

OpenStack Block Storage API in CEE

Cloud Execution Environment

INTERWORK DESCRIPTION

Copyright

© Ericsson AB 2016–2018. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

1	Introduction	1
1.1	API Versions	1
1.2	Document References	1
2	Supported Operations	2
2.1	Basic OpenStack Operations	2
2.2	OpenStack Extensions	2
3	Ericsson Extensions	3
4	Limitations and Recommendations	4
4.1	General Limitations	4
4.2	Limitations with EMC ScaleIO	5
4.3	Recommendations	11
5	Additional Information	12
5.1	General Information	12
5.2	EMC ScaleIO-specific Information	13





1 Introduction

This document is an introduction to the use of the OpenStack Application Programming Interface (API) component “block storage” in the Cloud Execution Environment (CEE).

While the aim of the document is primarily to present the block storage API in CEE, it also contains descriptive information about CEE.

In this document, the term Logical Unit Number (LUN) is used both for the logical unit number and for the logical unit.

1.1 API Versions

OpenStack Block Storage API v3 is supported and recommended. Block Storage API v3 is functionally identical to Block Storage API v2.

OpenStack Block Storage API v2 is supported.

OpenStack Block Storage API v1 is supported, but deprecated.

1.2 Document References

This section lists the official OpenStack API references.

1.2.1 API Design Base Reference

For the description of the API operations and extensions of block storage, refer to the section Block Storage API v2 in the [OpenStack API Complete Reference](#).

Note: The date on the front page shows the date on which the PDF document was generated. The date of the latest change of the actual content can be different.

This is a stored copy of the OpenStack API Reference document version that was the base for the development of this CEE version. All references to the OpenStack API made in this document are based on this specific document version. The document can include API extensions that are not supported in this CEE version. Refer to Section 2.2 on page 2 for details.

The OpenStack Block Storage API in CEE is based on OpenStack Mitaka.



2 Supported Operations

The following sections contain the API operations and API extensions that are supported in CEE.

2.1 Basic OpenStack Operations

For the detailed description of basic block storage API operations, refer to the section Block Storage API v2 in the [OpenStack API Complete Reference](#).

2.1.1 Limitations

For CEE-specific limitations and recommendations, see Section 4 on page 4.

2.2 OpenStack Extensions

No OpenStack API extensions are used in CEE.



3 Ericsson Extensions

There are no Ericsson API extensions that are specific to CEE.



4 Limitations and Recommendations

This section describes CEE specific limitations and recommendations.

4.1 General Limitations

Note: If EMC² ScaleIO Storage solution is used, other limitations also apply. Refer to Section 4.2 on page 5.

4.1.1 OpenStack Block Storage Volumes Only Supported with EMC ScaleIO

OpenStack Block Storage volumes are only supported when an EMC ScaleIO is present in the system. The corresponding OpenStack Block Storage services are active even if an EMC ScaleIO is not present.

4.1.2 OpenStack Block Storage Volume Migration Not Possible

The migration of OpenStack Block Storage volume between different storage back ends is not possible, as configuration of multiple storage back ends is not possible, see Section 4.1.3 on page 4.

4.1.3 Multiple Storage Back-End Configuration Not Possible

The configuration of multiple storage back ends is not possible.

4.1.4 Quota Sets Extension Do Not Validate Tenant-ID

OpenStack Block Storage quota management commands (`quota-show`, `quota-update`, `quota-defaults`, `quota-usage`) require a `tenant_id` as command argument. These commands can also be executed with a `tenant_name` as command argument without receiving an appropriate error message, but in this case the desired effect is not achieved.

If a name or ID is given that does not correspond to a tenant in Keystone, the command does not show an appropriate error message. When updating the quotas for a non-existing tenant, the new quotas are stored in the OpenStack Block Storage database where the given `tenant_id` is used as `project_id`.

Refer to the section Block Storage API v2 in the [OpenStack API Complete Reference](#) for further information on quota sets extension.



4.1.5 Multiple Storage Pool Configuration

By default one storage pool is configured for one Cinder back end, so that only that storage pool is used.

4.1.6 Trust Not Supported

Trust is not supported by 3PP in the current version of Cinder. If the duration of a Cinder operation is longer than the configured token life span, the token expires and the operation fails. For example, the upload of a large volume file to image can take more than 3600 seconds, which is the default token life span.

To update the token life span, do the following:

Note: This workaround is not recommended as it increases the risk of leaking sensitive information. The security risks of performing this procedure are the responsibility of the user. Only perform the procedure if security risks are known.

1. Log on to one of the vCICs using SSH. For more information, refer to the *CEE Connectivity User Guide*.
2. In the `/etc/keystone/keystone.conf` configuration file under the token section, update the `expiration` parameter to the required duration in seconds. The default value is 3600.
3. Restart the keystone service by executing the following command:

`service apache2 restart`
4. Repeat the procedure on each vCIC.

4.1.7 Cinder Backup Storage Back End

Only Swift or external Network File System (NFS) storage back end is supported for Cinder backup in CEE. Using the service with any other type of back-end store requires further SI activity.

4.1.8 Parallel Backup and Restore of OpenStack Block Storage Volumes

If an external NFS storage back end is configured for Cinder backup, parallel backup and restore operations are not recommended. Parallel invocation of the Cinder backup API can result in a corrupted Cinder database, or backup volumes locked in error state.

4.2 Limitations with EMC ScaleIO

This section list limitations that apply if EMC² ScaleIO Storage solution is used, in addition to the limitations listed in Section 4.1 on page 4.



4.2.1 Supported operations

The ScaleIO Cinder driver supports the following operations:

- Create volume
- Delete volume
- Attach volume
- Detach volume
- Create snapshot
- Delete snapshot
- Create volume from snapshot
- Copy image to volume
- Copy volume to image
- Extend volume

4.2.2 Volume Limitations

ScaleIO volume size is limited to a basic granularity of 8 GiB. If volume size is not a multiple of 8 GiB, the size is rounded up. For example: a request to create a volume of 10 GiB creates a volume of 16 GiB. In OpenStack, the volume is displayed with its requested size (10 GiB) instead of its actual size (16 GiB). The limitation also applies to the printout of the command `cinder quota-usage <tenant_id>`.

4.2.3 Snapshots

The ScaleIO storage system enables the user to take snapshots of existing volumes, up to 31 per volume. The number of volumes and snapshots combined belonging to one volume tree is limited to a maximum of 32.

For example, the operations `create volume > snapshot volume > create volume from snapshot` use ScaleIO snapshots all the way, that is, the second volume created from the snapshot does not start a new volume tree, all volumes belong to the same, original volume tree.

4.2.4 Power Cycle

This limitation is applicable, if `/var/lib/glance/` is migrated to the ScaleIO storage area on a vCIC. If all nodes in the CEE Region are abruptly powered off or a power failure occurs, follow the controlled power sequence steps described in this section to recover the system.



Do the following on each affected vCIC:

1. Verify whether the directories are under `/var/lib/glance`. If any of the “nodes”, “conversions” or “images” directories does not exist, it indicates that the mounting of the external storage device was not successful.

Example:

```
root@cic-1:/var/lib/glance# ll
total 12
drwxr-xr-x 3 glance root 4096 Nov 8 11:00 ./
drwxr-xr-x 79 root root 4096 Nov 2 04:59 ../
drwxr-x--- 5 glance glance 4096 Nov 8 11:00 image-cache/

root@cic-1:~# findmnt /var/lib/glance
TARGET SOURCE FSTYPE OPTIONS
/var/lib/glance /dev/dm-3[/var/lib/glance] ext4
rw,relatime,errors=panic,data=ordered
```

Note: If the output of `findmnt` does not display `/var/lib/glance/dev/mapper/image-glance xfs rw,relatime,attr2,inode64,noquota`, then the mounting of external storage device was not successful.

2. Verify the connectivity of the ScaleIO nodes.

Note: The following substeps are only applicable to the managed ScaleIO configuration.

- a. Verify the connectivity of the ScaleIO nodes with the `fuel node` command on vFuel:

Example:

```
root@fuel ~# fuel node
id | status | name | cluster | ip |
---+-----+-----+-----+---
4 | ready | scaleio-0-6 | 1 | 192.168.0.24 |
3 | ready | scaleio-0-4 | 1 | 192.168.0.25 |
2 | ready | scaleio-0-5 | 1 | 192.168.0.23 |
10 | ready | compute-0-2 | 1 | 192.168.0.30 |
7 | ready | cic-3 | 1 | 192.168.0.29 |
1 | ready | compute-0-1 | 1 | 192.168.0.20 |
9 | ready | cic-1 | 1 | 192.168.0.27 |
5 | ready | compute-0-3 | 1 | 192.168.0.26 |
8 | ready | cic-2 | 1 | 192.168.0.28 |

mac | roles | pending_roles | online | group_id
-----+-----+-----+-----+-----
00:17:a4:77:00:0a | base-os, scaleio | | 1 | 1
00:17:a4:77:00:06 | base-os, scaleio | | 1 | 1
00:17:a4:77:00:08 | base-os, scaleio | | 1 | 1
```



```
00:17:a4:77:00:02 | compute, virt | | 1 | 1
d2:54:90:63:b8:46 | controller, mongo | | 1 | 1
00:17:a4:77:00:00 | compute, virt | | 1 | 1
22:4a:e5:5f:60:46 | controller, mongo | | 1 | 1
00:17:a4:77:00:04 | compute, virt | | 1 | 1
36:c7:00:2b:1c:44 | controller, mongo | | 1 | 1
```

- b. Check the connectivity to the ScaleIO nodes from affected vCICs.

Example:

```
root@cic-1:~# ping scaleio-0-4
PING scaleio-0-4.domain.tld (192.168.2.21)
56(84) bytes of data.
64 bytes from scaleio-0-4.domain.tld (192.168.2.21):
  icmp_seq=1 ttl=64 time=1.19 ms
64 bytes from scaleio-0-4.domain.tld (192.168.2.21):
  icmp_seq=2 ttl=64 time=0.116 ms
64 bytes from scaleio-0-4.domain.tld (192.168.2.21):
  icmp_seq=3 ttl=64 time=0.114 ms
64 bytes from scaleio-0-4.domain.tld (192.168.2.21):
  icmp_seq=4 ttl=64 time=0.108 ms
```

- c. If SSH connectivity is successful, check if the Meta Data Manager (MDM) cluster is formed.

Example:

```
root@fuel ~# ssh scaleio-0-4
root@scaleio-0-5:~# scli --query_cluster
Cluster:
Mode: 3_node, State: Normal, Active: 3/3, Replicas: 2/2
Virtual IPs: N/A
Master MDM:
Name: scaleio-0-4, ID: 0x31d83e663b9d8390
IPs: 192.168.11.22, 192.168.12.22,
  Management IPs: 192.168.2.22,
  Port: 9011, Virtual IP interfaces: N/A
Version: 2.0.13000
Slave MDMs:
Name: scaleio-0-5, ID: 0x4b6cfbca75b11b31
IPs: 192.168.11.21, 192.168.12.21,
  Management IPs: 192.168.2.21,
  Port: 9011, Virtual IP interfaces: N/A
Status: Normal, Version: 2.0.13000
Tie-Breakers:
Name: scaleio-0-6, ID: 0x7f66703f1d7792a2
IPs: 192.168.11.23, 192.168.12.23, Port: 9011
Status: Normal, Version: 2.0.13000
```



3. Verify if the volumes of the image store are accessible:

```
/opt/emc/scaleio/sdc/bin/drv_cfg --query_vols
```

If there are no returned volumes, that indicates a ScaleIO connectivity issue. In this case, contact next level of support.

Example:

```
root@cic-1:~# /opt/emc/scaleio/sdc/bin/drv_cfg⇒
--query_vols
Retrieved 1 volume(s)
VOL-ID 88bc963f00000000 MDM-ID 6877a55955944ad1
```

4. After the ScaleIO cluster is formed, connect to all affected vCICs and execute the below commands:

- a. Stop Cinder services:

Note: As Cinder service is managed by Pacemaker, this step needs to be executed on one vCIC.

```
root@cic-1:~# crm resource stop p_cinder-volume
```

- b. Stop Glance services:

```
root@cic-1:~# for service in `ls⇒
/etc/init.d/glance-*`; do $service stop; done
```

```
glance-api stop/waiting
glance-glare stop/waiting
glance-registry stop/waiting
```

- c. Stop Swift services:

```
root@cic-1:~# for swift_service in ⇒
swift-account-auditor swift-account ⇒
swift-account-reaper swift-account-replicator ⇒
swift-container-auditor swift-container ⇒
swift-container-reconciler swift-container-replicator ⇒
swift-container-sync swift-container-updater ⇒
swift-object-auditor swift-object ⇒
swift-object-reconstructor swift-object-replicator ⇒
swift-object-updater swift-proxy; ⇒
do service $swift_service stop; done

swift-account-auditor stop/waiting
swift-account stop/waiting
swift-account-reaper stop/waiting
swift-account-replicator stop/waiting
swift-container-auditor stop/waiting
```



```
swift-container stop/waiting
swift-container-reconciler stop/waiting
swift-container-replicator stop/waiting
swift-container-sync stop/waiting
swift-container-updater stop/waiting
swift-object-auditor stop/waiting
swift-object stop/waiting
swift-object-reconstructor stop/waiting
swift-object-replicator stop/waiting
swift-object-updater stop/waiting
swift-proxy stop/waiting
```

- d. Mount the remote external storage:

```
root@cic-1:~# mount -a
```

- e. Check if the remote image store device is mounted correctly.

Example:

```
root@cic-1:~# findmnt /var/lib/glance
TARGET SOURCE FSTYPE OPTIONS
/var/lib/glance /dev/dm-4[/var/lib/glance]
ext4 rw,relatime,errors=panic,data=ordered
/var/lib/glance /dev/mapper/image-glance
xfs rw,relatime,attr2,inode64,noquota
```

- f. Check if the mounted /var/lib/glance contains all subdirectories:

```
root@cic-1:~# ls -l /var/lib/glance/
```

Example output:

```
root@cic-1:~# ls -l /var/lib/glance/
total 0
drwxr-xr-x 2 cinder cinder 6 Dec 2 05:48 conversion
drwxr-xr-x 5 glance glance 64 Dec 2 01:30 image-cache
drwxr-xr-x 2 glance glance 6 Oct 6 18:28 images
drwxr-xr-x 4 swift swift 22 Dec 2 01:08 node
```

- g. Start Swift services by executing the below command on each vCIC:

```
root@cic-1:~# for swift_service in ⇒
swift-account-auditor swift-account ⇒
swift-account-reaper swift-account-replicator ⇒
swift-container-auditor swift-container ⇒
swift-container-reconciler swift-container-replicator ⇒
swift-container-sync swift-container-updater ⇒
swift-object-auditor swift-object ⇒
swift-object-reconstructor ⇒
```



```
swift-object-replicator swift-object-updater =>
swift-proxy; do service $swift_service start; done
```

```
swift-account-auditor start/running, process 23172
swift-account start/running, process 23188
swift-account-reaper start/running, process 23197
swift-account-replicator start/running, process 23206
swift-container-auditor start/running, process 23215
swift-container start/running, process 23224
swift-container-reconciler start/running, process 23233
swift-container-replicator start/running, process 23243
swift-container-sync start/running, process 23252
swift-container-updater start/running, process 23263
swift-object-auditor start/running, process 23272
swift-object start/running, process 23281
swift-object-reconstructor start/running, process 23292
swift-object-replicator start/running, process 23307
swift-object-updater start/running, process 23320
swift-proxy start/running, process 23336
```

- h. Start Glance services by executing the below command on each vCIC:

```
root@cic-1:~# for service in `ls /etc/init.d/glance-*`; do $service start; done
glance-api start/running, process 5787
glance-glare start/running, process 5812
glance-registry start/running, process 5852
```

- i. Start Cinder service by executing the below command on one vCIC:

Note: As Cinder service is managed by Pacemaker, this step needs to be executed on one vCIC.

```
root@cic-1:~# crm resource start p_cinder-volume
```

4.3 Recommendations

Not applicable.



5 Additional Information

5.1 General Information

5.1.1 Quota Sets Extension for Block Storage

To prevent system capacities from being exhausted, OpenStack comes with strict operational limits. For block storage, only 10 volumes and 1000 GB per tenant are allowed by default. You must have administrator rights to apply any changes to the settings and carefully consider system capabilities before doing so. Check the section Block Storage API v2 in the [OpenStack API Complete Reference](#) for further information on limits. To check and change these quotas, use the following commands.

Note: Quotas for tenants can also be managed with Atlas.

View all tenants:

```
openstack project list
```

List the default quotas for a tenant:

```
cinder quota-defaults <tenant_id>
```

Show the current use of a per-tenant quota:

```
cinder quota-usage <tenant_id>
```

Update a particular quota value:

```
cinder quota-update --<quota_name> <new_value> <tenant_id>
```

For example:

```
cinder quota-update --volumes <new_value> <tenant_id>
cinder quota-usage <tenant_id>
```

```
+-----+-----+-----+-----+
|   Type   | In_use | Reserved | Limit |
+-----+-----+-----+-----+
| gigabytes |    0   |    0     | 1000  |
| snapshots |    0   |    0     |   10  |
| volumes   |    0   |    0     |   10  |
+-----+-----+-----+-----+
```




5.2 EMC ScaleIO-specific Information

5.2.1 Provisioning Type

The ScaleIO Cinder driver supports creation of thin provisioned volumes as well as thick provisioning. If the provisioning type is not specified, the default value of thick is used.