HTTP/1.1 200 OK
x-amz-request-id: tx6313b3960a2a4b5ead0c7-006448847a
Date: Thu, 12 Apr 2023 20:05:21 GMT
ETag: "1b2cf535f27731c974343645a3985328"
Content-Length: 0
```

Upload Part Copy

The Upload Part Copy operation uploads a part of a multipart upload by copying data from an existing object as the data source. In the request, specify the data source with request header `x-amz-copy-source` and specify the byte range with request header `x-amz-copy-source-range`.

Syntax Example

Request syntax example for Upload Part Copy:

```
PUT https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name/object_name?
partNumber=PartNumber&uploadId=UploadId
```

Versioning

If versioning is enabled for the bucket, multiple versions of the same object could exist. By default, `x-amz-copy-source` identifies the current version of an object to copy as part of a multipart upload. However, if the current version is a delete marker, Amazon S3 behaves as if the object were deleted.

To copy a different version, use the `versionId` subresource. For example: `x-amz-copy-source: /source-bucket/source-object?versionId=version id`.

Request Parameters

This implementation of the Upload Part Copy operation supports the following request parameters, as described in Upload Part Copy:

- `partNumber` - required
- `uploadId` - required

Request Headers

This implementation of the Upload Part Copy operation supports the following request headers, as described in Upload Part Copy:

- `x-amz-copy-source`
- `x-amz-copy-source-if-match`
- `x-amz-copy-source-if-modified-since`
- `x-amz-copy-source-if-none-match`
- `x-amz-copy-source-if-unmodified-since`
- `x-amz-copy-source-range`
- `x-amz-source-expected-bucket-owner`

Request Elements

This implementation of the Upload Part Copy operation does not support the use of request elements.

Response Headers

This implementation of the Upload Part Copy operation supports the use of the following response header, as described in Upload Part Copy: `x-amz-copy-source-version-id`.

Response Elements

This implementation of the Upload Part Copy operation returns the following response elements, as described in Upload Part Copy:

- `CopyPartResult`
- `ETag`
- `LastModified`

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The Upload Part Copy operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
PUT /example-object?partNumber=2&uploadId=a8251fc8-6e82-4b2c-af7a-74ed97dd6c5d
HTTP/1.1
x-amz-date: Thu, 12 Apr 2023 20:04:21 GMT
Authorization: authorization string
```

```
x-amz-copy-source: /source-bucket/sourceobject?versionId=9e4034c5-9d80-45b4-
a98a-344f5ed6526c

HTTP/1.1 200 OK
x-amz-request-id: tx6313b3960a2a4b5ead0c7-006448847a
x-amz-copy-source-version-id: 9e4034c5-9d80-45b4-a98a-344f5ed6526c
Date: Thu, 12 Apr 2023 20:05:21 GMT
<CopyPartResult>
  <LastModified>2023-04-12T20:05:21.000Z</LastModified>
  <ETag>"cc9472149704128a4a7e3f90f828e394"</ETag>
</CopyPartResult>
```

List Parts

For a specific multipart upload, the List Parts operation lists the parts that have been uploaded. In the request, use the upload ID from the Create Multipart Upload operation. The default and maximum number of uploaded parts is 1000. To restrict the number of parts returned, use the `max-parts` request parameter, with a value from 1 to 1000. If the number of uploaded parts exceeds the list's limit, response field `IsTruncated` is set to true, and the response includes element `NextPartNumberMarker`, which can be used in a subsequent List Parts request.

Syntax Example

Request syntax example for List Parts:

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name/object_name?
max-parts=MaxParts&part-number-marker=PartNumberMarker&uploadId=UploadId
```

Request Parameters

This implementation of the List Parts operation supports the following request parameters, as described in List Parts:

- `max-parts` - default value is 1000; error is returned if value is not between 1 to 1000
- `part-number-marker` - error is returned if part not found
- `uploadId`

Request Headers

This implementation of the List Parts operation supports the following request header, as described in Supported Common Request Headers: `x-amz-expected-bucket-owner`.

Request Elements

This implementation of the List Parts operation does not support the use of request elements.

Response Headers

This implementation of the List Parts operation supports response headers that are common to all operations, as described in table "Common Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the List Parts operation returns the following response elements, as described in List Parts:

- `Bucket`
- `Initiator`
- `IsTruncated`
- `Key`
- `ListPartsResult`
- `MaxParts`
- `NextPartNumberMarker`
- `Owner`
- `Part`
- `PartNumberMarker`
- `StorageClass`
- `UploadId`

Expected HTTP Response Code

`200 OK`

Error Response Code

The List Parts operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET example-object?max-parts=1&part-number-
marker=1&uploadId=a7846887-2ad7-48d0-8547-d81549a24853 HTTP/1.1
x-amz-date: Thu, 12 Apr 2023 20:04:21 GMT
Authorization: authorization string

HTTP/1.1 200 OK
x-amz-request-id:txb71706d7a30f432cbe99d-0064488d79
Date: Thu, 12 Apr 2023 20:05:21 GMT
Content-Length: 693
<ListPartsResult>
  <Bucket>example-bucket</Bucket>
  <Key>example-object</Key>
  <UploadId>a7846887-2ad7-48d0-8547-d81549a24853</UploadId>
  <Initiator>
    <ID>user_tester</ID>
    <DisplayName>user_tester</DisplayName>
  </Initiator>
  <Owner>
    <ID>user_tester</ID>
    <DisplayName>user_tester</DisplayName>
```

```
  </Owner>
  <StorageClass>STANDARD</StorageClass>
  <PartNumberMarker>1</PartNumberMarker>
  <NextPartNumberMarker>2</NextPartNumberMarker>
  <MaxParts>1</MaxParts>
  <IsTruncated>false</IsTruncated>
  <Part>
    <PartNumber>2</PartNumber>
    <LastModified>2023-04-12T20:05:21.000Z</LastModified>
    <ETag>"cc9472149704128a4a7e3f90f828e394"</ETag>
    <Size>168</Size>
  </Part>
</ListPartsResult>
```

List Multipart Uploads

The List Multipart Uploads operation lists in-progress multipart uploads that were started with the Initiate Multipart Upload request, but have not completed nor aborted. The default and maximum number of multipart uploads is 1000. To restrict the number of uploads returned, use the `max-uploads` request parameter, with a value from 1 to 1000. If the number of in-progress uploads exceeds the list's limit, response field `IsTruncated` is set to true, and the response includes elements `key-marker` and `upload-id-marker`, which can be used in a subsequent List Multipart Uploads request.

Syntax Example

Request syntax example for List Mulitpart Uploads:

```
GET https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name?
uploads&delimiter=Delimiter&encoding-type=EncodingType&key-marker=KeyMarker&max-
uploads=MaxUploads&prefix=Prefix&upload-id-marker=UploadIdMarker
```

Request Parameters

This implementation of the List Multipart Uploads operation supports the following request parameters, as described in List Multipart Uploads:

- `delimiter`
- `encoding-type`
- `key-marker`
- `max-uploads`
- `prefix`
- `upload-id-marker`

Request Headers

This implementation of the List Multipart Uploads operation supports the following request header, as described in Supported Common Request Headers: `x-amz-expected-bucket-owner`.

Request Elements

This implementation of the List Multipart Uploads operation does not support the use of request elements.

Response Headers

This implementation of the List Multipart Uploads operation supports response headers that are common to all operations, as described in table "Common Supported Response Headers" in Supported Common Response Headers.

Response Elements

For the List Multipart Uploads operation, the following response elements are supported, as described in List Multipart Uploads:

- `Bucket`
- `CommonPrefixes`
- `Delimiter`
- `EncodingType`
- `IsTruncated`
- `KeyMarker`
- `ListMultipartUploadsResult`
- `MaxUploads`
- `NextKeyMarker`
- `NextUploadIdMarker`
- `Prefix`
- `Upload`
- `UploadIdMarker`

Expected HTTP Response Code

```
200 OK
```

Error Response Code

The List Multipart Uploads operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
GET ?uploads&max-uploads=3&key-marker=apple.jpg HTTP/1.1
x-amz-date: Mon, 17 Apr 2023 20:04:21 GMT
Authorization: authorization string

HTTP/1.1 200 OK
```

```
x-amz-request-id:txa3b321e1a34d4628a6546-00644896f0
Date: Mon, 17 Apr 2023 20:05:21 GMT
Content-Length: 733
<ListMultipartUploadsResult>
  <Bucket>example-bucket</Bucket>
  <KeyMarker>apple.jpg</KeyMarker>
  <UploadIdMarker></UploadIdMarker>
  <NextKeyMarker>banana.jpg</NextKeyMarker>
  <NextUploadIdMarker>43fff991-1e1b-4088-9753-d0bd0b7439ca</NextUploadIdMarker>
  <MaxUploads>1</MaxUploads>
  <IsTruncated>true</IsTruncated>
  <Upload>
    <Key>banana.jpg</Key>
    <UploadId>43fff991-1e1b-4088-9753-d0bd0b7439ca</UploadId>
    <Initiator>
      <ID>user_terster</ID>
      <DisplayName>user_tester</DisplayName>
    </Initiator>
    <Owner>
      <ID>user_tester</ID>
      <DisplayName>user_tester</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <Initiated>2023-04-16T20:48:33.000Z</Initiated>
  </Upload>
</ListMultipartUploadsResult>
```

Complete Multipart Upload

The Complete Multipart Upload operation assembles previously uploaded parts to complete the upload. In the request, include the parts list. For each part in the list, provide the part number and the `ETag` value that was returned after part upload.

Syntax Example

Request syntax example for Complete Mulitpart Upload:

`POST https://`*`appliance`*`:443/s3/v1/export/`*`share_mount_point_path/bucket_name/object_name`*`?uploadId=`*`UploadId`*

Request Parameters

This implementation of the Complete Multipart Upload operation supports the following request parameter, as described in Complete Multipart Upload: `uploadId`.

Request Headers

This implementation of the Complete Multipart Upload operation supports the following request header, as described in Supported Common Request Headers: `x-amz-expected-bucket-owner`.

Request Elements

This implementation of the Complete Multipart Upload operation supports the following request elements, as described in Complete Multipart Upload:

- `CompleteMultipartUpload`

- `Part`

Response Headers

This implementation of the Complete Multipart Upload operation supports the following response header, as described in Complete Multipart Upload: `x-amz-version-id`.

Response Elements

For the Complete Multipart Upload operation, the following response elements are supported, as described in Complete Multipart Upload:

- `Bucket`
- `CompleteMultipartUploadResult`
- `ETag`
- `Key`

Expected HTTP Response Code

`200 OK`

Error Response Code

The Complete Multipart Upload operation returns special errors described in Complete Multipart Upload. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
POST example-object?uploadId=a7846887-2ad7-48d0-8547-d81549a24853 HTTP/1.1
x-amz-date: Mon, 17 Apr 2023 20:04:21 GMT
Authorization: authorization string
<CompleteMultipartUpload xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Part>
    <ETag>cc9472149704128a4a7e3f90f828e394</ETag>
    <PartNumber>1</PartNumber>
  </Part>
  <Part>
    <ETag>0c78aef83f66abc1fa1e8477f296d3943</ETag>
    <PartNumber>2</PartNumber>
  </Part>
</CompleteMultipartUpload>

HTTP/1.1 200 OK
x-amz-request-id: txf8a4dc13d8ee4463bfe17-00644899b3
Date: Mon, 17 Apr 2023 20:05:21 GMT
Content-Length: 178
<CompleteMultipartUploadResult>
    <Bucket>example-bucket</Bucket>
    <Key>exmple-object</Key>
    <ETag>"018030db837fde4dad2d68efd69985d5"</ETag>
</CompleteMultipartUploadResult>
```

Abort Multipart Upload

The Abort Multipart Upload operation aborts a multipart upload for the specified upload ID. After issuing this request, no additional parts can be uploaded for the same upload ID. For further details, see Abort Multipart Upload.

Syntax Example

Request syntax example for Abort Mulitpart Upload:

```
DELETE https://appliance:443/s3/v1/export/share_mount_point_path/bucket_name/
object_name?uploadId=UploadId
```

Request Parameters

This implementation of the Abort Multipart Upload operation supports the following request parameter, as described in Abort Multipart Upload: `uploadId`.

Request Headers

This implementation of the Abort Multipart Upload operation supports the following request header, as described in Supported Common Request Headers: `x-amz-expected-bucket-owner`.

Request Elements

This implementation of the Abort Multipart Upload operation does not support the use of request elements.

Response Headers

This implementation of the Abort Multipart Upload operation supports response headers that are common to all operations, as described in table "Common Supported Response Headers" in Supported Common Response Headers.

Response Elements

This implementation of the Abort Multipart Upload operation does not support the use of response elements.

Expected HTTP Response Code

```
204 No Content
```

Error Response Code

The Abort Multipart Upload operation does not return special errors. For general information about S3 errors and a list of error codes, see S3 Client Error Handling Reference.

Example

```
DELETE example-object?uploadId=a7846887-2ad7-48d0-8547-d81549a24853 HTTP/1.1
x-amz-date: Mon, 17 Apr 2023 20:04:21 GMT
Authorization: authorization string

HTTP/1.1 204 NO CONTENT
x-amz-request-id: txf8a4dc13d8ee4463bfe17-00644899b3
Date: Mon, 17 Apr 2023 20:05:21 GMT
```

4

S3 Client Error Handling Reference

This reference identifies errors supported by the Oracle ZFS Storage Appliance S3 Object API.

Error Response Format

When an error occurs, the header contains the following information:

- Content-Type: application/xml
- HTTP status code (4xx or 5xx)

Information about an error is also available in the body or response. The following example shows the structure of response elements that are common to all REST error responses.

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>NoSuchKey</Code>
  <Message>The resource you requested does not exist</Message>
  <Resource>/mybucket/myfoto.jpg</Resource>
  <RequestId>4442587FB7D0A2F9</RequestId>
</Error>
```

For further information about REST error response elements, see the Amazon S3 REST API Error Responses documentation.

S3 Client Error Codes

For a list of supported error codes, see the Amazon S3 REST API Error Responses documentation.