ERICSSON

# Swift Store on ScaleIO Activation

## Cloud Execution Environment

OPERATING INSTRUCTIONS

1/1543-CNA 403 3308/5 Uen A

# Contents

# 1 Introduction

This Operational Instruction (OPI) describes how to activate the feature Swift store on ScaleIO. It moves the location of the Swift store from the local disks to the distributed storage EMC² ScaleIO and increases the Swift storage space. For information on how to expand an already existing Swift store on ScaleIO, see *Swift Store on ScaleIO Expansion*.

## 1.1 Description

By default the Swift store is located on the local disks of the CIC hosts. This storage has capacity limitations due to the limited number and capacity of the local disks.

Part of the storage pool used for Cinder located on ScaleIO is therefore used for Swift store. Each Cloud Infrastructure Controller (CIC) gets its own logical volume in the storage pool used by Cinder. This OPI describes the procedure for moving the existing Swift storage from the local disks to ScaleIO without downtime.

## 1.2 Prerequisites

### 1.2.1 Documents

Ensure that the following documents have been read:

- *IP and VLAN Plan*, updated with customer and site-specific values.

  **Note:** All examples in this document use the default values from the document IP and VLAN plan, Reference [1]. The actual customer-defined addresses must be used when performing the steps in this document.

- *CEE Connectivity User Guide*

### 1.2.2 Tools and Equipment

A computer is required, that can be used to connect to the Cloud Execution Environment (CEE) by using Secure Shell (SSH) protocol.

### 1.2.3 Conditions

Before starting this procedure, ensure that the following conditions are met:

- There are no errors reported for ScaleIO (for example, check the watchmen client.
- There are no alarms reported with `Critical` or `Major` severity in the CEE.
- The user of this OPI must be familiar with how to log in to all three CIC nodes and to the Fuel node from a remote location. For more information about these procedures, see the *CEE Connectivity User Guide* and Section 7.1 on page 15 in the Appendix.
- The IdAM credentials for remote CIC login are available. See the *CEE Connectivity User Guide*.
- The credentials for logging in as `ceeadm` user with sudo privileges are available.

  **Note:** All commands in this OPI (except logging in to the CIC node from a remote location) must be executed as user `ceeadm`.

- Ensure that no other maintenance activities are taking place at the same time.

### 1.2.4 Installation Data

Before starting this procedure, make sure that the following data is available:

| Variable | Value |
|---|---|
| *<scaleio.user>* | User name of admin user on ScaleIO system. |
| *<scalio.password>* | Password of admin user on ScaleIO system. |
| *<STORAGE.POOL.NAME>* | This is the name of the storage pool used for Cinder as defined during ScaleIO SW installation. Use the same name that was noted down for the *Configuration File Guide*. |
| *<PROTECTION.DOMAIN.NAME>* | This is the name of the protection domain used for Cinder as defined during ScaleIO SW installations. Use the same name that was noted down for the *Configuration File Guide*. |
| *<size>* | This is the planned initial size of the Swift Store on ScaleIO in GiB. (1). Swift Store size can be expanded as described in the document *Swift Store on ScaleIO Expansion*. |

*(1) Volume sizes on ScaleIO are always multiples of 8GiB. A volume size not matching this constraint is automatically rounded up to the next multiple of 8GiB, but this is not visible in Cinder. Use volume sizes which are multiples of 8GiB.*

## 1.3 Procedure Overview

Figure 1 gives an overview of the procedures covered by this OPI.

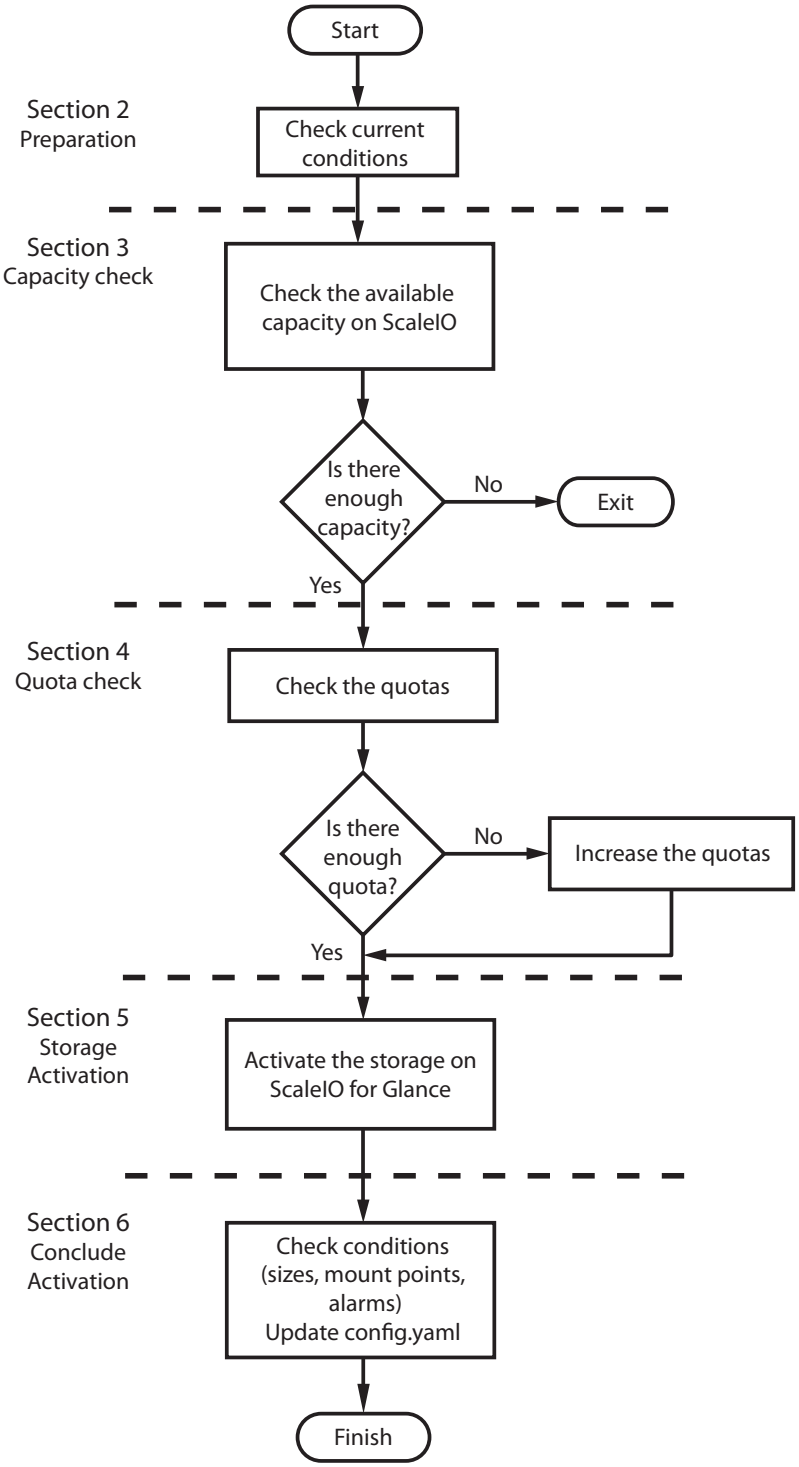


*Figure 1 Procedure Overview*

# 2 Preparation

This section describes how to prepare for creating the LUNs on ScaleIO and moving the Swift store.

Do the following:

1. If ScaleIO activation is performed from a remote location, log on to one of the vCICs using IdAM credentials, then change to user `ceeadm` using the command **`su - ceeadm`**, and log in to the Fuel node. Else, start the procedure from Step 2.

2. From the Fuel node, log on to one of the CIC nodes as user `ceeadm`.

3. Print the properties of the logical volume for the swift store by issuing the following command:

   **`sudo lvdisplay image`**

**Note:** The path is located in the line starting with "LV Path", the size of the swift store is located in the line starting with "LV Size". Note down these values for later use.

Example 1 shows a printout of the **`sudo lvdisplay image`** command.

```
ceeadm@cic-1:~# sudo lvdisplay image
  --- Logical volume ---
  LV Path                /dev/image/glance
  LV Name                glance
  VG Name                image
  LV UUID                crQyfr-r8y9-99qE-Iydo-7LPw-rhso-LDTeEC
  LV Write Access        read/write
  LV Creation host, time ,
  LV Status              available
  # open                 1
  LV Size                649.91 GiB
  Current LE             20797
  Segments               3
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:4
```

*Example 1 Printout of the sudo lvdisplay image Command*

4. Check that the file `/etc/backend_storage_connector.conf` exists, and that it contains the default values as described in Section 7.3 on page 17.

5. Repeat Step 2, Step 3, and Step 4 in Section 2 on page 4 for the other two CIC nodes.

   **Note:** The displayed volume sizes must be the same on all CIC nodes. If they are not equal, note down the highest value and keep it for later use.

6. Continue with the procedures in Section 3 on page 6.

# 3 Check Available Capacity on ScaleIO

This section describes how to verify that there is enough capacity available on ScaleIO to be able to move the Swift store there.

1. Ensure that you are logged on to the Fuel node as `ceeadm` user.

2. Log on to the ScaleIO node on which the Master MDM is running: **`ssh`** ***`<scaleio-node>`***. For details about the identification of the Master MDM node and possible prompts during the process, see Section 7.2 on page 15.

3. Now log on to the Master MDM using the following command:
   **`scli --login --user admin`** ***`<scaleio.user>`*** **`--password`** ***`<scalio.password>`***

4. Check the space available for volume allocation in the storage pool used by the `cinder` service on ScaleIO, by issuing the following command:
   **`scli --query_storage_pool --storage_pool_name`** ***`<STORAGE.POOL.NAME>`*** **`--protection_domain_name`** ***`<PROTECTION.DOMAIN.NAME>`***

   Example 2 shows a partial printout. In Example 2, the storage pool is ‘‘pool1’’ and the space available for allocation is 872GB.

```
ceeadm@scaleio-0-4:~$ scli --query_storage_pool --storage_pool_name pool1\
 --protection_domain_name protection_domain1
Storage Pool pool1 (Id: 3e1c10f900000000) has \
1 volumes and 872.0 GB (892928 MB) available for volume allocation
        The number of parallel rebuild/rebalance jobs: 2
[...]
        1.9 TB (1972 GB) total capacity
        1.7 TB (1759 GB) unused capacity
        0 Bytes snapshots capacity
        16.0 GB (16384 MB) in-use capacity
        0 Bytes thin capacity
        16.0 GB (16384 MB) protected capacity
        0 Bytes failed capacity
[...]
        197.3 GB (202026 MB) spare capacity
        16.0 GB (16384 MB) at-rest capacity
[...]
Volumes summary:
        1 thick-provisioned volume. Total size: 8.0 GB (8192 MB)
[...]
```

*Example 2 Available Capacity on ScaleIO*

**Note:** Although in the printout the available capacity is displayed in ‘‘GBs’’, in fact the correct unit of measure for the displayed capacity is ‘‘GiBs’’.

5. Check if the space available for volume allocation is sufficient and proceed according to the result as follows:

   - If the space available for volume allocation is at least three times as big as the planned new size of the Swift storage for one CIC on ScaleIO, continue with the procedures in Section 4 on page 7. This triple size for the Swift storage must be available on ScaleIO since each CIC must be extended by the same amount of storage capacity.

   - If the space available for volume allocation is less than three times the planned new size of the Swift storage for one CIC on ScaleIO, then the available capacity on ScaleIO must be increased first. This is out of the scope of this document.Refer to manufacturer documentation or contact the next level of maintenance support.

6. Continue with the procedures in Section 4 on page 7.

# 4 Check the OpenStack Quotas

This section describes how to check the OpenStack quotas for the ‘‘admin’’ project and how to expand them if needed.

1. Ensure that you are logged in to one of the CIC nodes as `ceeadm` user.

2. Print the project list by issuing the following command:

   **`openstack project list`**

```
ceeadm@cic-2:/root$ openstack project list
+----------------------------------+----------+
| ID                               | Name     |
+----------------------------------+----------+
| 3052cafca2e14f85b8f02263025d2a8f | admin    |
| c16e42d79bfb406fb4730fa8bdbb6d8f | services |
+----------------------------------+----------+
```

*Example 3 Openstack Project List*

**Note:** Identify the project with the name ‘‘admin’’ and note down its ID (*<project_id>*) for later use.

3. Print the quota usage by issuing the following command:

   **`cinder quota-usage`** ***<project_id>***

   Example 4 is a printout of the following **`cinder quota-usage`** command:
   **`cinder quota-usage 3052cafca2e14f85b8f02263025d2a8f`**

```
ceeadm@cic-1:~# cinder quota-usage ⇒
3052cafca2e14f85b8f02263025d2a8f
+----------------+--------+----------+-------+
|      Type      | In_use | Reserved | Limit |
+----------------+--------+----------+-------+
|      [...]     |  [...] |   [...]  | [...] |
|   gigabytes    |  3100  |    0     | 10000 |
|      [...]     |  [...] |   [...]  | [...] |
|    volumes     |   4    |    0     |  100  |
|      [...]     |  [...] |   [...]  | [...] |
+----------------+--------+----------+-------+
```

*Example 4 quota-usage Command Printout*

**Note:** In the **`cinder`** command printouts the capacity values are given in “GBs”, but in fact the correct unit of measure for the same values are “GiBs”.

4. Verify the quota types `gigabytes` and `volumes` as follows:

   - For the type `volumes`, the difference between `Limit` and `In_use` must be at least 3.

   - For the type `gigabytes`, the difference between `Limit` and `In_use` must be at least three times as big as the intended Swift store size for a single CIC. For example if the intended Swift store size for one CIC is 1000 GiB, the difference between `Limit` and `In_use` must be at least 3000 GiB.

   The example printout in Step 3 in Section 4 on page 7 shows a disposition of 96 volumes and 6900 GiB of capacity.

   After calculating the quotas perform the relevant one of the following actions:

   - If enough quotas are available for the activation, then continue with the procedures in Section 5 on page 10.

   - If the `gigabytes` quota must be increased continue with Step 5 in Section 4 on page 7.

   - If the `volumes` quota must be increased continue with Step 6 in Section 4 on page 7.

5. If the quota `gigabytes` needs to be increased, issue the following command:

   **`cinder quota-update <project_id> --gigabytes`**
   ***`<new_g_limit>`***

   where

   - `<new_g_limit>` must be equal to or greater than:

     [(3*`<size>`)+`<current_use>`].

     Here `<size>` is the planned new size of the Swift storage and `current_use` is the `in_use` value of `gigabytes` from Step 3 in Section 4 on page 7.

   - `<project_id>` is the project ID of the project `admin`.

Example 5 shows how to increase the gigabytes quota by 99, from 10000 GiB to 10099 GiB for the project `admin` and an example printout:

```
ceeadm@cic-1:~# cinder quota-update ⇒
3052cafca2e14f85b8f02263025d2a8f --gigabytes 10099
+-----------------------+-------+
|       Property        | Value |
+-----------------------+-------+
|   backup_gigabytes    |  1000 |
|       backups         |   10  |
|      gigabytes        | 10099 |
| per_volume_gigabytes  |   -1  |
|      snapshots        |   10  |
|       volumes         |  100  |
+-----------------------+-------+
```

*Example 5 cinder quota-update— 'gigabytes'*

6. If the `volumes` quota needs to be increased issue the following command:

   **cinder quota-update <project_id> --volumes <new_vol_limit>**

   where

   - `<new_vol_limit>` is at least three volumes more than the current value,
   - `<project_id>` is the project ID of the project `admin`.

Example 6 shows how to increase the `volumes` quota by 3, from 100 to 103, for the project `admin`:

```
ceeadm@cic-1:~# cinder quota-update 3052cafca2e14f8
5b8f02263025d2a8f --volumes 103
```

*Example 6 cinder quota-update — volumes*

7. Continue with the procedures in Section 5 on page 10.

# 5 Activate the Storage on ScaleIO for Swift

This section describes how to activate the storage on ScaleIO for Swift.

1. Ensure that you are logged in to one of the CIC nodes as `ceeadm` user.
2. Start a `screen` session with the following command:

   **sudo -E screen**

3. Press **Space** or **Return** after you have read the instruction on the screen.

   **Note:** The **`screen`** command starts a session, that is independent from the current terminal window. Even if the terminal window crashes or is ended by other means, it is possible to reconnect to the session by using the command **`sudo -E screen -x`** from another terminal on the same host.

4. To expand and activate the Swift storage on ScaleIO, issue the following command:

   **`backend-storage-connector activate`** *`<path>`*⇒
   *`<size>`*

   The arguments are explained below:

| Variable | Comment |
|---|---|
| *<path>* | This is the path of the logical volume (`LV Path`) noted down in Step 3 in Section 2 on page 4 . |
| *<size>* | This is the planned initial Swift store size on ScaleIO for each CIC in GiB. (1). Bigger Swift Store sizes can be created later, as described in the document *Swift Store on ScaleIO Expansion*.<br><br>Provide a size in GiBs, that is equal to or larger, than the largest value noted down during the preparation in Section 2 on page 4. The activation does not work if a smaller size is specified. (2) |

*(1) Consider that volume sizes on ScaleIO are always multiples of 8GiB. A volume size not matching that rule will be automatically rounded up to the next multiple of 8GiB, but this will not (yet) be reflected in cinder. Therefore make sure to use volume sizes which are multiples of 8GiB.*
*(2) All CIC nodes must have the same storage capacity for Swift. Use the same value of <size> for each CIC.*

   **Note:** The transfer speed itself from the local disks to the distributed storage amounts to approximately 6GiB/min.

   The successful job is indicated by the final status information message `Success.`, as shown in Example 7 :

```
root@cic-1:~# backend-storage-connector activate⇒
 /dev/image/glance 650
[...]
<ts> INFO: Updated fstab with option "_netdev,...
<ts> INFO: Extending volume group: image with...
<ts> INFO: Move device /dev/sda7. This may take a while.
<ts> INFO: Move device /dev/sdb2. This may take a while.
<ts> INFO: Successfully disconected all local...
<ts> INFO: Extending logical volume /dev/imag...
<ts> INFO: Growing xfs filesystem of logical...
<ts> INFO: Success.
root@cic-1:~#
```

*Example 7 Expand and Activate the Swift Store on ScaleIO*

5. If the execution fails, the file `/var/backend-storage-connector.fail` is created. Before another attempt at activation, remove the `.fail` file using the following command:
   **`rm /var/backend-storage-connector.fail`**

6. End the screen session by pressing **CTRL+D**.

7. Repeat all the steps in Section 5 on page 10 for the other two CIC nodes and then continue with the procedures in Section 6 on page 12.

   **Note:** A log file is generated under the following path on each CIC:

   `/var/log/backend-storage-connector.log`

# 6 Conclude Activation

This section describes how to confirm that the activation has taken place.

1. Ensure that you are logged in to one of the CIC nodes as `ceeadm` user.

2. Check the logical volume size for the Swift store by issuing the following command on one of the CIC nodes:

   **`sudo lvdisplay image`**

   `LV Size` displays the size of the Swift store in GiB. The size must match the chosen value for *<size>*.

   Example printout:

```
ceeadm@cic-1:~# sudo lvdisplay image
  --- Logical volume ---
  LV Path                /dev/image/glance
  LV Name                glance
  VG Name                image
  LV UUID                crQyfr-r8y9-99qE-Iydo-7LPw-rhso-LDTeEC
  LV Write Access        read/write
  LV Creation host, time ,
  LV Status              available
  # open                 1
  LV Size                649.97 GiB
  Current LE             22396
  Segments               4
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           252:4
```

**Note:** In the example the `LV Size` of `649.97 GiB` does not match the defined 650GiB exactly, the difference is caused by rounding inaccuracy.

3. Repeat Step 1 and Step 2 in Section 6 on page 12 for the other two CIC nodes and then continue with Step 4.

4. Issue the following command on one of the CIC nodes:

   **`cinder list`**

```
ceeadm@cic-1:~# cinder list
+----------+--------+--------------------------------+------+-[...]-+
|    ID    | Status |          Display Name          | Size | [...] |
+----------+--------+--------------------------------+------+-[...]-+
| 49[...]6 | in-use | CEE+cic-1+/dev/image/glance    | 650  | [...] |
| 8d[...]1 | in-use | CEE+cic-2+/dev/image/glance    | 650  | [...] |
| ac[...]0 | in-use | CEE+cic-3+/dev/image/glance    | 650  | [...] |
+----------+--------+--------------------------------+------+-[...]-+
ceeadm@cic-1:~#
```

*Example 8 cinder list Command Printout*

For each CIC one volume must exist with a `Display Name` starting with `CEE+`, followed by the CIC name and the logical volume path. The size must match the value *<size>*, which has been provided in Section 5 on page 10.

5. From the CIC node log in to the Fuel node by using SSH as user `ceeadm`.

6. On the Fuel node, edit the `config.yaml` file under `/mnt/cee_config` with root privileges (for example with the **`sudo vi /mnt/cee_config/config.yaml`** command) as follows:

a. Under `swift: > swift_on_backend_storage: > activation_mode` change the value from `manual` to `automatic`.

b. Under `swift: > swift_on_backend_storage: > lun_size` set the value to the *<size>* (as it has been set in Step 4 in Section 5 on page 10).

**Note:** These modifications are necessary, so that in case of a rollback and CIC repair, the Swift store is set up again in ScaleIO automatically with the correct size. Once the Swift storage is moved to ScaleIO, it cannot be reversed back to the local disk.

Example 9 shows the relevant `config.yaml` hierarchy.

```
ericsson:
  swift:
    swift_on_backend_storage:
      type: centralized
      activation_mode: automatic
      lun_size: 650GiB
```

*Example 9 Editing config.yaml*

7. Save the changes in the `config.yaml` file.

# Appendix

# 7 Additional Information

This section describes the following:

- How to list the hostnames and addresses of the CIC nodes
- How to identify the Master MDM
- Configurable storage connector parameters

## 7.1 List CIC, Compute and ScaleIO Nodes

To display the hostnames and IP addresses of the CIC nodes, issue the following command on Fuel:

```
sudo fuel node
```

**Note:** From a remote location only the CIC servers can be reached. For more information see the *CEE Connectivity User Guide*.

Example 10 is a partial printout that shows only the relevant information like names and IP addresses:

```
[ceeadm@fuel ~]# sudo fuel node
id | status | name        |...| ip           |...
---+--------+-------------+...+--------------+...
 1 | ready  | scaleio-0-5 |...| 192.168.0.28 |...
 2 | ready  | scaleio-0-4 |...| 192.168.0.23 |...
11 | ready  | cic-3       |...| 192.168.0.32 |...
 4 | ready  | scaleio-0-6 |...| 192.168.0.25 |...
10 | ready  | cic-2       |...| 192.168.0.31 |...
 3 | ready  | compute-0-1 |...| 192.168.0.20 |...
 8 | ready  | compute-0-3 |...| 192.168.0.29 |...
 9 | ready  | cic-1       |...| 192.168.0.30 |...
 5 | ready  | scaleio-0-8 |...| 192.168.0.27 |...
 7 | ready  | compute-0-2 |...| 192.168.0.21 |...
 6 | ready  | scaleio-0-7 |...| 192.168.0.26 |...
```

*Example 10 CIC, Compute and ScaleIO Node Printout*

## 7.2 Identify Master MDM

**Note:** The identification procedure must be performed at each CEE logon as the Master MDM role may move to another ScaleIO node.

It is possible that the Master role moves to another MDM while being logged on. If `scli` commands fail suddenly, repeat the identification procedure.

To identify the Master Meta Data Manager (MDM), the command `scli --query_cluster` must be performed on each ScaleIO node after logon. The Master MDM is identified based on its unique printout.

Do the following:

1. List all Compute, CIC, and ScaleIO nodes as described in Section 7.1 on page 15, and note down all ScaleIO nodes and their respective IP address.

   ScaleIO nodes are identified in the following format: `scaleio-<shelf_id>-<blade_id>`, for example `scaleio-0-4`

2. Log on to the first ScaleIO node using the following command:
   **`sudo ssh <scaleio-node>`**.

   An example of the command is the following:
   **`sudo ssh scaleio-0-4`**

3. If prompted by SSH authentication confirmation request, continue by entering **`yes`** and pressing **Enter**. The following is an example of the authentication request:

```
The authenticity of host 'scaleio-0-4 (192.168.0.23)' can't be established.
ECDSA key fingerprint is b1:09:a8:f3:de:b7:93:c8:37:d4:24:e7:19:b9:d0:45.
Are you sure you want to continue connecting (yes/no)?
```

4. Execute the command **`scli --query_cluster`**.

   If prompted, approve a certificate and add it to the truststore by pressing **y** and **Enter**. An example of the MDM certificate approval is the following:

```
ceeadm@scaleio-0-4:~$ scli --query_cluster
Certificate info:
subject: /GN=MDM/CN=scaleio-0-4.domain.tld/L=Hopkinton/ST=Massachusetts/C=US/O=EMC/OU=ASD
issuer:  /GN=MDM/CN=scaleio-0-4.domain.tld/L=Hopkinton/ST=Massachusetts/C=US/O=EMC/OU=ASD
Valid-From: Oct 20 11:30:47 2016 GMT
Valid-To:   Oct 19 12:30:47 2026 GMT
Thumbprint: BA:2C:17:90:D9:10:47:21:0B:AD:D0:2B:BA:10:62:7C:FE:47:11:74

Press 'y' to approve this certificate and add it to the truststore
```

5. Compare the printout of the command `scli --query_cluster` to the following examples to identify the type of the node.

   The following is an example printout of the Master MDM node in a three-node configuration:

```
ceeadm@scaleio-0-4:~$ scli --query_cluster
```

```
Cluster:
    Mode: 3_node, State: Normal, Active: 3/3, Replicas: 2/2
Master MDM:
    Name: scaleio-0-4, ID: 0x009f453b08513d70
      IPs: 192.168.11.21, 192.168.12.21, Management IPs: 192.168.2.21, Port: 9011
      Version: 2.0.5014
Slave MDMs:
    Name: scaleio-0-5, ID: 0x5afccfcb0cc9b8d1
      IPs: 192.168.11.20, 192.168.12.20, Management IPs: 192.168.2.20, Port: 9011
      Status: Normal, Version: 2.0.5014
Tie-Breakers:
    Name: scaleio-0-7, ID: 0x20089db33308e7f2
      IPs: 192.168.11.25, 192.168.12.25, Port: 9011
      Status: Normal, Version: 2.0.5014
ceeadm@scaleio-0-4:~$
```

The following is an example printout of the Master MDM Node in a five-node configuration:

```
root@scaleio-0-5:~# scli --query_cluster
Cluster:
    Mode: 5_node, State: Normal, Active: 5/5, Replicas: 3/3
    Virtual IPs: N/A
Master MDM:
    Name: scaleio-0-5, ID: 0x0db1e6c34049c130
      IPs: 192.168.11.25, 192.168.12.25, Management IPs: 192.168.2.25, Port: 9011, Virtual IP interfaces: N/A
      Version: 2.0.10000
Slave MDMs:
    Name: scaleio-0-6, ID: 0x7810f20f245ef491
      IPs: 192.168.11.23, 192.168.12.23, Management IPs: 192.168.2.23, Port: 9011, Virtual IP interfaces: N/A
      Status: Normal, Version: 2.0.10000
    Name: scaleio-0-4, ID: 0x1425248c45a38562
      IPs: 192.168.11.24, 192.168.12.24, Management IPs: 192.168.2.24, Port: 9011, Virtual IP interfaces: N/A
      Status: Normal, Version: 2.0.10000
Tie-Breakers:
    Name: scaleio-0-7, ID: 0x789a0a834440baa3
      IPs: 192.168.11.28, 192.168.12.28, Port: 9011
      Status: Normal, Version: 2.0.10000
    Name: scaleio-0-8, ID: 0x2b08640c5f57d344
      IPs: 192.168.11.22, 192.168.12.22, Port: 9011
      Status: Normal, Version: 2.0.10000
```

If the printout is different from the above two examples, the node is not the Master MDM node.

If the node is the Master MDM node, note down the hostname and IP address of the node for further use, based on the printout of Step 1.

If the node is not the Master MDM node, continue with Step 6.

6. Exit the ScaleIO node by entering **`exit`** and pressing **Enter**, and repeat the procedure from Step 2 on another ScaleIO node.

## 7.3 Configurable Storage Connector Parameters

Some storage connector parameters can be configured according to local requirements in the file `/etc/backend_storage_connector.conf`. The configurable parameters and the default values of the parameters are listed in Table 1.

*Table 1 Storage Connector Parameters*

| Parameter | Default value | Description |
|---|---|---|
| *<volume_wait_timeout>* | 1800 | Maximum time to wait in seconds until a created cinder volume has status `available` |
| *<volume_wait_time>* | 10 | Scan interval in seconds checking for volume status |
| *<volume_deletion_timeout>* | 600 | Maximum time to wait in seconds until the deleted cinder volume or volumes are not found anymore using Openstack commmands(1) |
| *<volume_deletion_time>* | 10 | Scan interval in seconds checking for cinder volume deletion |
| *<retry_vol_create_on_error>* | True | If value `True`is set, volumes in error status are deleted and re-created(1)(2) |
| *<log_path>* | /var/log/backend-storage-connector.log | Path to log file |
| *<astute>* | /etc/swift_backend_ astute.yaml | Path to `astute` file |
| *<use_multipath>* | False | This parameter is not applicable for Swift on ScaleIO. |
| *<supported_storage_types>* | scaleio | |
| *<external_path_list>* | /dev/scini, /dev/disk/by-id/emc-vol | |
| *<swift_logical_volume_path>* | /dev/image/glance | |
| *<supported_logical_volume_path>* | /dev/image/glance | |
| | | |

*(1) Used only in repair mode*
*(2) In repair mode volumes are deleted and re-created. Due to the latency of the backend storage system, when deleting volumes, volume creation can fail if the backend storage system has limited capacity. In this case, the created volumes end up in error status. The number of retries and the time interval are automatically calculated depending on the size of the volume.*

# Reference List

[1] *IP and VLAN plan*, 2/102 62-CRA 119 1862/5 Uen