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

F13773-03
January 2023

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

F13773-03

Primary Author: Heidi Hall

Contents

1 About ZFS Object Store

Oracle ZFS Storage Appliance systems allow users to store data objects using ZFS object store. This capability is implemented using either OpenStack Object Storage API V1.0, also known as Swift, which uses an HTTP RESTful interface, or the Amazon Simple Storage Service (S3) API, which also uses an HTTP RESTful interface. This guide describes how to use the OpenStack Object Storage APIs.

For information on how to configure the ZFS object store data service, see Object API Configuration in *Oracle ZFS Storage Appliance Administration Guide, Release OS8.8.x*.

For information on how to use the Amazon S3 API, see *Oracle ZFS Storage Appliance Object API Guide for Amazon S3 Service Support, Release OS8.8.x* .

For best practices and more information about object store, refer to the technical brief *Using Oracle ZFS Storage Appliance as an Object Store Repository* .

2 OpenStack Object Storage API

Oracle ZFS Storage Appliance systems implement their object capability with OpenStack Object Storage API v1.0, also known as Swift, using an HTTP RESTful interface.

The OpenStack Object Storage API uses three elements to access an object: `<Account-Name>`, `<Container-Name>`, and `<Object-Name>`.

In the Oracle ZFS Storage Appliance object storage implementation, the share export path is used for the OpenStack account object. It is specified in the storage URL; for example, `http://<ObjectStoreNode>/v1/export/<ShareName>/<ContainerName>/<ObjectName>`

For more information on OpenStack Object Storage API, see the following topics:

- Supported OpenStack Object Storage API Capabilities
- Swift Command-Line Client
- Curl Command-Line Examples

Supported OpenStack Object Storage API Capabilities

The following topics describe the supported OpenStack Object Storage API V1.0 capabilities for Oracle ZFS Storage Appliance.

For full details about OpenStack Object Storage API, including descriptions of the request and response parameters, see the OpenStack Object Storage API documentation: https://docs.openstack.org/api-ref/object-store/index.html.

> **Note:**
>
> Some optional request and response parameters are not supported by Oracle ZFS Storage Appliance systems. Unsupported parameters are listed in the following tables.

Discoverability

If configured, lists the activated capabilities for this version of the OpenStack Object Storage API.

Table 2-1 Discoverability Commands

Request	Path	Description	Normal Response Codes	Unsupported Request Parameters	Unsupported Response Parameters
GET	/info	Lists the activated capabilities for this version of the OpenStack Object Storage API.	200	swiftinfo_sig, swiftinfo_expires	

Accounts

These operations are used to perform account-level tasks.

Table 2-2 Accounts Commands

Request	Path	Description	Normal Response Codes	Unsupported Request Parameters	Unsupported Response Parameters
GET	/v1/{account}	Shows account details and list containers, sorted by name, in the account.	200, 204	X-Newest, Accept, X-Trans-Id-Extra	X-Account-Meta-Temp-URL-Key, X-Account-Meta-Temp-URL-Key-2
POST	/v1/{account}	Creates, updates, or deletes account metadata.	204	X-Account-Meta-Temp-URL-Key, X-Account-Meta-Temp-URL-Key-2, X-Trans-Id-Extra	
HEAD	/v1/{account}	Shows account metadata.	204	X-Newest, X-Trans-Id-Extra	X-Account-Meta-Temp-URL-Key, X-Account-Meta-Temp-URL-Key-2

Containers

These operations are used to perform container-level tasks.

Table 2-3 Containers Commands

Request	Path	Description	Normal Response Codes	Unsupported Request Parameters	Unsupported Response Parameters
GET	/v1/{account}/{container}	Shows details for a container and lists objects, sorted by name, in the container.	200, 204	X-Newest, Accept, X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2, X-Trans-Id-Extra	X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2
PUT	/v1/{account}/{container}	Creates a container.	201, 204	X-Container-Sync-To, X-Container-Sync-Key, X-Container-Meta-Access-Control-Allow-Origin, X-Container-Meta-Access-Control-Max-Age, X-Container-Meta-Access-Control-Expose-Headers, X-Newest, Accept, X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2, X-Trans-Id-Extra	
DELETE	/v1/{account}/{container}	Deletes an empty container.	204	X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2, X-Trans-Id-Extra	
POST	/v1/{account}/{container}	Creates, updates, or deletes container metadata.	204	X-Remove-Container-name, X-Container-Sync-To, X-Container-Sync-Key, X-Container-Meta-Access-Control-Allow-Origin, X-Container-Meta-Access-Control-Max-Age, X-Container-Meta-Access-Control-Expose-Headers, X-Container-Meta-Quota-Bytes, X-Container-Meta-Quota-Count, X-Container-Meta-Web-Directory-Type, X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2, X-Trans-Id-Extra	

Table 2-3 (Cont.) Containers Commands

Request	Path	Description	Normal Response Codes	Unsupported Request Parameters	Unsupported Response Parameters
HEAD	/v1/{account}/{container}	Shows container metadata, including the number of objects and the total bytes of all objects stored in the container.	204	X-Newest, X-Trans-Id-Extra, X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2, X-Trans-Id-Extra	X-Container-Meta-Access-Control-Allow-Origin, X-Container-Meta-Access-Control-Max-Age, X-Container-Meta-Access-Control-Expose-Headers,, X-Container-Meta-Quota-Bytes, X-Container-Meta-Quota-Count, X-Container-Sync-To, X-Container-Sync-Key, X-Container-Meta-Temp-URL-Key, X-Container-Meta-Temp-URL-Key-2

Objects

These operations are used to perform object-level tasks.

Table 2-4 Objects Commands

Request	Path	Description	Normal Response Codes	Unsupported Request Parameters	Unsupported Response Parameters
GET	/v1/{account}/{container}/{object}	Downloads the object content and gets the object metadata.	200	X-Newest, temp_url_sig, temp_url_expires, filename, X-Trans-Id-Extra	Content-Disposition, X-Static-Large-Object
PUT	/v1/{account}/{container}/{object}	Creates an object with data content and metadata, or replaces an existing object with data content and metadata.	201	temp_url_sig, temp_url_expires, filename, Transfer-Encoding, Content-Disposition, X-Trans-Id-Extra	
COPY	/v1/{account}/{container}/{object}	Copies an object to another object in the object store.	201	Content-Disposition, X-Fresh-Metadata, X-Trans-Id-Extra	

Table 2-4 (Cont.) Objects Commands

Request	Path	Description	Normal Response Codes	Unsupported Request Parameters	Unsupported Response Parameters
DELETE	/v1/{account}/{container}/{object}	Permanently deletes an object from the object store.	204	X-Trans-Id-Extra	
POST	/v1/{account}/{container}/{object}	Creates or updates object metadata.	202	Content-Disposition, X-Trans-Id-Extra,	
HEAD	/v1/{account}/{container}/{object}	Shows object metadata.	200, 204	temp_url_sig, temp_url_expires, filename, X-Newest, X-Trans-Id-Extra	Content-Disposition, X-Static-Large-Object

Swift Command-Line Client

OpenStack Object Storage provides a command-line client, Swift, that can be used to interface with ZFS object store. The Swift command-line client is an open-source Python client that can be run on Linux or Apple Mac OS X systems.

The following table describes the Swift client commands.

Table 2-5 Swift Commands

Command	Description
delete	Delete a container or objects within a container.
download	Download objects from containers.
list	Lists the containers for the account or the objects for a container.
post	Updates meta information for the account, container, or object; creates containers if not present.
copy	Copies object, optionally adds meta.
stat	Displays information for the account, container, or object.
upload	Uploads files or directories to the given container.

Swift Examples

The following examples show how to use the Swift command-line client.

> **Note:**
>
> The HTTPS protocol is supported, as well as HTTP.

List containers using the default path

```
swift -A http://zfssa:80/auth/v1.0 -U user -K key list
```

The output lists the containers that are part of an account.

List containers using a specified share location

```
swift -A http://zfssa:80/auth/v1.0/export/fs1 -U user -K key list
```

List objects in a container

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key list containerA
```

Create/upload object to container

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key upload containerA object1
```

Download container

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key download containerA
```

Delete container

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key delete containerA
```

Download object

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key download containerA object1
```

Delete object

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key delete containerA object1
```

Set metadata to an object

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key post containerA object1 --
header "X-Object-Meta-Movie: comedy”
```

Get metadata from an object

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key stat -v containerA object1
```

Upload a large file with segments using Dynamic Large Object

Dynamic Large Object is a middleware component supported by Oracle ZFS Storage Appliance.

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key upload containerA large_file -
S 100000000
```

Download large file uploaded using Dynamic Large Object

Dynamic Large Object is a middleware component supported by Oracle ZFS Storage Appliance.

```
swift -A http://zfssa:80/auth/v1.0/ -U user -K key download containerA large_file
```

Bulk Delete or Bulk Upload

Bulk Delete and Bulk Upload are middleware components supported by Oracle ZFS Storage Appliance.

In the following example, three objects are created in a container called `Various`. A file called `object_to_delete` contains the name of the objects to be deleted. A `curl` command is used to delete those objects.

Multiple file uploads can be accomplished in a similar manner, using the `bulk-upload` option.

```
                     # swift -U swift1 -K swift1 \ -A http://192.168.0.200/v1/export/
MyMusic \ upload --object-name a1 Various aa.txt
a1
# swift -U swift1 -K swift1 \ -A http://192.168.0.200/v1/export/MyMusic \ upload --
object-name a2 Various aa.txt
a2
# swift -U swift1 -K swift1 \ -A http://192.168.0.200/v1/export/MyMusic \ upload --
object-name a3 Various aa.txt
a3
# vi object_to_delete
                     # cat objects_to_delete
Various/a1
Various/a2
Various/a3
# swift -U swift1 -K swift1 \ -A http://192.168.0.200/v1/export/MyMusic/Various list
DabyToure&SkipMcDonald.mp3
a1
a2
a3
aa.txt
# curl -X DELETE -H "X-Auth-Token: $token" \ -T objects_to_delete \ "http://
192.168.0.200/v1/export/MyMusic/Various/?bulk-delete"
Number Deleted: 3
Number Not Found: 0
Response Body:
Response Status: 200 OK
Errors:
# swift -U swift1 -K swift1 \ -A http://192.168.0.200/v1/export/MyMusic/Various list
DabyToure&SkipMcDonald.mp3
aa.txt
#
```

Curl Command-Line Examples

This section shows examples of how to use curl to execute different requests.

Get authorization token

Example input:

```
curl -i http://zfssa:80/auth/v1.0 -X GET -H "X-Auth-User: user" -H "X-Auth-Key:
key"
curl -i http://zfssa:80/auth/v1.0/export/fs1 -X GET -H "X-Auth-User: user" -H "X-
Auth-Key: key"
```

Example output:

```
HTTP/1.1 200 OK
Date: Wed, 04 Apr 2018 15:09:39 GMT
Server: Apache
X-Storage-Url: http://zfssa:80/object/v1/export/fs1
X-Storage-Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f
X-Auth-Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f
Content-Length: 0
X-Trans-Id: tx8f482f16643e495eadee4-0056d45f34
```

> **Note:**
>
> Use X-Storage-Url as the endpoint, and X-Auth-Token for the remainder of the transactions.

Create a container

The following example input creates a container named containerA:

```
curl -i http://zfssa:80/v1/export/fs1/containerA -X PUT -H "Content-Length: 0" -
H "X-Auth-Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f"
```

List containers in plain text format

Example input:

```
curl -i http://zfssa:80/object/v1/export/fs1 -X GET -H "X-Auth-Token:
ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f"
```

Example output:

```
HTTP/1.1 200 OK
Date: Wed, 04 Apr 2018 13:48:57 GMT
Server: Apache
X-Timestamp: 1491400137.51
X-Account-Object-Count: 39
X-Account-Container-Count: 3
X-Account-Bytes-Used: 9823586
X-Trans-Id: tx1fd6145d8b3d4d9fdf0a8-0058e4f5c9
Transfer-Encoding: chunked
Content-Type: text/plain; charset=utf-8

cont1
container2
test2
```

List containers in json format

Example input:

```
curl "http://zfssa:80/v1/export/fs1?format=json" -X GET -H "X-Auth-Token:
ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f" | python -mjson.tool
```

Example output:

```
[
    {
        "bytes": 1664,
        "count": 5,
        "name": "cont1",
        ….
    },
    {
        "bytes": 8902134,
        "count": 25,
        "name": "container2",
        ….
    },
    {
        "bytes": 919788,
        "count": 9,
        "name": "test2",
        …
    }
]
```

List objects inside a container

Example input:

```
curl -i "http://zfssa:80/v1/export/fs1/cont1" -X GET -H "X-Auth-Token:
ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc"
```

Example output:

```
HTTP/1.1 200 OK
Date: Wed, 04 Apr 2018 14:01:16 GMT
X-Container-Bytes-Used: 1664
…
X-Container-Object-Count: 5
…
source.conf
source.conf_cam_rev
source.conf_chena
source.conf_demo
source.sh
]
```

List objects inside a container in json format

Example input:

```
curl "http://zfssa:80/v1/export/fs1/cont1?format=json" -X GET -H "X-Auth-Token:
ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc" | python -mjson.tool
```

Example output:

```
[
    {
        "bytes": 360,
        "content_type": "None",
        "hash": "d3ca560c566fb522e5cb3d923ffd398a",
        "last_modified": "2018-03-08T00:21:18",
        "name": "source.conf"
    },
    {
        "bytes": 229,
        "content_type": "None",
        "hash": "cde5bd3e14dc1a1f6182a09f38b9c7f5",
        "last_modified": "2018-03-08T00:21:12",
        "name": "source.conf_cam_rev"
    },
    {
        "bytes": 223,
        "content_type": "None",
        "hash": "264d93bd075da83fd2f027bf96eb508f",
        "last_modified": "2018-03-08T00:21:15",
        "name": "source.conf_chena"
    },
    {
        "bytes": 375,
        "content_type": "None",
        "hash": "21b1e9e1378a825ac026ced97cccc6b9",
        "last_modified": "2018-03-08T00:21:16",
        "name": "source.conf_demo"
    },
    {
        "bytes": 477,
        "content_type": "None",
        "hash": "2b79330805bd6c6163404b5ec85ea776",
        "last_modified": "2018-03-08T00:21:12",
        "name": "source.sh"
    }
]
```

Upload a local file image.jpg to a container

Example input:

```
curl -i http://zfssa:80/v1/export/fs1/cont1/image.jpg -X PUT -H "X-Auth-Token: 
ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc" -T image.jpg
```

Example output:

```
HTTP/1.1 201 Created
Date: Wed, 04 Apr 2018 14:24:17 GMT
Etag: 93e67dc3fc447b9a368dafa03cbf4aa7
```

Delete an object from a container

Example input:

```
curl -i http://zfssa:80/v1/export/fs1/cont1/image.jpg -X DELETE -H "X-Auth-
Token: ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc"
```

Example output:

```
HTTP/1.1 204 No Content
Date: Wed, 04 Apr 2018 14:27:34 GMT
…
```

Download an object to the local filesystem

Example input:

```
curl  http://zfssa:80/v1/export/fs1/cont1/image.jpg -X GET -H "X-Auth-Token:
ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc" -O
```

Set user metadata to an object

A user's metadata can be done as part of a PUT operation when uploading the object, or a POST operation (updating an existing object). In this example, the user's metadata is tagged as `movie` and the value is `comedy`.

Example input:

```
curl -i http://zfssa:80/v1/export/fs1/cont1/source.conf_demo -X POST -H "X-Object-Meta-
Movie: comedy” -H "X-Auth-Token: ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc"
```

Get metadata of an object

Getting metadata of an object does not download the content of the object.

Example input:

```
curl -i http://zfssa:80/v1/export/fs1/cont1/source.conf_demo -I -H "X-Auth-Token:
ZFSSA_efcd2042-1a05-11e7-baf9-80144f20c6bc"
```

Example output:

```
HTTP/1.1 200 OK
Content-Length: 375
…
X-Object-Meta-MOVIE: comedy
…
```

Copy an object from one container to another container

In this example, `image.jpg` is copied from container `cont1` to container `cont2` on the same account.

Example input:

```
curl -i -X PUT -H "Content-Length: 0" -H "X-Auth-Token: ZFSSA_efcd2042-1a05-11e7-
baf9-80144f20c6bc" -H "X-Copy-From: /cont1/image.jpg" http://zfssa:80/v1/export/fs1/
cont2/image.jpg
```

Example output:

```
HTTP/1.1 201 Created
Date: Wed, 04 Apr 2018 15:24:21 GMT
X-Object-Meta-MTIME: 1491401363.904874
X-Copied-From: cont1/image.jpg
…
```

```
Etag: 93e67dc3fc447b9a368dafa03cbf4aa7
X-Copy-From-Account: user
X-Copied-From-Last-Modified: 1491402764
…
```

Versioning Objects

The following examples show how to create a container to archive objects, enable versioning on that container, upload several versions of an object, and then show the archive for the container.

Create a container to archive objects:

```
curl -i http://zfssa:80/v1/export/fs1/archive -X PUT -H "Content-Length: 0" -H
"X-Auth-Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f"
```

Enable versioning on a container:

```
curl -i http://zfssa:80/v1/export/fs1/container1 -X PUT -H "Content-Length: 0" -
H "X-Auth-Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f" -H "X-Versions-
Location: archive"
```

Upload object to container `container1`:

```
echo xx > example.txt
curl -i http://zfssa:80/v1/export/fs1/container1/example.txt -X PUT -H "X-Auth-
Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f" -T example.txt
```

Modify object and upload it again:

```
echo yy > example.txt
curl -i http://zfssa:80/v1/export/fs1/container1/example.txt -X PUT -H "X-Auth-
Token: ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f" -T example.txt
```

List `container1` content:

```
curl -i http://zfssa:80/v1/export/fs1/container1 -H "X-Auth-Token:
ZFSSA_4607de64-def4-11e5-9326-7b36bda5fa3f"
HTTP/1.1 200 OK
.....
X-Container-Object-Count: 1
X-Versions-Location: archive
....
f.txt
```

List archive content:

```
curl -i http://zfssa:80/v1/export/fs1/archive -H "X-Auth-Token: ZFSSA_4607de64-
def4-11e5-9326-7b36bda5fa3f"
X-Container-Bytes-Used: 33
...
X-Container-Object-Count: 2
...
005example.txt/1493210484.16
005example.txt/1493210812.52
```

List archive content using Swift:

```
swift -A http://zfssa:80/v1/export/fs1/archive  -U user -K key list -l archive
          15 2018-04-26 12:41:24 f.txt
```

```
18 2018-04-26 12:46:52 f.txt
33
```