NVMesh CLI Guide

1.3 — Last update: 2019/07/26

Excelero, Inc.

Table of Contents

1. Copyright and Trademark Information

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.

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

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

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

4.1. Supported Environments

Linux and MacOS running Python version 2.

> ! Python minimum requirement is 2.7.5. Python v3 is not supported.

4.2. Installation Requirements

You need a working pip environment before attempting to install the tool.
More information and how to install pip can be found here: Installing pip.

4.3. 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/NVM`

`esh/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 var
ious 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 under the user's home directory, see the nvmesh CLI Files section for more details.

5.2. nvmesh CLI Files

There are other files stored in the users home directory in addition to the credentials.

`/etc/opt/NVMesh/nvmesh.conf`	Stores the NVMesh management server name
`~/.nvmesh_api_secrets`	Stores the API username and password
`~/.nvmesh_cli_history`	Stores the NVMesh shell cli tool command history.
`~/.nvmesh_ssh_secrets`	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

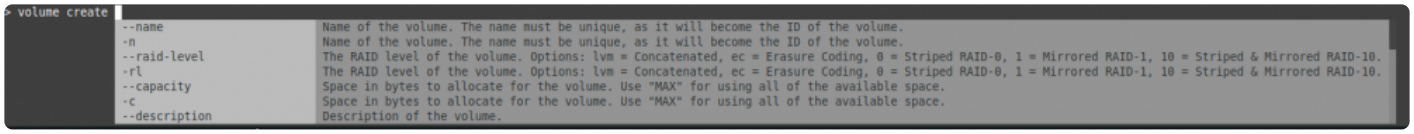
To use the interactive mode just type: `nvmesh` (with no additional arguments)

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:



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-classe
s 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]...

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
  volume       Group Volumes related operations
  vpg          Group VPGs related operations
```

All of the commands (except for `logout` and `define-ssh`) 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 100000000000 --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]
  -c, --capacity INTEGER          Space in bytes to allocate for the volume.
  -d, --description TEXT          Description of the volume.
  --domain TEXT                   Domain to use.
  -dc, --drive-classes TEXT       Limit volume allocation to specific drive
                                  classes.
  -tc, --target-classes 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 [full|minimal|ignore]
                                  Protection level to use. Required if RAID
                                  Level is ec. Options: full = Full
                                  Separation, minimal = Minimal Separation,
                                  ignore = Ignore Separation.
  --help                          Show this message and exit.
```

6. Command Reference

First level commands:

6.1. logout

```
Usage: nvmesh logout [OPTIONS]

  Logout the current user from the CLI

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

6.2. define-ssh

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

  Define new SSH credentials

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

6.3. client

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

```
  Group Clients 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.
```

6.3.1. attach

```
Usage: nvmesh client attach [OPTIONS]

  Attach volumes to clients.

  usage example: -c client-1 -c client-2 -v volume-1 -v volume-2

Options:
  -c, --clients TEXT  The id of the client  [required]
  -v, --volumes TEXT  The id of the volume to attach  [required]
  --help              Show this message and exit.
```

6.3.2. count

```
Usage: nvmesh client count [OPTIONS]

  Get total clients count.

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

6.3.3. delete

```
Usage: nvmesh client delete [OPTIONS]

  Delete clients. usage example: -n client-1 -n client-2
```

```
Options:
  -n, --names TEXT  The id of the client to delete.  [required]
  --help            Show this message and exit.
```

6.3.4. detach

```
Usage: nvmesh client detach [OPTIONS]

  Detach volumes from clients.

  usage example: -c client-1 -c client-2 -v volume-1 -v volume-2

Options:
  -c, --clients TEXT  The id of the client  [required]
  -v, --volumes TEXT  The id of the volume to detach  [required]
  --help              Show this message and exit.
```

6.3.5. show

```
Usage: nvmesh client show [OPTIONS]

  Show all 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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

6.4. cluster

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

  Group Cluster related operations

Options:
  --help  Show this message and exit.

Commands:
  show  Show NVMesh cluster.
```

6.4.1. 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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

6.5. drive

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

  Group Drives related operations
```

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

Commands:
  delete  Delete specific drives by serial number.
  evict   Evict specific drives by serial number.
  format  Starts the format process for the specified drives.
```

6.5.1. delete

```
Usage: nvmesh drive delete [OPTIONS]

  Delete specific drives by serial number.

  usage example: -n drive-1 -n drive-2

Options:
  -n, --names TEXT  The serial number of the drive to delete  [required]
  --help            Show this message and exit.
```

6.5.2. evict

```
Usage: nvmesh drive evict [OPTIONS]

  Evict specific drives by serial number.

  usage example: -n drive-1 -n drive-2

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

6.5.3. format

```
Usage: nvmesh drive format [OPTIONS]

  Starts the format process for the specified drives.

  usage example: -n drive-1 -n drive-2
```

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

6.6. driveclass

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

  Group Drive Classes 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.
```

6.6.1. create

```
Usage: nvmesh driveclass create [OPTIONS]

  Create a drive class.

  usage example: --name dc1 --drives samsungDriveSerial1 --drives
  intelDriveSerial12 --domains Rack:A --domains DataCenter:DRSite

Options:
  -n, --name TEXT         The name of the drive class  [required]
  -dr, --drives TEXT      Drive serials to group under the drive class
                          [required]
  -d, --description TEXT  The description of the drive class
  --domains TEXT          Domain in the following format: <scope:identifier>
  --help                  Show this message and exit.
```

6.6.2. delete

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

```
  Delete drive classes.

  usage example: -n dc1 -n dc2

Options:
  -n, --names TEXT  The id of the drive class to delete  [required]
  --help            Show this message and exit.
```

6.6.3. show

```
Usage: nvmesh driveclass show [OPTIONS]

  Show all 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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

6.6.4. update

```
Usage: nvmesh driveclass update [OPTIONS]

  Update a drive class.

  usage example: --name dc1 --drives samsungDriveSerial1 --drives
  intelDriveSerial12 --domains Rack:A --domains DataCenter:DRSite

Options:
  -n, --name TEXT         The name of the drive class  [required]
```

```
  -dr, --drives TEXT      Drive serials to group under the drive class
                          [required]
  -d, --description TEXT  The description of the drive class
  --domains TEXT          Domain in the following format: <scope:identifier>
  --help                  Show this message and exit.
```

6.7. target

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

  Group Targets 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.
```

6.7.1. count

```
Usage: nvmesh target count [OPTIONS]

  Get total targets count.

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

6.7.2. delete

```
Usage: nvmesh target delete [OPTIONS]

  Delete targets.

  usage example: -n server-1 -n server-2

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

6.7.3. delete-nic

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

  Delete specific NIC.

  usage example: -n 0xfe80000000000000001e670300932499

Options:
  -n, --name TEXT  ID of the NIC to delete  [required]
  --help           Show this message and exit.
```

6.7.4. show

```
Usage: nvmesh target show [OPTIONS]

  Show all targets.

  --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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

6.8. targetclass

```
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.
```

6.8.1. create

```
Usage: nvmesh targetclass create [OPTIONS]

  Create a target class.

  usage example: --name tc1 -t server-22 -t server-1 --domains Rack:A
  --domains DataCenter:DRSite

Options:
  -n, --name TEXT         The name of the target class  [required]
  -t, --targets TEXT      Targets to group under the target class  [required]
  -d, --description TEXT  The description of the target class
  --domains TEXT          Domain in the following format: <scope:identifier>
  --help                  Show this message and exit.
```

6.8.2. delete

```
Usage: nvmesh targetclass delete [OPTIONS]

  Delete target classes.

  usage example: -n tc1 -n tc2

Options:
  -n, --names TEXT  The id of the drive class to delete  [required]
  --help            Show this message and exit.
```

6.8.3. show

```
Usage: nvmesh targetclass show [OPTIONS]
```

```
  Show all 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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

6.8.4. update

```
Usage: nvmesh targetclass update [OPTIONS]

  Update a target class.

  usage example: --name tc1 -t server-22 -t server-1 --domains Rack:A
  --domains DataCenter:DRSite

Options:
  -n, --name TEXT         The name of the target class  [required]
  -t, --targets TEXT      Targets to group under the target class  [required]
  -d, --description TEXT  The description of the target class
  --domains TEXT          Domain in the following format: <scope:identifier>
  --help                  Show this message and exit.
```

6.9. volume

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

  Group Volumes related operations

Options:
```

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

Commands:
  create           Create volume.
  delete           Delete volumes.
  rebuild-volumes  Rebuild volumes.
  show             Show all volumes.
  update           Update volume.
```

6.9.1. 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. The hierarchy
  is: 1. vpg, 2. target/drive classes, 3. limit by targets/drives. That is,
  if you provide all three options, only the VPG will be considered because
  it is the first in the hierarchy, and so on.

  usage example: -n v11 --raid-level ec -c 100000000000 --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]
  -c, --capacity TEXT             Space in bytes to allocate for the volume.
                                  Use "MAX" for using all of the available
                                  space.  [required]
  -d, --description TEXT          Description of the volume.
  --domain TEXT                   Domain to use.
  --vpg TEXT                      The VPG to use.
  --relative-rebuild-priority INTEGER RANGE
                                  Sets the volume relative rebuild priority.
  -dc, --drive-classes TEXT       Limit volume allocation to specific drive
                                  classes.
  -tc, --target-classes TEXT      Limit volume allocation to specific target
                                  classes.
  --limit-by-drives TEXT          Limit volume allocation to specific drives.
  --limit-by-targets TEXT         Limit volume allocation to specific targets.
```

```
  --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 [full|minimal|ignore]
                                  Protection level to use. Required if RAID
                                  Level is ec. Options: full = Full
                                  Separation, minimal = Minimal Separation,
                                  ignore = Ignore Separation.
  --wait-for-status [online|offline|degraded|initializing]
                                  Wait for the volume to become in a specific
                                  status before continue. Options: online,
                                  offline, degraded, initializing.
  --help                          Show this message and exit.
```

6.9.2. delete

```
Usage: nvmesh volume delete [OPTIONS]

  Delete volumes.

   usage example: -n v1 -n v2

Options:
  -n, --names TEXT     [required]
  -y, --yes            Automatically answer "yes" and skip operational
                       warnings.
  --wait-for-deletion  Wait for the volume to be deleted
  --help               Show this message and exit.
```

6.9.3. rebuild-volumes

```
Usage: nvmesh volume rebuild-volumes [OPTIONS]

  Rebuild volumes.

   usage example: -n v1 -n v2

Options:
  -n, --names TEXT  [required]
  --help            Show this message and exit.
```

6.9.4. show

```
Usage: nvmesh volume show [OPTIONS]

  Show all 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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

6.9.5. update

```
Usage: nvmesh volume update [OPTIONS]

  Update volume.

  usage example: -n v11 -c 200000000000

Options:
  -n, --name TEXT                 Name of the volume. The name must be unique,
                                  as it will become the ID of the volume.
                                  [required]
  -c, --capacity TEXT             Space in bytes to allocate for the volume.
                                  Use "MAX" for using all of the available
                                  space.
  -d, --description TEXT          Description of the volume.
  --relative-rebuild-priority INTEGER RANGE
                                  Sets the volume relative rebuild priority.
  --help                          Show this message and exi
```

6.10. vpg

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

  Group VPGs related operations

Options:
  --help  Show this message and exit.

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

6.10.1. create

```
Usage: nvmesh vpg create [OPTIONS]

  Create VPGs.

  usage example: -n v11 --raid-level ec -c 100000000000 --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]
  -c, --capacity INTEGER          Space in bytes to allocate for the volume.
  -d, --description TEXT          Description of the volume.
  --domain TEXT                   Domain to use.
  -dc, --drive-classes TEXT       Limit volume allocation to specific drive
                                  classes.
  -tc, --target-classes 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 [full|minimal|ignore]
```

```
                                  Protection level to use. Required if RAID
                                  Level is ec. Options: full = Full
                                  Separation, minimal = Minimal Separation,
                                  ignore = Ignore Separation.
  --help                          Show this message and exit.
```

6.10.2. delete

```
Usage: nvmesh vpg delete [OPTIONS]

  Delete VPGs.

   usage example: -n v1 -n v2

Options:
  -n, --names TEXT  [required]
  --help            Show this message and exit.
```

6.10.3. show

```
Usage: nvmesh vpg show [OPTIONS]

  Show all 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: --output-format tabular

Options:
  --output-format [tabular|rows|json]
                                  The representation in which the data will be
                                  displayed. Options: tabular ,rows, json
  --help                          Show this message and exit.
```

Command Line Utilities

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

nvmesh_attach_volumes

Attach the specified volume or list of volumes.

```
usage: attach volume(s) [-h] [--debug] [-w] [-d] [-c]
                        [volumes [volumes ...]]

positional arguments:
  volumes               attached volumes, to be detached

optional arguments:
  -h, --help            show this help message and exit
  --debug         print attach debug information
  -w, --wait_for_attach
                        wait for attach to finish
  -d, --hidden          attach volume as hidden
  -c, --no_cancel       do not cancel attach request on timeout
```

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_detach_volumes

Detach the specified volume or list of volumes.

```
usage: nvmesh_detach_volumes [-h] [--print_debug] [--client_shutdown] [-f] [-d]
                        [--all]
                        [volumes [volumes ...]]

positional arguments:
  volumes            attached volumes, to be detached

optional arguments:
  -h, --help         show this help message and exit
  --print_debug      print detach debug information
```

```
  --client_shutdown  client shutdown
  -f, --force        force detach
  -d, --hidden       detach hidden attached volume
  --all              detach all existing volumes
```

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_target

Configure target_devices.conf options.

> The `add block` functionality is currently not functioning.

```
usage: nvmesh_target [-h] <add|remove|exclude|include> <block|nvme|arbiter> <devi
ce>

positional arguments:
  device                the full NVMe/SAS/SATA device path or device serial numbe
r (see examples)

optional arguments:
  -h, --help            show this help message and exit
  add <device>          add a generic (SAS/SATA/NVMf) block device to the configu
ration
  add arbiter           add an arbitration device
  remove <device>       remove a generic (SAS/SATA/NVMf) block device from the co
nfiguration
```

```
  include nvme <sn>     Include a previously excluded NVMe device
  exclude nvme <sn>

examples:

To exclude NVMe serial number S3HCNX0JC02078 from NVMesh use:

nvmesh_format exclude nvme S3HCNX0JC02078

To add an arbitration device to this node:

nvmesh_format add arbiter

To add /dev/sdb to NVNesh configuration:

nvmesh_format add block /dev/sdb
```

nvmesh_update

Updates NVMesh kernel module.

7. Document Reference

Typographical Conventions

Throughout this document, the following typographical conventions are followed:

Style	Meaning
bold text	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
term in italics	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
Management Server	The server(s), or OS image(s) running the **management module** software
Target Node/ Target	A physical server containing one or more NVMe SSDs running the **storage target module**
Client Node/ Client	An OS image instance running the **block storage client** software
Converged Node	A *target node* that is also running the **block storage client** software
Logical Volume/ Volume	A logical block device defined with the **NVMesh management module** that can be attached to *client nodes*
RDDA	Remote Direct Drive Access. Excelero's patented low-latency and CPU bypass transport technology.
TOMA	**To**pology **Ma**nager. The **storage target module** component that handles error detection and volume rebuild activities.