



# NVMesh CLI Guide

2.2.0 — Last update: 26 July 2021

Excelero, Ltd.

# Table of Contents

<b>1. Copyright and Trademark Information .....</b>	<b>2</b>
<b>2. Preface .....</b>	<b>3</b>
<b>3. Introduction .....</b>	<b>4</b>
<b>4. Installation .....</b>	<b>5</b>
4.1. Supported Environments .....	5
4.2. Installation & Start .....	5
<b>5. Using the NVMesh CLI .....</b>	<b>6</b>
5.1. Prerequisites .....	6
5.2. nvmesh CLI Files .....	7
5.3. Interactive vs CLI .....	7
5.4. Command Structure .....	8
5.5. Help .....	9
<b>6. Command Reference .....</b>	<b>11</b>
6.1. exit .....	11
6.2. logout .....	11
6.3. login .....	11
6.4. define-ssh .....	12
6.5. version .....	12
6.6. client .....	12
6.6.1. attach .....	12
6.6.2. count .....	13
6.6.3. delete .....	13
6.6.4. detach .....	14
6.6.5. show .....	14
6.7. cluster .....	15
6.7.1. show .....	15
6.7.2. shut-down .....	16
6.8. drive .....	16
6.8.1. show .....	16
6.8.2. delete .....	17
6.8.3. evict .....	17
6.8.4. format .....	18
6.8.5. add-arbiter .....	18
6.8.6. remove-arbiter .....	18
6.8.7. include-nvme .....	19
6.8.8. exclude-nvme .....	19
6.9. driveclass .....	19

6.9.1. create .....	20
6.9.2. delete.....	20
6.9.3. show .....	20
6.9.4. update .....	21
6.10. target .....	22
6.10.1. count.....	22
6.10.2. delete.....	22
6.10.3. delete-nic .....	22
6.10.4. show .....	23
6.11. targetclass .....	24
6.11.1. create .....	24
6.11.2. delete.....	24
6.11.3. show .....	25
6.11.4. update.....	25
6.12. volume .....	26
6.12.1. create .....	26
6.12.2. delete.....	28
6.12.3. rebuild.....	29
6.12.4. show .....	29
6.12.5. update.....	30
6.13. vpg.....	31
6.13.1. create .....	32
6.13.2. delete.....	33
6.13.3. show .....	33
6.14. mongo-db.....	34
6.14.1. show .....	34
6.15. volume-security-group .....	35
6.15.1. create .....	35
6.15.2. show .....	35
6.15.3. update.....	36
6.15.4. delete.....	37
6.16. key-pair .....	37
6.16.1. download .....	37
6.16.2. create .....	38
6.16.3. show .....	38
6.16.4. update.....	39
6.16.5. delete.....	39
6.17. log.....	39
6.17.1. acknowledge-all-alerts .....	40
6.17.2. acknowledge .....	40
6.17.3. show .....	40

<b>7. Document Reference .....</b>	<b>44</b>
------------------------------------	-----------

# 1. Copyright and Trademark Information

---

© 2015-2021 Excelero, Inc. All rights reserved. Specifications are subject to change without notice. Excelero, the Excelero logo, Remote-Direct-Drive-Access (RDDA) and MeshProtect are trademarks Excelero, Inc. in the United States and/or other countries. NVMesh® is a registered trademark of Excelero, Inc. in the United States.

All other brands or products are trademarks or registered trademarks of their respective holders and should be treated as such.

## 2. Preface

---

Excelero™ creates innovative, high performance storage solutions that accelerate business applications and deliver outstanding return on investment with the lowest cost of ownership. The NVMesh® software defined block storage product offers the performance of local flash with the convenience, efficiency and redundancy of an all-flash-array. For details, go to: [www.excelero.com](http://www.excelero.com).

This document describes the command-line interface of the Excelero NVMesh storage solution and accompanying command-line utilities. For more information on NVMesh refer to [NVMesh User Guide](#).

### AUDIENCE

The primary audience for this document is intended to be storage and/or application administration personnel responsible for installing and deploying the Excelero NVMesh product.

### NON-DISCLOSURE REQUIREMENTS

© Copyright 2015-2021 Excelero, Inc. All rights reserved. This document contains the confidential and proprietary information of Excelero, Inc. Do not reproduce or distribute without the prior written consent of Excelero.

### FEEDBACK

We continually try to improve the quality and usefulness of Excelero documentation. If you have any corrections, feedback, or requests for additional documentation, send an e-mail message to [support@excelero.com](mailto:support@excelero.com)

### INFORMATION ABOUT THIS DOCUMENT

All information about this document including typographical conventions, references, and a glossary of terms can be found in the [Document Reference Section](#).

# 3. Introduction

---

The `nvmesh` CLI tool provides a command-line user interface to manage NVMesh. This interface can be used to send one-line management commands to NVMesh or write shell scripts. Additionally, it offers an interactive shell.

`nvmesh` uses the NVMesh RESTful API, terminal command line tools and `ssh` for day-to-day management and provisioning activities with homogeneous semantics.

## 4. Installation

---

- [Supported Environments](#)
- [Installation Requirements](#)
- [Installation & Start](#)

### 4.1. Supported Environments

---

Any Linux distribution with GLIBC >= 2.12.

### 4.2. Installation & Start

---

1. Install the **nvmesh-utils** package. (Available from the Excelero NVMesh yum/apt repo.)
2. To start the NVMesh-shell tool, simply run/type: `nvmesh` in your terminal window.

## 5. Using the NVMesh CLI

---

Initially, `nvmesh` doesn't know anything about the NVMesh environment and no credentials are set. The tool requires NVMesh management / API login information (administrative account) and if there is no pre-shared SSH key set up with all the involved hosts, servers and clients, the root SSH credential is required as well. The easiest and quickest way to configure the required credentials is to launch `nvmesh` and run the `check cluster` command:

```
$ nvmesh check cluster
```

The tool will ask will ask for the SSH credentials where you can choose between `sudo` and `root`.

To use `sudo` for SSH:

```
nvmesh # define sshuser
Do you require sudo for SSH remote command execution? [Yes|No] :y
Please provide the user name to be used for SSH connectivity: <your username>
Please provide the SSH password:
```

To use `root` for SSH:

```
nvmesh # define sshuser
Do you require sudo for SSH remote command execution? [Yes|No] :n
Please provide the root level SSH user name: root
Please provide the SSH password:
```

If preshared keys are set up throughout, leave the password prompt empty and just hit enter. There is no need to provide a password if preshared keys for the root level user are set up. Then it will ask for the NVMesh API user credentials and the management server to be used.

The API user and password, and the SSH user and password are stored under the users home directory. Passwords are stored encoded and obfuscated as additional protection. In addition, the NVMesh management server information is stored in the users home directory.

### 5.1. Prerequisites

---

Two configurations should be made in order to use to CLI for the first time:

1. Configuring the `nvmesh.conf` file to the management we want to work with and save it to: `/etc/opt/NVMesh/nvmesh.conf`.

If the **nvmesh-core** package is installed on the machine, the file will be present, otherwise, it should be created under the mentioned path and include the following content:



```
# NVMesh configuration file
# This configuration file is utilized by Excelero NVMesh(tm) applications for various options.

# Define the management protocol
# MANAGEMENT_PROTOCOL="<https/http>"
# Example
# MANAGEMENT_PROTOCOL="https"

MANAGEMENT_PROTOCOL="https"

# Define the location of the NVMesh Management Websocket servers
# MANAGEMENT_SERVERS="<server name or IP>:<port>,<server name or IP>:<port>,..."
# Example:
# MANAGEMENT_SERVERS="nvmesh-management1:4001,nvmesh-management2:4001"

MANAGEMENT_SERVERS="localhost:4001"
```

2. When you first try to use the CLI/shell you will be prompted to provide a valid user name and password for the *management server* API, the CLI will authenticate the user, then you will not be prompted again unless you logout from the CLI/shell.

Also, you will need to provide valid SSH credentials for performing attach/detach operations on a remote client, you can change those credentials in the future using the `define-ssh` command.

The API and SSH credentials will be stored in the user's home directory under `.nvmesh_cli_files/`, see the `nvmesh` CLI Files section for more details.

## 5.2. nvmesh CLI Files

<code>/etc/opt/NVMesh/nvmesh.conf</code>	Stores the NVMesh management server name
<code>~/.nvmesh_cli_files/nvmesh_api_secrets</code>	Stores the API username and password
<code>~/.nvmesh_cli_files/nvmesh_cli_history</code>	Stores the NVMesh shell cli tool command history.
<code>~/.nvmesh_cli_files/nvmesh_ssh_secrets</code>	Stores the SSH user information

## 5.3. Interactive vs CLI

All of the tool's capabilities are available in two modes: Interactive and CLI.

### CLI Mode

To use the CLI mode, just invoke `nvmesh` with all the commands and options you need to complete an action such as this example:

```
nvmesh client attach -c client1 -v volume1
```

## Interactive Mode

To use the interactive mode just type: `nvmesh` (with no additional arguments)

### Interactive mode features:

1. Use the '!' prefix to execute shell commands locally:

```
> !date
Tue Apr 29 19:08:48 IDT 2019
```

2. Get auto-completion by hitting tab:

```
> volume create
--name      Name of the volume. The name must be unique, as it will become the ID of the volume.
-n          Name of the volume. The name must be unique, as it will become the ID of the volume.
--raid-level The RAID level of the volume. Options: lvm = Concatenated, ec = Erasure Coding, 0 = Striped RAID-0, 1 = Mirrored RAID-1, 10 = Striped & Mirrored RAID-10.
-rl         The RAID level of the volume. Options: lvm = Concatenated, ec = Erasure Coding, 0 = Striped RAID-0, 1 = Mirrored RAID-1, 10 = Striped & Mirrored RAID-10.
--capacity  Space in bytes to allocate for the volume. Use "MAX" for using all of the available space.
-c          Space in bytes to allocate for the volume. Use "MAX" for using all of the available space.
--description Description of the volume.
```

3. Traverse and search the shell history using up, down arrows and Ctrl + r respectively:

```
(reverse-i-search)`vol`: volume create --name v3 --raid-level 0 --target-classes rc --capacity 1000000000000 --stripe-width 2 --stripe-size 32
```

## 5.4. Command Structure

The full command structure is as follows:

Typing `nvmesh --help` will provide the first level of the available commands:

```
Usage: nvmesh [OPTIONS] COMMAND [ARGS]...
```

For interactive mode run 'nvmesh' without any additional commands. While in interactive mode you can use '!' prefix to execute shell commands, traverse and search the CLI history using 'up', 'down' arrows and 'Ctrl+r' respectively, and auto-complete commands by hitting 'tab'.

#### Options:

```
--help  Show this message and exit.
```

#### Commands:

```
client      Group Clients related operations
cluster     Group Cluster related operations
define-ssh  Define new SSH credentials
drive       Group Drives related operations
driveclass  Group Drive Classes related operations
```

```

logout      Logout the current user from the CLI
target      Group Targets related operations
targetclass Group Target Classes related operations
version     Show nvmesh CLI version
volume      Group Volumes related operations
vpg         Group VPGs related operations

```

All of the commands (except for `logout`, `define-ssh` and `version`) are NVMesh entities.

For every entity, there is a second level of commands that are operations on that entity. Typing `nvmesh targetclass --help` will provide us with the target class operations:

```
Usage: nvmesh targetclass [OPTIONS] COMMAND [ARGS]...
```

```
Group Target Classes related operations
```

```
Options:
```

```
--help Show this message and exit.
```

```
Commands:
```

```

create Create a target class.
delete Delete target classes.
show   Show all target classes.
update Update a target class.

```

## 5.5. Help

Typing `--help` with any combination of commands will provide a help screen with the optional commands/arguments for that command.

For example:

```
vpg create --help
```

```
Usage: nvmesh vpg create [OPTIONS]
```

```
Create VPGs.
```

```
usage example: -n v11 --raid-level ec -c 1000000000000 --data-blocks 2
--parity-blocks 1 --protection-level full --stripe-width 1
```

```
Options:
```

```

-n, --name TEXT          Name of the volume. The name must be unique,
                        as it will become the ID of the volume.
                        [required]

```

```

-rl, --raid-level [lvm|ec|0|1|10]
                        The RAID level of the volume. Options: lvm =
                        Concatenated, ec = Erasure Coding, 0 =
                        Striped RAID-0, 1 = Mirrored RAID-1, 10 =
                        Striped & Mirrored RAID-10. [required]

```

<code>-c, --capacity INTEGER</code>	Space in bytes to allocate for the volume.
<code>-d, --description TEXT</code>	Description of the volume.
<code>--domain TEXT</code>	Domain to use.
<code>-dc, --drive-classes TEXT</code>	Limit volume allocation to specific drive classes.
<code>-tc, --target-classes TEXT</code>	Limit volume allocation to specific target classes.
<code>--stripe-width INTEGER</code>	Number of disks to use. Required if RAID Level is 0 or 10.
<code>--data-blocks INTEGER RANGE</code>	Number of disks to use. Required if RAID Level is ec.
<code>--parity-blocks INTEGER RANGE</code>	Number of disks to use. Required if RAID Level is ec.
<code>--protection-level [full minimal ignore]</code>	Protection level to use. Required if RAID Level is ec. Options: full = Full Separation, minimal = Minimal Separation, ignore = Ignore Separation.
<code>--help</code>	Show this message and exit.

## 6. Command Reference

---

First level commands:

- [logout](#)
- [define-ssh](#)
- [client](#)
- [cluster](#)
- [drive](#)
- [driveclass](#)
- [target](#)
- [targetclass](#)
- [volume](#)
- [vpg](#)

### 6.1. exit

---

Usage: nvmesh exit [OPTIONS]

Exit interactive mode

Options:

-h, --help Show this message and exit.

### 6.2. logout

---

Usage: nvmesh logout [OPTIONS]

Logout the current user from the CLI

Options:

--help Show this message and exit.

### 6.3. login

---

Usage: nvmesh login [OPTIONS]

Define new API credentials, previous credentials will be overridden.

Options:

-u, --user TEXT API user name

-p, --password TEXT API password

```
-h, --help      Show this message and exit.
```

## 6.4. define-ssh

---

```
Usage: nvmesh define-ssh [OPTIONS]
```

```
Define new SSH credentials
```

```
Options:
```

```
--help  Show this message and exit.
```

## 6.5. version

---

```
Usage: nvmesh version [OPTIONS]
```

```
Show nvmesh CLI version
```

```
Options:
```

```
--help  Show this message and exit.
```

## 6.6. client

---

```
Usage: nvmesh client [OPTIONS] COMMAND [ARGS]...
```

```
Client related operations
```

```
Options:
```

```
--help  Show this message and exit.
```

```
Commands:
```

```
attach  Attach volumes to clients.  
count   Get total clients count.  
delete  Delete clients.  
detach  Detach volumes from clients.  
show    Show all clients.
```

### 6.6.1. attach

---

```
Usage: nvmesh client attach [OPTIONS]
```

```
Attach volumes to clients. The default access level is SHARED_READ_WRITE.  
If an access mode is provided then you can only specify one client. The  
preempt flag must be used together with an access level.
```

#### Usage examples:

Attach 2 volumes named 'volume-1' and 'volume-2' to 2 clients named 'client-1' and 'client-2':

```
$ nvmesh client attach -c client-1 -c client-2 -v volume-1 -v volume-2
```

Attach a volume named 'volume-3' in an exclusive read/write mode to a client named 'client-1':

```
$ nvmesh client attach -c client-1 -v volume-3 -a EXCLUSIVE_READ_WRITE
```

Attach a volume named 'volume-3' in a shared read only mode to a client named 'client-1', preempting any existing access mode:

```
$ nvmesh client attach -c client-1 -v volume-3 -a SHARED_READ_ONLY -p
```

#### Options:

<code>-c, --client TEXT</code>	The hostname or address of the client [required]
<code>-v, --volume TEXT</code>	The id of the volume to attach [required]
<code>-a, --access [EXCLUSIVE_READ_WRITE SHARED_READ_ONLY SHARED_READ_WRITE]</code>	Volume access level: EXCLUSIVE_READ_WRITE, SHARED_READ_ONLY, SHARED_READ_WRITE
<code>-p, --preempt</code>	Use preempt for applying access level
<code>-h, --help</code>	Show this message and exit.

## 6.6.2. count

Usage: `nvmesh client count [OPTIONS]`

Get total clients count.

#### Options:

`--help` Show this message and exit.

## 6.6.3. delete

Usage: `nvmesh client delete [OPTIONS]`

Delete clients.

Usage example, delete 2 clients named 'client1' and 'client2':

```
$ nvmesh client delete -n client1 -n client2
```

#### Options:

<code>-n, --name TEXT</code>	The id of the client to delete. [required]
<code>-h, --help</code>	Show this message and exit.

## 6.6.4. detach

Usage: nvmesh client detach [OPTIONS]

Detach volumes from clients. For client that is not a localhost the operation will be performed via SSH.

Usage example, detach 2 volumes named 'volume-1' and 'volume-2' from 2 clients named 'client-1' and 'client-2':

```
$ nvmesh client detach -c client-1 -c client-2 -v volume-1 -v volume-2
```

Options:

```
-c, --client TEXT  The hostname or address of the client  [required]
-v, --volume TEXT  The id of the volume to detach        [required]
-f, --force                Force detach
-h, --help                Show this message and exit.
```

## 6.6.5. show

Usage: nvmesh client show [OPTIONS]

Show clients.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 clients, limit the results to 5 clients and present it in a tabular format:

```
$ nvmesh client show --output-format tabular --skip 20 --limit 5
```

Options:

```
-s, --skip INTEGER           Specifies the number of entities to skip
                              when fetching the result.
-l, --limit INTEGER RANGE    Specifies how many entities to fetch after
                              the skip, the maximum value is: 50.
--output-format [tabular|rows|json]
                              The representation in which the data will be
                              displayed. Options: tabular ,rows, json.
```



```
-n, --name TEXT      Show a specific client by name. NOTE: This
                     option is mutually exclusive with the
                     follwoing options: `skip`, `limit`.
-h, --help           Show this message and exit.
```

## 6.7. cluster

Usage: nvmesh cluster [OPTIONS] COMMAND [ARGS]...

Cluster related operations

Options:

```
-h, --help  Show this message and exit.
```

Commands:

```
show          Show NVMesh cluster.
shut-down     Start shut down process for an NVMesh cluster.
```

### 6.7.1. show

Usage: nvmesh cluster show [OPTIONS]

Show NVMesh cluster.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example, show cluster information in a tabular format:

```
$nvmesh cluster show --output-format tabular
```

Options:

```
--output-format [tabular|rows|json]
```

The representation in which the data will be displayed. Options: tabular , rows, json

```
-h, --help  Show this message and exit.
```

## 6.7.2. shut-down

---

Usage: nvmesh cluster shut-down [OPTIONS]

Start shut down process for an NVMesh cluster.

Usage example:

```
$nvmesh cluster shut-down
```

Options:

-h, --help Show this message and exit.

## 6.8. drive

---

Usage: nvmesh drive [OPTIONS] COMMAND [ARGS]...

Drive related operations

Options:

-h, --help Show this message and exit.

Commands:

add-arbiter	Add an arbitration device.
delete	Delete drives.
evict	Evict specific drives by serial number.
exclude-nvme	Exclude nvme drive from NVMesh pool.
format	Starts the format process for the specified drives.
include-nvme	Include an nvme drive in the NVMesh pool.
remove-arbiter	Remove an arbitration device.
show	Show drives.

### 6.8.1. show

---

Usage: nvmesh drive show [OPTIONS]

Show drives.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 drives, limit the results to 5 drives and present it in a tabular format:

```
$ nvmesh drive show --output-format tabular --skip 20 --limit 5
```

Options:

-s, --skip INTEGER	Specifies the number of entities to skip when fetching the result.
-l, --limit INTEGER RANGE	Specifies how many entities to fetch after the skip, the maximum value is: 50.
--output-format [tabular rows json]	The representation in which the data will be displayed. Options: tabular ,rows, json.
-n, --name TEXT	Show a specific drive by name NOTE: This option is mutually exclusive with the following options: `skip`, `limit`.
-h, --help	Show this message and exit.

## 6.8.2. delete

---

Usage: nvmesh drive delete [OPTIONS]

Delete drives.

Usage example, delete 2 drives named 'serial1' and 'serial2':

```
$ nvmesh drive delete -n serial1 -n serial2
```

Options:

-n, --name TEXT	The serial number of the drive to delete. [required]
-h, --help	Show this message and exit.

## 6.8.3. evict

---

Usage: nvmesh drive evict [OPTIONS]

Evict specific drives by serial number.

Usage example, evict 2 drives named 'serial.1' and 'serial.2':

```
$ nvmesh drive evict -n serial.1 -n serial.2
```

Options:

-n, --name TEXT	The serial number of the drive to evict [required]
-y, --yes	Automatically answer "yes" and skip operational warnings.
-h, --help	Show this message and exit.

## 6.8.4. format

---

Usage: nvmesh drive format [OPTIONS]

Starts the format process for the specified drives.

Usage example, format 2 drives named 'serial.1' and 'serial.2':

```
$ nvmesh drive format -n serial.1 -n serial.2
```

Options:

- n, --name TEXT The serial number of the drive to format [required]
- y, --yes Automatically answer "yes" and skip operational warnings.
- h, --help Show this message and exit.

## 6.8.5. add-arbiter

---

Usage: nvmesh drive add-arbiter [OPTIONS]

Add an arbitration device. For a target that is not localhost, the operation will be performed via SSH.

Usage example, add an arbitration device to 2 targets named 'target-1' and 'target-2':

```
$ nvmesh drive add-arbiter -t target-1 -t target-2
```

Options:

- t, --target TEXT The hostname or address of the target [required]
- h, --help Show this message and exit.

## 6.8.6. remove-arbiter

---

Usage: nvmesh drive remove-arbiter [OPTIONS]

Remove an arbitration device. For a target that is not localhost, the operation will be performed via SSH.

Usage example, remove an arbitration device from 2 targets named 'target-1' and 'target-2':

```
$ nvmesh drive remove-arbiter -t target-1 -t target-2
```

Options:

- t, --target TEXT The hostname or address of the target [required]
- h, --help Show this message and exit.

## 6.8.7. include-nvme

---

Usage: nvmesh drive include-nvme [OPTIONS]

Include an nvme drive in the NVMesh pool. For a target that is not localhost the operation will be performed via SSH.

Usage example, include a nvme drive named 'serial.1' into the NVMesh pool of a target named 'target-1':

```
$ nvmesh drive include-nvme -t target-1 -s serial.1
```

Options:

-t, --target TEXT	The hostname or address of the target	[required]
-s, --serial TEXT	The serial number of the drive	[required]
-h, --help	Show this message and exit.	

## 6.8.8. exclude-nvme

---

Usage: nvmesh drive exclude-nvme [OPTIONS]

Exclude nvme drive from NVMesh pool. For a target that is not localhost, the operation will be performed via SSH.

Usage example, exclude a nvme drive named 'serial.1' from the NVMesh pool of a target named 'target-1':

```
$ nvmesh drive exclude-nvme -t target-1 -s serial.1
```

Options:

-t, --target TEXT	The hostname or address of the target	[required]
-s, --serial TEXT	The serial number of the drive	[required]
-h, --help	Show this message and exit.	

## 6.9. driveclass

---

Usage: nvmesh driveclass [OPTIONS] COMMAND [ARGS]...

Drive Class related operations

Options:

--help	Show this message and exit.
--------	-----------------------------

Commands:

create	Create a drive class.
delete	Delete drive classes.
show	Show all drive classes.
update	Update a drive class.

## 6.9.1. create

Usage: nvmesh driveclass create [OPTIONS]

Create a drive class.

Usage example, create a drive class named 'dc1' from 2 drives named 'samsungDriveSerial1' and 'intelDriveSerial12', the drive class will consider a domain with a scope named 'Rack' and an identifier named 'A':

```
$ nvmesh driveclass create --name dc1 --drive samsungDriveSerial1 --drive intelDriveSerial12 --domain Rack:A
```

Options:

-n, --name TEXT	The name of the drive class [required]
-dr, --drive TEXT	The drive's serial number. NOTE: This option can be used multiple times, e.g. -dr <value> -dr <value>. It can also be used once with multiple values separated with the ',' character, e.g. -dr <value>,<value>,<value> [required]
-d, --description TEXT	The description of the drive class
--domain TEXT	Domain in the following format: <scope:identifier>
-h, --help	Show this message and exit.

## 6.9.2. delete

Usage: nvmesh driveclass delete [OPTIONS]

Delete drive classes.

Usage example, delete 2 drive classes named 'dc1' and 'dc2':

```
$ nvmesh driveclass delete -n dc1 -n dc2
```

Options:

-n, --name TEXT	The id of the drive class to delete. NOTE: This option can be used multiple times, e.g. -n <value> -n <value>. It can also be used once with multiple values separated with the ',' character, e.g. -n <value>,<value>,<value> [required]
-h, --help	Show this message and exit.

## 6.9.3. show

Usage: nvmesh driveclass show [OPTIONS]

Show drive classes.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 drive classes, limit the results to 5 drive classes and present it in a tabular format:

```
$ nvmesh driveclass show --output-format tabular --skip 20 --limit 5
```

Options:

-s, --skip INTEGER	Specifies the number of entities to skip when fetching the result.
-l, --limit INTEGER RANGE	Specifies how many entities to fetch after the skip, the maximum value is: 50.
--output-format [tabular rows json]	The representation in which the data will be displayed. Options: tabular , rows, json.
-n, --name TEXT	Show a specific drive class by name. NOTE: This option is mutually exclusive with the following options: `skip`, `limit`.
-h, --help	Show this message and exit.

## 6.9.4. update

Usage: nvmesh driveclass update [OPTIONS]

Update a drive class.

Usage example, update a drive class named 'dc1' to include 2 drives named 'samsungDriveSerial1' and 'intelDriveSerial12', and consider a domain with a scope named 'Rack' and an identifier named 'A':

```
$ nvmesh driveclass update --name dc1 --drive samsungDriveSerial1 --drive intelDriveSerial12 --domain Rack:A
```

Options:

-n, --name TEXT	The name of the drive class [required]
-dr, --drive TEXT	The drive's serial number. NOTE: This option can be used multiple times, e.g. -dr <value> -dr <value>. It can also be used once with multiple values separated with the ',' character, e.g. -dr <value>,<value>,<value> [required]
-d, --description TEXT	The description of the drive class
--domain TEXT	Domain in the following format: <scope:identifier>

```
-h, --help          Show this message and exit.
```

## 6.10. target

---

```
Usage: nvmesh target [OPTIONS] COMMAND [ARGS]...
```

```
Target related operations
```

```
Options:
```

```
--help  Show this message and exit.
```

```
Commands:
```

```
count          Get total targets count.
delete         Delete targets.
delete-nic     Delete specific NIC.
show           Show all targets.
```

### 6.10.1. count

---

```
Usage: nvmesh target count [OPTIONS]
```

```
Get total targets count.
```

```
Options:
```

```
--help  Show this message and exit.
```

### 6.10.2. delete

---

```
Usage: nvmesh target delete [OPTIONS]
```

```
Delete targets.
```

```
Usage example, delete 2 targets named 'server1' and 'server2':
```

```
$ nvmesh target delete -n server1 -n server2
```

```
Options:
```

```
-n, --name TEXT  The id of the server to delete [required]
-h, --help      Show this message and exit.
```

### 6.10.3. delete-nic

---

```
Usage: nvmesh target delete-nic [OPTIONS]
```





## 6.11. targetclass

Usage: nvmesh targetclass [OPTIONS] COMMAND [ARGS]...

Target Class related operations

Options:

--help Show this message and exit.

Commands:

create Create a target class.  
 delete Delete target classes.  
 show Show all target classes.  
 update Update a target class.

### 6.11.1. create

Usage: nvmesh targetclass create [OPTIONS]

Create a target class.

Usage example, create a target class named 'tcl' from 2 targets named 'server-1' and 'server-2', the target class will consider a domain with a scope named 'Rack' and an identifier named 'A':

```
$ nvmesh targetclass create --name tcl --target server-1 --target server-2
--domain Rack:A
```

Options:

-n, --name TEXT	The name of the target class [required]
-t, --target TEXT	The name of the target to group under the target class. NOTE: This option can be used multiple times, e.g. -t <value> -t <value>. It can also be used once with multiple values separated with the ',' character, e.g. -t <value>,<value>,<value> [required]
-d, --description TEXT	The description of the target class
--domain TEXT	Domain in the following format: <scope:identifier>
-h, --help	Show this message and exit.

### 6.11.2. delete

Usage: nvmesh targetclass delete [OPTIONS]

Delete target classes.

Usage example, delete 2 target classes named 'tcl1' and 'tcl2':

```
$ nvmesh targetclass delete -n tc1 -n tc2
```

Options:

```
-n, --name TEXT    The id of the drive class to delete. NOTE: This option can
                   be used multiple times, e.g. -n <value> -n <value>. It can
                   also be used once with multiple values separated with the
                   ',' character, e.g. -n <value>,<value>,<value> [required]

-h, --help        Show this message and exit.
```

## 6.11.3. show

Usage: nvmesh targetclass show [OPTIONS]

Show target classes.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 target classes, limit the results to 5 target classes and present it in a tabular format:

```
$ nvmesh targetclass show --output-format tabular --skip 20 --limit 5
```

Options:

```
-s, --skip INTEGER          Specifies the number of entities to skip
                             when fetching the result.
-l, --limit INTEGER RANGE   Specifies how many entities to fetch after
                             the skip, the maximum value is: 50.
--output-format [tabular|rows|json]
                             The representation in which the data will be
                             displayed. Options: tabular ,rows, json.
-n, --name TEXT             Show a specific target class by name. NOTE:
                             This option is mutually exclusive with the
                             following options: `skip`, `limit`.
-h, --help                  Show this message and exit.
```

## 6.11.4. update

Usage: nvmesh targetclass update [OPTIONS]

Update a target class.

Usage example, update a target class named 'tcl' with 2 targets named 'server-1' and 'server-2', the target class will consider a domain with a scope named 'Rack' and an identifier named 'A':

```
$ nvmesh targetclass update --name tcl --target server-1 --target server-2
--domain Rack:A
```

Options:

-n, --name TEXT	The name of the target class [required]
-t, --target TEXT	The name of the target to group under the target class. NOTE: This option can be used multiple times, e.g. -t <value> -t <value>. It can also be used once with multiple values separated with the ',' character, e.g. -t <value>,<value>,<value> [required]
-d, --description TEXT	The description of the target class
--domain TEXT	Domain in the following format: <scope:identifier>
-h, --help	Show this message and exit.

## 6.12. volume

Usage: nvmesh volume [OPTIONS] COMMAND [ARGS]...

Volume related operations

Options:

-h, --help Show this message and exit.

Commands:

create	Create volume.
delete	Delete volumes.
rebuild	Rebuild volumes.
show	Show volumes.
update	Update volume.

### 6.12.1. create

Usage: nvmesh volume create [OPTIONS]

Create volume.

**IMPORTANT:** When creating a volume only one of the following can be defined: VPG, target/drive classes, limit by targets/drives.

Usage example, create a volume named 'r1\_vol' whose type is Mirrored RAID-1 and has a capacity of 1TB:

```
$ nvmesh volume create -n r1_vol --raid-level 1 -c 1T
```

## Options:

<code>-n, --name TEXT</code>	The name of the volume. The name must be unique, as it will become the ID of the volume. [required]
<code>-rl, --raid-level [lvm ec 0 1 10]</code>	The RAID level of the volume. Options: lvm = Concatenated, ec = Erasure Coding, 0 = Striped RAID-0, 1 = Mirrored RAID-1, 10 = Striped & Mirrored RAID-10. NOTE: This option is mutually exclusive with the following options: `vpg`. [required]
<code>-c, --capacity TEXT</code>	The capacity of the allocated volume in bytes. Minimal volume capacity is 1GB. The number of bytes may be followed by the following multiplicative suffixes: K/KB =1000, KiB =1024, M/MB =1000^2, MiB =1024^2, G/GB =1000^3, GiB =1024^3, T/TB =1000^4, TiB =1024^4, , PiB =1024^5. Use "MAX" for using all of the available space. All the units are case insensitive. [required]
<code>-d, --description TEXT</code>	Description of the volume.
<code>--domain TEXT</code>	Domain to use.
<code>--vpg TEXT</code>	The VPG to use.
<code>--relative-rebuild-priority INTEGER RANGE</code>	Sets the volume relative rebuild priority.
<code>-dc, --drive-class TEXT</code>	Limit volume allocation to specific drive classes. NOTE: This option can be used multiple times, e.g. <code>-dc &lt;value&gt; -dc &lt;value&gt;</code> . It can also be used once with multiple values separated with the ',' character, e.g. <code>-dc &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code>
<code>-tc, --target-class TEXT</code>	Limit volume allocation to specific target classes. NOTE: This option can be used multiple times, e.g. <code>-tc &lt;value&gt; -tc &lt;value&gt;</code> . It can also be used once with multiple values separated with the ',' character, e.g. <code>-tc &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code>
<code>-D, --limit-by-drive TEXT</code>	Limit volume allocation to specific drives. NOTE: This option can be used multiple times, e.g. <code>-D &lt;value&gt; -D &lt;value&gt;</code> . It can also be used once with multiple values separated with the ',' character, e.g. <code>-D &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code>
<code>-T, --limit-by-target TEXT</code>	Limit volume allocation to specific targets. NOTE: This option can be used multiple times, e.g. <code>-T &lt;value&gt; -T &lt;value&gt;</code> . It can also be used once with multiple values separated with the ',' character, e.g. <code>-T &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code>
<code>--stripe-width INTEGER</code>	Number of disks to use. Required if RAID Level is 0 or 10.
<code>--wait-for-status [online offline degraded]</code>	

```

--timeout INTEGER          Wait for the volume to become in a specific
                           status before continue. Options: online,
                           offline, degraded.
--data-blocks INTEGER RANGE Maximum time in seconds to wait for status.
                           (default: 60 seconds)
--parity-blocks INTEGER RANGE Number of disks to use. Required if RAID
                              Level is ec.
--protection-level, --target-node-redundancy [full|minimal|ignore]
                              Protection level to use. Required if RAID
                              Level is ec. Options: full = Full Separation
                              (N+2 Target redundancy), minimal = Minimal
                              Separation (N+1 Target Redundancy), ignore =
                              Ignore Separation (No Target Redundancy).
                              Optional if the RAID Level is 1 or 10.
                              Options: full = Full Separation (1+1 Target
                              Node Separation) - default, ignore = Ignore
                              Separation (No Target Redundancy).

--enable-nvmf BOOLEAN      Enables access to the NVMesh volume using
                           the NVMf protocol.
-nc, --nvmf-client TEXT    The ID of a client that allowed to export
                           via NVMf protocol. NOTE: This option can be
                           used multiple times, e.g. -nc <value> -nc
                           <value>. It can also be used once with
                           multiple values separated with the ','
                           character, e.g. -nc <value>,<value>,<value>

--enable-crc-check BOOLEAN Enables CRC check of the volume.
-vsg, --volume-security-group TEXT
                           The name of the associated volume security
                           group. NOTE: This option can be used
                           multiple times, e.g. -vsg <value> -vsg
                           <value>. It can also be used once with
                           multiple values separated with the ','
                           character, e.g. -vsg <value>,<value>,<value>

-h, --help                Show this message and exit.

```

## 6.12.2. delete

Usage: nvmesh volume delete [OPTIONS]

Delete volumes.

Usage example, delete 2 volumes named 'vol1' and 'vol2':

```
$ nvmesh volume delete -n vol1 -n vol2
```

Options:

```

-n, --name TEXT    The name of the volume to delete. NOTE: This option can
                   be used multiple times, e.g. -n <value> -n <value>. It
                   can also be used once with multiple values separated
                   with the ',' character, e.g. -n <value>,<value>,<value>

```

	[required]
-y, --yes	Automatically answer "yes" and skip operational warnings.
--wait-for-deletion	Wait for the volume to be deleted.
--timeout INTEGER	Maximum time in seconds to wait for deletion. (default: 60 seconds)
-h, --help	Show this message and exit.

## 6.12.3. rebuild

Usage: nvmesh volume rebuild [OPTIONS]

Rebuild volumes.

Usage example, rebuild 2 volumes named 'v1' and 'v2':

```
$ nvmesh volume rebuild -n v1 -n v2
```

Options:

-n, --name TEXT	The name of the volume to rebuild. NOTE: This option can be used multiple times, e.g. -n <value> -n <value>. It can also be used once with multiple values separated with the ',' character, e.g. -n <value>,<value>,<value> [required]
-h, --help	Show this message and exit.

## 6.12.4. show

Usage: nvmesh volume show [OPTIONS]

Show volumes.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 volumes, limit the results to 5 volumes and present it in a tabular format:

```
$ nvmesh volume show --output-format tabular --skip 20 --limit 5
```

## Options:

<code>-s, --skip INTEGER</code>	Specifies the number of entities to skip when fetching the result.
<code>-l, --limit INTEGER RANGE</code>	Specifies how many entities to fetch after the skip, the maximum value is: 50.
<code>--output-format [tabular rows json]</code>	The representation in which the data will be displayed. Options: <code>tabular</code> , <code>rows</code> , <code>json</code> . NOTE: This option is mutually exclusive with the following options: <code>'layout'</code> .
<code>-n, --name TEXT</code>	Show a specific volume by name. NOTE: This option is mutually exclusive with the following options: <code>'skip'</code> , <code>'limit'</code> .
<code>--layout</code>	Show the layout of the volumes in a tabular representation, the rest of the data will be displayed in rows format.
<code>-h, --help</code>	Show this message and exit.

## 6.12.5. update

Usage: `nvmesh volume update [OPTIONS]`

Update volume.

Usage example, update a volume named 'r1\_vol' and set its capacity to 2TB:

```
$ nvmesh volume update -n r1_vol -c 2T
```

## Options:

<code>-n, --name TEXT</code>	The name of the volume. The name must be unique, as it will become the ID of the volume. [required]
<code>-c, --capacity TEXT</code>	The capacity of the allocated volume in bytes. Minimal volume capacity is 1GB. The number of bytes may be followed by the following multiplicative suffixes: K/KB =1000, KiB =1024, M/MB =1000^2, MiB =1024^2, G/GB =1000^3, GiB =1024^3, T/TB =1000^4, TiB =1024^4, , PiB =1024^5. Use "MAX" for using all of the available space. All the units are case insensitive.
<code>-d, --description TEXT</code>	Description of the volume.
<code>--domain TEXT</code>	Domain to use.
<code>-dc, --drive-class TEXT</code>	Limit volume allocation to specific drive classes. NOTE: This option can be used multiple times, e.g. <code>-dc &lt;value&gt; -dc &lt;value&gt;</code> . It can also be used once with multiple values separated with the ',' character, e.g. <code>-dc &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code>
<code>-tc, --target-class TEXT</code>	Limit volume allocation to specific target classes. NOTE: This option can be used multiple times, e.g. <code>-tc &lt;value&gt; -tc</code>



<p><code>-D, --limit-by-drive TEXT</code></p> <p><code>-T, --limit-by-target TEXT</code></p> <p><code>--relative-rebuild-priority INTEGER RANGE</code></p> <p><code>--enable-nvmf BOOLEAN</code></p> <p><code>-nc, --nvmf-client TEXT</code></p> <p><code>--enable-crc-check BOOLEAN</code></p> <p><code>-vsg, --volume-security-group TEXT</code></p> <p><code>-h, --help</code></p>	<p>&lt;value&gt;. It can also be used once with multiple values separated with the ',' character, e.g. <code>-tc &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code></p> <p>Limit volume allocation to specific drives. NOTE: This option can be used multiple times, e.g. <code>-D &lt;value&gt; -D &lt;value&gt;</code>. It can also be used once with multiple values separated with the ',' character, e.g. <code>-D &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code></p> <p>Limit volume allocation to specific targets. NOTE: This option can be used multiple times, e.g. <code>-T &lt;value&gt; -T &lt;value&gt;</code>. It can also be used once with multiple values separated with the ',' character, e.g. <code>-T &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code></p> <p>Sets the volume relative rebuild priority.</p> <p>Enables access to the NVMesh volume using the NVMf protocol.</p> <p>The ID of a client that allowed to export via NVMf protocol. NOTE: This option can be used multiple times, e.g. <code>-nc &lt;value&gt; -nc &lt;value&gt;</code>. It can also be used once with multiple values separated with the ',' character, e.g. <code>-nc &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code></p> <p>Enables CRC check of the volume. Can only be updated on an EC volume.</p> <p>The name of the associated volume security group. NOTE: This option can be used multiple times, e.g. <code>-vsg &lt;value&gt; -vsg &lt;value&gt;</code>. It can also be used once with multiple values separated with the ',' character, e.g. <code>-vsg &lt;value&gt;,&lt;value&gt;,&lt;value&gt;</code></p> <p>Show this message and exit.</p>
---	--

## 6.13. vpg

Usage: `nvmesh vpg [OPTIONS] COMMAND [ARGS]...`

VPG related operations

Options:

`--help` Show this message and exit.

Commands:

`create` Create VPGs.  
`delete` Delete VPGs.  
`show` Show all VPGs.

## 6.13.1. create

Usage: nvmesh vpg create [OPTIONS]

Create VPG.

Usage example, create a VPG named 'r1\_vpg' whose type is Mirrored RAID-1 and has a capacity of 1TB:

```
$ nvmesh vpg create -n r1_vpg --raid-level 1 -c 1T
```

Options:

-n, --name TEXT	The name of the volume. The name must be unique, as it will become the ID of the volume. [required]
-rl, --raid-level [lvm ec 0 1 10]	The RAID level of the volume. Options: lvm = Concatenated, ec = Erasure Coding, 0 = Striped RAID-0, 1 = Mirrored RAID-1, 10 = Striped & Mirrored RAID-10. [required]
-c, --capacity TEXT	The capacity of the allocated volume in bytes. Minimal volume capacity is 1GB. The number of bytes may be followed by the following multiplicative suffixes: K/KB =1000, KiB =1024, M/MB =1000^2, MiB =1024^2, G/GB =1000^3, GiB =1024^3, T/TB =1000^4, TiB =1024^4, , PiB =1024^5. All the units are case insensitive.
-d, --description TEXT	Description of the volume.
--domain TEXT	Domain to use.
-dc, --drive-class TEXT	Limit volume allocation to specific drive classes.
-tc, --target-class TEXT	Limit volume allocation to specific target classes.
--stripe-width INTEGER	Number of disks to use. Required if RAID Level is 0 or 10.
--data-blocks INTEGER RANGE	Number of disks to use. Required if RAID Level is ec.
--parity-blocks INTEGER RANGE	Number of disks to use. Required if RAID Level is ec.
--protection-level, --target-node-redundancy [full minimal ignore]	Protection level to use. Required if RAID Level is ec. Options: full = Full Separation (N+2 Target redundancy), minimal = Minimal Separation (N+1 Target Redundancy), ignore = Ignore Separation (No Target Redundancy). Optional if the RAID Level is 1 or 10. Options: full = Full Separation (1+1 Target Node Separation) - default, ignore = Ignore Separation (No Target Redundancy).
--enable-crc-check BOOLEAN	Enables CRC check of the volume.
-vsg, --volume-security-group TEXT	The name of the associated volume security

-h, --help

group. NOTE: This option can be used multiple times, e.g. -vsg <value> -vsg <value>. It can also be used once with multiple values separated with the ',' character, e.g. -vsg <value>,<value>,<value> Show this message and exit.

## 6.13.2. delete

Usage: nvmesh vpg delete [OPTIONS]

Delete VPGs.

Usage example, delete 2 VPGs named 'vpg1' and 'vpg2':

```
$ nvmesh vpg delete -n vpg1 -n vpg2
```

Options:

-n, --name TEXT The name of the VPG to delete. NOTE: This option can be used multiple times, e.g. -n <value> -n <value>. It can also be used once with multiple values separated with the ',' character, e.g. -n <value>,<value>,<value> [required]

-h, --help Show this message and exit.

## 6.13.3. show

Usage: nvmesh vpg show [OPTIONS]

Show VPGs.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 VPGs, limit the results to 5 VPGs and present it in a tabular format:

```
$ nvmesh vpg show --output-format tabular --skip 20 --limit 5
```

Options:

<code>-s, --skip INTEGER</code>	Specifies the number of entities to skip when fetching the result.
<code>-l, --limit INTEGER RANGE</code>	Specifies how many entities to fetch after the skip, the maximum value is: 50.
<code>--output-format [tabular rows json]</code>	The representation in which the data will be displayed. Options: <code>tabular</code> , <code>rows</code> , <code>json</code> .
<code>-n, --name TEXT</code>	Show a specific VPG by name. NOTE: This option is mutually exclusive with the following options: <code>`skip`</code> , <code>`limit`</code> .
<code>-h, --help</code>	Show this message and exit.

## 6.14. mongo-db

Usage: `nvmesh mongo-db [OPTIONS] COMMAND [ARGS]...`

MongoDB related operations

Options:

`-h, --help` Show this message and exit.

Commands:

`show` Show Mongo DB status.

### 6.14.1. show

Usage: `nvmesh mongo-db show [OPTIONS]`

Show Mongo DB status.

`--output-format` options:

`tabular` - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

`rows` - Render tabular data with one column per line (allowing columns with line breaks).

`json` - Format output as DB JSON.

Usage example, show MongoDB information in a tabular format:

```
$nvmesh mongo-db show --output-format tabular
```

Options:

`--output-format [tabular|rows|json]`

The representation in which the data will be displayed. Options: `tabular`, `rows`, `json`

`-h, --help`

Show this message and exit.

## 6.15. volume-security-group

Usage: `nvmesh volume-security-group [OPTIONS] COMMAND [ARGS]...`

Volume Security Group related operations

Options:

`-h, --help` Show this message and exit.

Commands:

`create` Create a VSG.

`delete` Delete VSGs.

`show` Show VSGs.

`update` Update a VSG.

### 6.15.1. create

Usage: `nvmesh volume-security-group create [OPTIONS]`

Create a VSG.

Usage example, create a VSG named 'vsg1':

```
$ nvmesh volume-security-group create --name vsg1 --key someKey --key
someOtherKey
```

Options:

`-n, --name TEXT` The name of the VSG. [required]

`-d, --description TEXT` The description of the VSG.

`-k, --key TEXT` The name of the key. NOTE: This option can be used multiple times, e.g. `-d <value> -d <value>`. It can also be used once with multiple values separated with the ',' character, e.g. `-d <value>,<value>,<value>`

### 6.15.2. show

Usage: `nvmesh volume-security-group show [OPTIONS]`

Show VSGs.

`--output-format options:`

`tabular` - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the

terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 VSGs, limit the results to 5 VSGs and present it in a tabular format:

```
$ nvmesh volume-security-group show --output-format tabular --skip 20
--limit 5
```

Options:

-s, --skip INTEGER	Specifies the number of entities to skip when fetching the result.
-l, --limit INTEGER RANGE	Specifies how many entities to fetch after the skip, the maximum value is: 50.
--output-format [tabular rows json]	The representation in which the data will be displayed. Options: tabular, rows, json.
-n, --name TEXT	Show a specific VSG by name. NOTE: This option is mutually exclusive with the following options: `skip`, `limit`.
-h, --help	Show this message and exit.

## 6.15.3. update

Usage: nvmesh volume-security-group update [OPTIONS]

Update a VSG. The name field cannot be updated.

Usage example, update a VSG named 'vsg1' with a description:

```
$ nvmesh volume-security-group update --name vsg1 --description 'very
important VSG'
```

Options:

-n, --name TEXT	The name of the VSG. [required]
-d, --description TEXT	The description of the VSG.
-k, --key TEXT	The name of the key. NOTE: This option can be used multiple times, e.g. -d <value> -d <value>. It can also be used once with multiple values separated with the ',' character, e.g. -d <value>,<value>,<value>
-h, --help	Show this message and exit.

## 6.15.4. delete

Usage: nvmesh volume-security-group delete [OPTIONS]

Delete VSGs.

Usage example, delete 2 VSGs named 'vsg1' and 'vsg2':

```
$ nvmesh volume-security-group delete -n vsg1 -n vsg2
```

Options:

-n, --name TEXT	The name of the VSG to delete. NOTE: This option can be used multiple times, e.g. -n <value> -n <value>. It can also be used once with multiple values separated with the ',' character, e.g. -n <value>,<value>,<value> [required]
-h, --help	Show this message and exit.

## 6.16. key-pair

Usage: nvmesh key-pair [OPTIONS] COMMAND [ARGS]...

Key Pair related operations

Options: -h, --help Show this message and exit.

Commands: create Create a key. delete Delete keys. download Download keys. show Show keys. update Update a key.

### 6.16.1. download

Usage: nvmesh key-pair download [OPTIONS]

Download keys.

Usage example, download a key named 'key1':

```
$ nvmesh key-pair download --name key1 --destination '/etc/nvmesh/keys'
```

Options:

-n, --name TEXT	The id of the key to download. NOTE: This option can be used multiple times, e.g. -n <value> -n <value>. It can also be used once with multiple values separated with the ',' character, e.g. -n <value>,<value>,<value> [required]
-d, --destination TEXT	The destination on the file system to where the key will be downloaded. [required]
-h, --help	Show this message and exit.

## 6.16.2. create

Usage: nvmesh key-pair create [OPTIONS]

Create a key.

Usage example, create a key named 'key1':

```
$ nvmesh key-pair create --name key1
```

Options:

-n, --name TEXT	The name of the key. [required]
-d, --description TEXT	The description of the key.
-h, --help	Show this message and exit.

## 6.16.3. show

Usage: nvmesh key-pair show [OPTIONS]

Show keys.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 keys, limit the results to 5 keys and present it in a tabular format:

```
$ nvmesh key-pair show --output-format tabular --skip 20 --limit 5
```

Options:

-s, --skip INTEGER	Specifies the number of entities to skip when fetching the result.
-l, --limit INTEGER RANGE	Specifies how many entities to fetch after the skip, the maximum value is: 50.
--output-format [tabular rows json]	The representation in which the data will be displayed. Options: tabular, rows, json.
-n, --name TEXT	Show a specific key by name. NOTE: This option is mutually exclusive with the following options: `skip`, `limit`.



-h, --help

Show this message and exit

## 6.16.4. update

Usage: nvmesh key-pair update [OPTIONS]

Update a key. The name field cannot be updated.

Usage example, update a key named 'key1' with a description:

```
$ nvmesh key-pair update --name key1 --description 'very important key'
```

Options:

-n, --name TEXT	The name of the key. [required]
-d, --description TEXT	The description of the key. [required]
-h, --help	Show this message and exit.

## 6.16.5. delete

Usage: nvmesh key-pair delete [OPTIONS]

Delete keys.

Usage example, delete 2 keys named 'key1' and 'key2':

```
$ nvmesh key-pair delete -n key1 -n key2
```

Options:

-n, --name TEXT	The id of the key to delete. NOTE: This option can be used multiple times, e.g. -n <value> -n <value>. It can also be used once with multiple values separated with the ',' character, e.g. -n <value>,<value>,<value> [required]
-h, --help	Show this message and exit.

## 6.17. log

Usage: nvmesh log [OPTIONS] COMMAND [ARGS]...

Log related operations

Options:

-h, --help Show this message and exit.

Commands:

acknowledge	Acknowledge the specified logs by their ID.
acknowledge-all-alerts	Acknowledge all logged alerts.
show	Show logs.

## 6.17.1. acknowledge-all-alerts

---

Usage: nvmesh log acknowledge-all-alerts [OPTIONS]

Acknowledge all logged alerts.

Options:

-h, --help Show this message and exit.

## 6.17.2. acknowledge

---

Usage: nvmesh log acknowledge [OPTIONS]

Acknowledge the specified logs by their ID.

Usage example, acknowledge 2 logs with the IDs: 5efc7901e963fb0fb7181f0d and 5efcabd2938d006ed224a0ac:

```
$ nvmesh logs acknowledge --id 5efc7901e963fb0fb7181f0d --id 5efcabd2938d006ed224a0ac
```

Options:

--id TEXT The ID of the log to acknowledge [required]  
-h, --help Show this message and exit.

## 6.17.3. show

---

Usage: nvmesh log show [OPTIONS]

Show logs.

--output-format options:

tabular - Render a table using characters like dashes and vertical bars to emulate borders, may overflow and wrap the output if the lines exceed the terminal width.

rows - Render tabular data with one column per line (allowing columns with line breaks).

json - Format output as DB JSON.

Usage example:

Skip the first 20 logs, limit the results to 5 logs and present it in a tabular format:

```
$ nvmesh log show --output-format tabular --skip 20 --limit 5
```

## Options:

<code>-s, --skip INTEGER</code>	Specifies the number of entities to skip when fetching the result.
<code>-l, --limit INTEGER RANGE</code>	Specifies how many entities to fetch after the skip, the maximum value is: 50.
<code>--output-format [tabular rows json]</code>	The representation in which the data will be displayed. Options: <code>tabular</code> , <code>rows</code> , <code>json</code> .
<code>--alerts</code>	Show alerts (errors that hasn't been acknowledged).
<code>-h, --help</code>	Show this message and exit.

# Command Line Utilities

---

The following section describes various command-line utilities provided in NVMesh.

## **`nvmesh_clnt_analyzer`**

---

Analyze NVMesh volumes.

```
usage: nvmesh_clnt_analyzer [-h] [-v <vol1> <vol2> ...] [-d <debug_level>]

positional arguments:
  volume                a volume name to analyze
  debug_level           the debug level of the output to trace/debug/info/notice/
warning/error
  output_file           the file name to use for an output

optional arguments:
  -h, --help            show this help message and exit
  -v <volume> [<volumeX> ...], --volumes volume [<volumeX> ...]
                        Volume list: -v vol1 vol2
                        if not used all volumes will be inspected
  -d <debug_level>, --debug_level <debug_level>
                        Set the debug level of the output to trace/debug/info/not
ice/warning/error
  -o <output_file>, --output_file <output_file>
                        Where to output the script

examples:

To analyze all volumes connected to this client:

nvmesh_clnt_analyzer
```

## **`nvmesh_configure_management_server`**

---

Set the management server for *client* or *target nodes*.

```
Usage: nvmesh_configure_management_server [--addresses <nvme42:4001,nvme43:4001>
--protocol <https/http>]

Example:
nvmesh_configure_management_server --addresses server82:4001 --protocol https
```

## **`nvmesh_configure_nics`**

---

Define which Network Interface Cards (NICs) should be used with NVMesh *client* or *target nodes*.

`nvmesh_configure_nics` is an interactive script.

## **`nvmesh_health_check`**

---

Display and validate the NVMesh configuration.

## **`nvmesh_logs_collector`**

---

## **`nvmesh_set_io1_interrupts`**

---

Distributes interrupts from drives across CPUs.

```
usage: nvmesh_set_io1_interrupts
```

## **`nvmesh_update`**

---

Updates NVMesh kernel module.

# 7. Document Reference

## Typographical Conventions

Throughout this document, the following typographical conventions are followed:

Style	Meaning
<b>bold text</b>	The name of an Excelero software component or technology
text	A file name, command or configuration text that can be utilized in a Linux terminal/shell, file or as a URL
<i>term in italics</i>	Generally, a term being used in specific relation to an element in the NVMesh

## Definitions

Throughout this document, these terms have the following meanings:

Term	Definition
<i>Management Server</i>	The server(s), or OS image(s) running the <b>management module</b> software
<i>Target Node/ Target</i>	A physical server containing one or more NVMe SSDs running the <b>storage target module</b>
<i>Client Node/ Client</i>	An OS image instance running the <b>block storage client</b> software
<i>Converged Node</i>	A <i>target node</i> that is also running the <b>block storage client</b> software
<i>Logical Volume/ Volume</i>	A logical block device defined with the <b>NVMesh management module</b> that can be attached to <i>client nodes</i>
<b>RDDA</b>	Remote Direct Drive Access. Excelero's patented low-latency and CPU bypass transport technology.
<b>TOMA</b>	<b>Topology Manager</b> . The <b>storage target module</b> component that handles error detection and volume rebuild activities.