IBM Storage Ceph 7

NVMe-oF Gateway Support Guide
2023-10-18

Important

DRAFT BETA 1 PROVIDED WITH NO SUPPORT: Do not upgrade a production installation to a beta release. Upgrade support is not provided between beta releases. No automated upgrade path is provided from a beta release to the final GA release. The cluster will need to be rebuilt for production use.

Last updated: 2023-10-18

Contents

Figures

NVMe over Fabrics

Ceph Block Device now offers NVMe-oF gateway support as a Technology Preview. Non-Volatile Memory express (NVMe) and NVMe over Fabrics (NVMe-oF) protocols are designed specifically to enable the full capabilities of parallelism and performance with all-flash storage systems.

Important: Technology Preview features are not supported with IBM production service level agreements (SLAs), might not be functionally complete, and IBM does not recommend using them for production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

NVMe-oF enables the performance of direct attached all-flash storage along with the flexibility and total cost of ownership (TCO) savings of shared storage. Implementing NVMe-oF support in enterprise storage arrays allows the combination of the NVMe protocol performance with the rich feature set of modern storage arrays. Using NVMe protocol enables modern storage arrays to meet growing customer demands.

NVMe-oF uses the TCP protocol as a ubiquitous transport that does not require special network configuration, by using existing Ethernet connectivity. Ethernet speeds currently support up to 400 Gib per second and the use of Ethernet technology is significant in data centers.

For more information about Ceph Block Device NVMe-oF gateway support, see Ceph NVMe-oF gateway.

Ceph NVMe-oF gateway (Technology Preview)

Storage administrators can install and configure an NVMe over Fabrics (NVMe-oF) gateway for an IBM Storage Ceph cluster. With the Ceph NVMe-oF gateway, you can effectively run a fully integrated block storage infrastructure with all features and benefits of a conventional Storage Area Network (SAN).

Important: Technology Preview features are not supported with IBM production service level agreements (SLAs), might not be functionally complete, and IBM does not recommend using them for production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

Traditionally, block-level access to a Ceph storage cluster has been limited to QEMU and `librbd`, which is a key enabler for adoption within OpenStack environments. Starting from IBM Storage Ceph 7, block-level access to the Ceph storage cluster can take advantage of the NVMe-oF standard to provide data storage.

The NVMe-oF gateway integrates IBM Storage Ceph with the NVMe over TCP (NVMe/TCP) protocol to provide an NVMe/TCP target that exports RADOS Block Device (RBD) images. The NVMe/TCP protocol allows clients, which are known as initiators, to send NVMe-oF commands to storage devices, which are known as targets, over an Internet Protocol network.

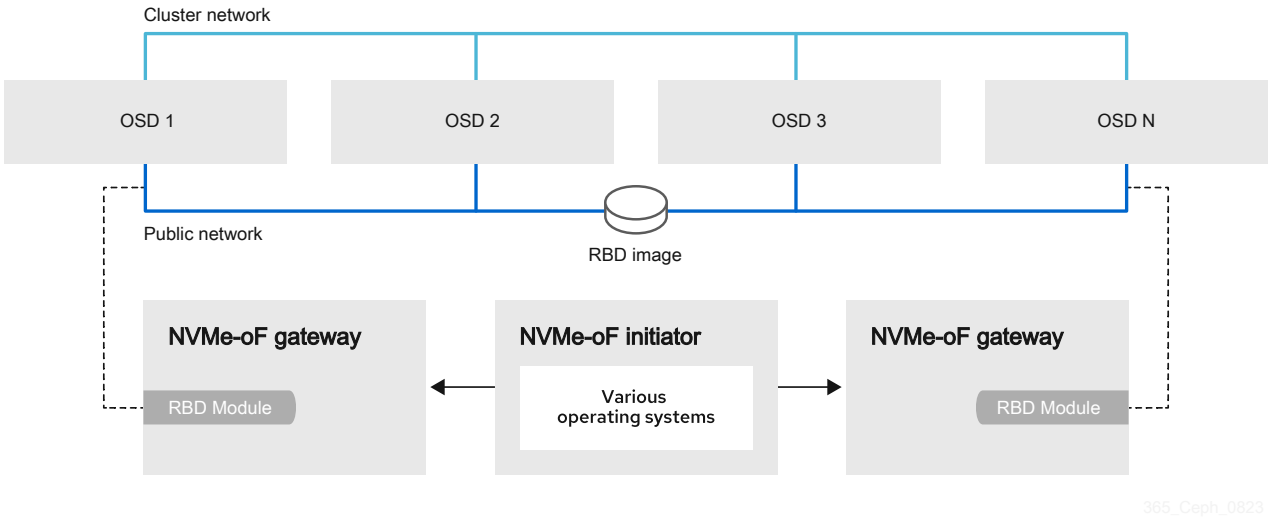

Figure 1. Ceph NVMe-oF gateway

For more information about the NVMe over Fabrics protocol, see NVMe over Fabrics.

Installing the NVMe-oF gateway

Before you can utilize the benefits of the Ceph NVMe-oF gateway, you must install the required software packages. You can install the Ceph NVMe-oF gateway by using the command-line interface.

Each NVMe-oF gateway runs the Storage Performance Development Kit (SPDK) to provide NVMe-oF protocol support. SPDK utilizes a user-space implementation of the NVMe-oF protocol to interact with the Ceph `librbd` library to expose RBD images to NVMe-oF clients. With the Ceph NVMe-oF gateway you can effectively run a fully integrated block storage infrastructure with all features and benefits of a conventional Storage Area Network (SAN).

Installing the Ceph NVMe-oF gateway using the command-line interface

The Ceph NVMe-oF gateway is the NVMe-oF target node and also a Ceph client node. The Ceph NVMe-oF gateway can be a stand-alone node or be collocated on a Ceph Object Storage Daemon (OSD) node.

Before you begin

Installing the NVMe-oF gateway requires a storage system running IBM Storage Ceph 7 or later cluster.

Procedure

Complete the following steps to install the Ceph NVMe-oF gateway.

1. Create a pool in which the gateway group configuration is to be managed.

   ```
   ceph osd pool create NVME-OF_POOL_NAME
   ```

   For example,

   ```
   [root@host01 ~]# ceph osd pool create nvmeof_pool01
   ```

2. Enable the RADOS Block Device (RBD) application on the NVMe-oF pool.

   ```
   rbd pool init NVME-OF_POOL_NAME
   ```

   For example,

   ```
   [root@host01 ~]# rbd pool init nvmeof_pool01
   ```

3. Optional: Work with a custom registry.

   a) Pull the official image of the gateway container from gateway.

   The command can be used with either `podman` or `docker`.

   Note: The `:latest` tag can be replaced with a specific version tag.

   ```
   podman pull quay.io/ceph/nvmeof:latest
   ```

   b) Configure the private registry by adding the `nvmeof-gw:private-image` tag into the official gateway image.

   ```
   ceph config set mgr mgr/cephadm/container_image_nvmeof PRIVATE_REGISTRY_NAME/nvmeof-gw:private-image
   ```

   For example,

   ```
   [root@host01 ~]# ceph config set mgr mgr/cephadm/container_image_nvmeof mycompany-01-reg.org/nvmeof-gw:private-image
   ```

4. Deploy the `nvmeof` manager daemons that use placement specification on a specific set of nodes.

```
ceph orch apply nvmeof NVME-OF_POOL_NAME --placement="NODE1, NODE2,..."
```

For example,

```
[root@host01 ~]# ceph orch apply nvmeof nvmeof_pool01 --placement "node01, node02"
```

Configuring the NVMe-oF gateway target

Configure targets, LUNs, and clients, using the Ceph gateway **nvmeof-cli** command-line utility.

Configuring the NVMe-oF gateway target requires a storage system running IBM Storage Ceph 7 or later cluster and a running Ceph NVMe-oF gateway.

Defining an NVMe-oF subsystem

Define an NVMe-oF subsystem. Defining the NVMe subsystem includes creating an NVMe subsystem, configuring the IP port for communication, and enabling hosts to use the subsystem.

About this task

Configure the Ceph NVMe-oF gateway by using the gateway `nvmeof-cli` utility.

Before you begin

The gateway `nvmeof-cli` container is automatically pulled during the subsystem creation. Alternatively, the `nvmeof-cli` container can be pulled before first use.

To pull the gateway `nvmeof-cli` container, use the following command:

```
podman pull quay.io/ceph/nvmeof-cli:0.0.3
```

Procedure

These commands can be run on either `podman` or `docker`.

1. Create an NVMe subsystem.

   Important: Run this command only *once*. Only one NVMe subsystem is supported.

   ```
   podman run -it <quay.io/ceph/nvmeof-cli:0.0.3> --server-address GATEWAY_NODE_IP --server-
   port 5500 create_subsystem --subnqn SUBSYSTEM_NQN
   ```

   The *SUBSYSTEM_NQN* is a user defined string. In this example it is defined as `nqn.2016-06.io.spdk:cnode1`.

   For example,

   ```
   [root@host01 ~]# podman run -it <quay.io/ceph/nvmeof-cli:0.0.3> --server-address 10.172.19.21
   --server-port 5500 create_subsystem --subnqn nqn.2016-06.io.spdk:cnode1
   ```

   Note: For additional input, `--serial` *SERIAL_NUMBER* can be added to the command.

2. Define the IP port on the gateway that is to process NVMe/TCP commands and I/O operations.

   a) From the installer node, get the NVMe-oF gateway name.

   ```
   ceph orch ps | grep nvme
   ```

   For example,

   ```
   [ceph: root@host01 /]# ceph orch ps | grep nvme

   nvmeof.rbd.host010.nl host010*:5500,4420,8009
   running (13d) - 13d 1970M - 470dd4ee78f0 e313e8b13a31
   ```

In this example, the gateway name is `client.nvmeof.rbd.host010.nl`.

b) Define the IP port on the gateway.

```
podman run -it quay.io/ceph/nvmeof-cli:0.0.3 --server-address NODE_IP --server-port 5500
create_listener -n SUBSYSTEM_NQN -g GATEWAY_NAME -a GATEWAY_NODE_IP -s 4420
```

For example,

```
[root@host01 ~]# podman run -it quay.io/ceph/nvmeof-cli:0.0.3 --server-address 10.172.19.01
--server-port 5500 create_listener -n nqn.2016-06.io.spdk:cnode1 -g client.nvmeof.test-
nvmeof-pool.rhel9-ceph-node1.ogqdhx -a 10.172.19.01 -s 4420
```

3. Get the host NVMe Qualified Name (NQN) for each host.

For Red Hat Enterprise Linux initiators

```
cat /etc/nvme/hostnqn
```

For example,

```
# cat /etc/nvme/hostnqn
nqn.2014-08.org.nvmexpress:uuid:950ddadf-f995-47b7-9416-b9bb233f66e3
```

where `950ddadf-f995-47b7-9416-b9bb233f66e3` is the UUID.

For VMware ESXi initiators

```
esxcli nvme info get
```

For example,

```
esxcli nvme info get
Host NQN: nqn.2014-08.com.ibm.ceph:nvme:host01
```

4. Allow the NVMe initiator host to run NVMe/TCP commands to the newly created NVMe subsystem.

a) Get the host NVMe Qualified Name (NQN) for each host.

Note: Specific hosts can be selected, as in the following example. To specify all hosts, use `--host "*"`.

```
podman run -it quay.io/ceph/nvmeof-cli:0.0.3 --server-address NODE_IP --server-port 5500
add_host --subnqn SUBSYSTEM_NQN --host "HOST01_NQN,
HOST02_NQN"
```

For example,

```
[root@host01 ~]# podman run -it quay.io/ceph/nvmeof-cli:0.0.3 --
server-address 10.172.19.01 --server-port 5500 add_host --subnqn
nqn.2016-06.io.spdk:cnode1 --host "nqn.2014-08.org.nvmexpress:uuid:950ddadf-f995-47b7-9416-
b9bb233f66e3,nqn.2014-010.org.nvmexpress:uuid:960ddadf-f995-47b7-9416-b9bb233f66e3"
```

Defining block devices to use NVMe/TCP

Define storage block devices to be used with NVMe/TCP.

About this task

Note: This procedure only needs to be run once per block device, even if there are multiple gateways.

Procedure

1. Create an image within any RBD application-enabled pool.

```
rbd create IMAGE_NAME --size MEGABYTES --pool POOL_NAME
```

For example,

```
[root@host01 ~]# rbd create image-1 --size 1024 --pool pool01
```

For more information about creating an image, see Creating images.

2. Using the gateway `nvmeof-cli` utility, create a block device.

   Run this command for every RBD image that needs to be exposed through NVMe/TCP.

   ```
   podman run -it <quay.io/ceph<nvmeof-cli:0.0.3> --server-address GATEWAY_NODE_IP --server-
   port 5500 create_bdev --pool POOL_NAME --image IMAGE_NAME --bdev BLOCK_DEVICE_NAME
   ```

   For example,

   ```
   [root@host01 ~]# podman run -it <quay.io/ceph<nvmeof-cli:0.0.3> --server-address 10.172.19.21
   --server-port 5500 create_bdev --pool pool01 --image image-01 --bdev ceph-block01
   ```

3. Add a namespace to each block device that was created in “Defining an NVMe-oF subsystem” on page 12.

   ```
   podman run -it <quay.io/ceph/nvmeof-cli:0.0.3> --server-address GATEWAY_NODE_IP --server-
   port 5500 add_namespace --subnqn SUBSYSTEM_NQN --bdev BLOCK_DEVICE_NAME
   ```

   For example,

   ```
   [root@host01 ~]# podman run -it <quay.io/ceph/nvmeof-cli:0.0.3> --server-address 10.172.19.21
   --server-port 5500 add_namespace --subnqn nqn.2016-06.io.spdk:cnode1 --bdev ceph-block01
   ```

Configuring the NVMe-oF gateway initiator

Configure the initiator to allow the NVMe/TCP protocol to send NVMe-oF commands to targets over an Internet Protocol network.

The NVMe-oF gateway initiator can be configured on either of the following platform version:

- Red Hat Enterprise Linux 9.2 or later
- VMware vSphere Hypervisor (ESXi) 7.0U3 or later

Configuring the NVMe-oF initiator for Red Hat Enterprise Linux

Configure the NVMe-oF initiator for Red Hat Enterprise Linux.

Before you begin

- Red Hat Enterprise Linux 9.2 or later

Procedure

1. Install the `nvme-cli`.

   ```
   yum install nvme-cli
   ```

2. Verify that the `nvme-tcp` driver is loaded.

   The output should contain `nvme-tcp`.

   ```
   cat /etc/modules-load.d/nvme-tcp.conf
   ```

   Note: If `nvme-tcp` is not displayed, create a `etc/modules-load.d/nvme-tcp.conf` file containing `nvme-tcp`.

3. Install the necessary NVMe packages.

```
modprobe nvme-fabrics
```

4. Verify that the target is reachable from the initiator.

```
nvme discover -t tcp -a GATEWAY_IP -s 4420
```

For example,

```
[root@host01 ~]# nvme discover -t tcp -a 10.172.19.01 -s 4420
```

5. Connect to the NVMe-oF target.

```
nvme connect -t tcp -a GATEWAY_IP -n SUBSYSTEM_NQN
```

For example,

```
[root@host01 ~]# nvme connect -t tcp -a 10.172.19.01 -n nqn.2016-06.io.spdk:cnode1
```

What to do next

Verify that the initiator is set up correctly.

1. List the NVMe-oF block devices.

```
nvme list
```

For example,

```
[root@host01 ~]# nvme list
Node                  Generic           SN                   Model
Namespace Usage                      Format           FW Rev
--------------------  ----------------  -------------------  ----------------------
--------- -------------------------- ---------------- --------
/home/nvme01_node01     /home/ng1n1       SPDK00000000000001   SPDK bdev Controller
1          10,49  MB /  10,49  MB      4 KiB +  0 B   23.01
...
```

2. Create a filesystem on the desired target, found in step “1” on page 15.

```
mkfs NVME_NODE_PATH
```

For example,

```
[root@host01 ~]# mkfs /home/nvme01_node01
mke2fs 1.46.5 (20-Dec-2023)
Discarding device blocks: done
Creating filesystem with 2560 4k blocks and 2560 inodes

Allocating group tables: done
Writing inode tables: done
Writing superblocks and filesystem accounting information: done
```

3. Mount the NVMe node on the NVMe-oF directory.
   a. Mount NVMe-oF.

```
mkdir /mnt/nvmeof
```

For example,

```
[root@host01 ~]# mkdir /mnt/nvmeof
```

   b. Mount the node on within the NVMe-oF directory.

```
mount NVME_NODE_PATH /mnt/nvmeof
```

For example,

```
[root@host01 ~]# mount /home/nvme01_node01 /mnt/nvmeof
```

4. Using sudo commands, list mounted NVMe-oF files.

```
ls /mnt/nvmeof
```

For example,

```
$ ls /mnt/nvmeof
lost+found
```

5. Create a text file within the `mnt/nvmeof` directory.

For example,

```
$ sudo bash -c "echo Hello NVMe-oF > /mnt/nvmeof/hello.txt"
```

6. Verify that the text file can now be reached.

For example,

```
$ cat /mnt/nvmeof/hello.txt
Hello NVMe-oF
```

Configuring the NVMe-oF initiator for VMware ESXi

Configure the NVMe-oF initiator for VMware vSphere Hypervisor (ESXi). You can set up a VMware ESX host as a NVMe/TCP initiator.

About this task

NVMe/TCP is supported by VMware vSphere Hypervisor (ESXi) 7.0U3 or later.

Before you begin

- A VMware ESXi host running VMware vSphere Hypervisor (ESXi) 7.0U3 version or later.
- Ceph NVMe-oF gateway deployed.
- Ceph cluster and `ceph-nvmeof` configuration is ready and healthy.
- A subsystem defined within the gateway. For more information, see “Defining an NVMe-oF subsystem” on page 12.
- NVMe/TCP adapter is configured.
  - Enabled NVMe/TCP on a physical network interface controller (NIC).

    ```
    esxcli nvme fabrics enable --protocol TCP --device vmnicN
    ```

    Here, N is the number of NIC.
  - Tag a VMkernel NIC to permit NVMe/TCP traffic.

    ```
    sxcli network ip interface tag add --interface-name vmk1 --tagname NVMeTCP
    ```

Procedure

Configuring the VMware ESXi host for NVMe/TCP transport includes discovering the NVMe/TCP targets and connecting to them.

1. List the NVMe-oF adapter.

```
esxcli nvme adapter list
```

For example,

```
$ esxcli nvme adapter list
Adapter  Adapter Qualified
Name                                                                    Transport Type
Driver     Associated Devices
-------
-------------------------------------------------------------------------------------
-------------  ---------  ------------------
vmhba64  aqn:nvmetcp:ac-1f-6b-0a-18-74-
T                                                       TCP             nvmetcp    vmnic0
```

2. Discover any NVMe-oF-gateway subsystems.

```
esxcli nvme fabrics discover -a NVME_TCP_ADAPTER -i GW_NODE_IP -p PORT
```

For example,

```
[root@host01:~] esxcli nvme fabrics discover -a vmhba64 -i 10.0.211.196 -p port01

Transport Type  Address Family  Subsystem Type  Controller ID  Admin Queue Max Size  Transport
Address  Transport Service ID  Subsystem NQN               Connected
-------------  -------------  -------------  -------------  --------------------
----------------  --------------------  -------------------------  ---------
TCP             IPv4            NVM                      65535                  128
10.0.211.196       5001                 nqn.2016-06.io.spdk:cnode1      false
TCP             IPv4            NVM                      65535                  128
10.0.211.196       5002                 nqn.2016-06.io.spdk:cnode2      false
```

3. Connect to NVMe-oF gateway subsystem.

```
esxcli nvme fabrics connect -a NVME_TCP_ADAPTER -i GW_NODE_IP -p PORT -s SUBSYSTEM_NQN
```

For example,

```
[root@host01:~] esxcli nvme fabrics connect  -a vmhba64 -i 10.0.211.196 -p port01 -s
nqn.2016-06.io.spdk:cnode1

[root@argo010:~] esxcli nvme fabrics discover -a vmhba64 -i 10.0.211.196 -p 5001
Transport Type  Address Family  Subsystem Type  Controller ID  Admin Queue Max Size  Transport
Address  Transport Service ID  Subsystem NQN               Connected
-------------  -------------  -------------  -------------  --------------------
----------------  --------------------  -------------------------  ---------
TCP             IPv4            NVM                      65535                  128
10.0.211.196       5001                 nqn.2016-06.io.spdk:cnode1       true
TCP             IPv4            NVM                      65535                  128
10.0.211.196       5002                 nqn.2016-06.io.spdk:cnode2      false
```

4. List NVMe/TCP controller list.

```
esxcli nvme controller list
```

```
[root@host01:~] esxcli nvme controller list
Name
Controller Number  Adapter  Transport Type  Is Online
-------------------------------------------------------------------------------------------
----------------  -------  --------------  ---------
nqn.2016-06.io.spdk:cnode1#vmhba64#10.0.211.196:5001
           301  vmhba64  TCP                true
```

5. List NVMe-oF namespaces in the subsystem.

```
esxcli nvme namespace list
```

For example,

```
[root@host01:~] esxcli nvme namespace list
Name                                  Controller Number  Namespace ID  Block Size  Capacity in MB
------------------                    -----------------  ------------  ----------  --------------
eui.0100000001000000e4d25c00001ae214               256             1         512
953869
eui.01abc123def456g7e4d25c00001ae214               301             1         512
500
eui.02abc123def456g7e4d25c00001ae215               301             2         512
500
eui.03abc123def456g7e4d25c00001ae216               301             3         512
500
```

What to do next

Verify that the initiator is set up correctly.

- Validate that all namespaces match with the UUID of namespaces on the gateway node. See `get_subsystems` output for the subsystem connected from the ESXi host.
- Compare the output of the **`esxcli nvme namespace list`** command with the following command:

```
podman run -it <quay.io/ceph<nvmeof-cli:0.0.3> --server-address GW_NODE_IP --server-port 5500
get_subsystems
```