Configuring NVMe/TCP with AWS EC2 Instance Running Red Hat Enterprise Linux on VSP One Block 20 NVM Subsystems

v2.0

Configuration Guide

This document shows how to configure NVMe/TCP on Hitachi Virtual Storage Platform One Block 20 (VSP One B20) family storage systems with AWS EC2 instance running RHEL operating systems. This document applies to VSP One B24, B26, and B28 storage systems.

MK-SL-412-02
November 2025

Table of Contents

Preface

About this document

This document shows how to configure NVMe/TCP with native NVMe multipathing for Hitachi Virtual Storage Platform One Block 20 (VSP One B20) NVM subsystems in an AWS EC2 instance running Red Hat Enterprise Linux (RHEL) operating system.

Hitachi Command Control Interface (CCI) is used to configure the storage system. Linux configurations can be applied to RHEL 9.4 and other versions with minimal or no changes. For more information, see the OS vendor manuals.

Release notes

Read the release notes and support matrix documents for Hitachi VSP One Block 20 family storage systems. It may contain requirements or restrictions that are not fully covered in this document.

Additionally, read the AWS documentation, RHEL Operating Systems release documents, user guide, installation and support documents for detailed configuration steps, restrictions, updates, and corrections.

Revision History

Revision	Changes	Date
v1.0	Initial Release	December 2024
v2.0	Added volume specification	November 2025

Accessing product documentation

Product user documentation is available on the Hitachi Vantara Support Site: https://docs.hitachivantara.com/.
Visit this site for the most current documentation, including important updates made after the product release. For more information on the operating system, storage systems and AWS, refer to the respective documentation.
The Product Compatibility Guide can be found at https://compatibility.hitachivantara.com/.
For RHEL OS support, visit their support site: https://access.redhat.com/products/red-hat-enterprise-linux.
AWS documentation is available on the following site: https://docs.aws.amazon.com/

Getting Help

Hitachi Vantara Support Connect is the destination for technical support of products and solutions sold by Hitachi Vantara. To contact technical support, log on to Hitachi Vantara Support Connect for contact information: https://support.hitachivantara.com/en_us/contact-us.html.

Hitachi Vantara Community is a global online community for customers, partners, independent software vendors, employees, and prospects. It is the destination to get answers, discover insights, and make connections. **Join the conversation today!** Go to community.hitachivantara.com, register, and complete your profile.

Comments

Please send us your comments on this document to Docs-Feedback@hitachivantara.com. Include the document title and number, including the revision level (for example, -07), and refer to specific sections and paragraphs whenever possible. All comments become the property of Hitachi Vantara LLC.

Introduction

This document shows how to configure NVMe/TCP with NVMe native multipathing for Hitachi Virtual Storage Platform One Block 20 (VSP One B20) family storage NVM subsystem in a Red Hat Enterprise Linux environment on an AWS EC2 instance. VSP One Block 20 is a 2U all-NVMe storage appliance with three dedicated models. All models have the same capacity and support FC, iSCSI, and NVMe TCP connectivity. The storage models are as follows:

- VSP One Block 24 – 256GB Cache + SW Advanced Data Reduction (ADR) + 24 cores
- VSP One Block 26 – 768GB Cache + 2x Compression Accelerator Module (CAM) + 24 cores
- VSP One Block 28 – 1TB Cache + 4x CAM + 64 cores

NVMe/TCP sets up how NVMe queues and data are handled over the standard TCP network protocol, while offering extra features such as data integrity checks (DIGEST) and Transport Layer Security (TLS) for added security. In addition, NVMe/TCP allows smooth and efficient NVMe operations between hosts and controllers over any standard IP network, with great performance and low delay. This means large data centers can use their existing Ethernet networks, including complex switch setups and regular Ethernet adapters.

Purpose

This document provides instructions for configuring NVMe/TCP on an AWS EC2 instance running RHEL OS environment. In addition, this document shows how to set up RHEL 9.4 on AWS EC2, including configuring Hitachi VSP One Block 20 family storage system with NVMe/TCP interfaces.

The purpose of this guide is to provide an overview of the implementation process, including VSP One Block 20 NVMe/TCP port configuration, subsystem ID creation, NVMe namespace provisioning, host-side configuration for storage namespace detection, native NVMe multipathing configuration, and I/O operation timeout value configuration.

Configuration Diagram

Figure 1 shows the environment used to validate compatibility between Hitachi VSP One Block 28 and AWS EC2 running RHEL 9.4 with NVMe\TCP configuration over 100G Direct Connect.

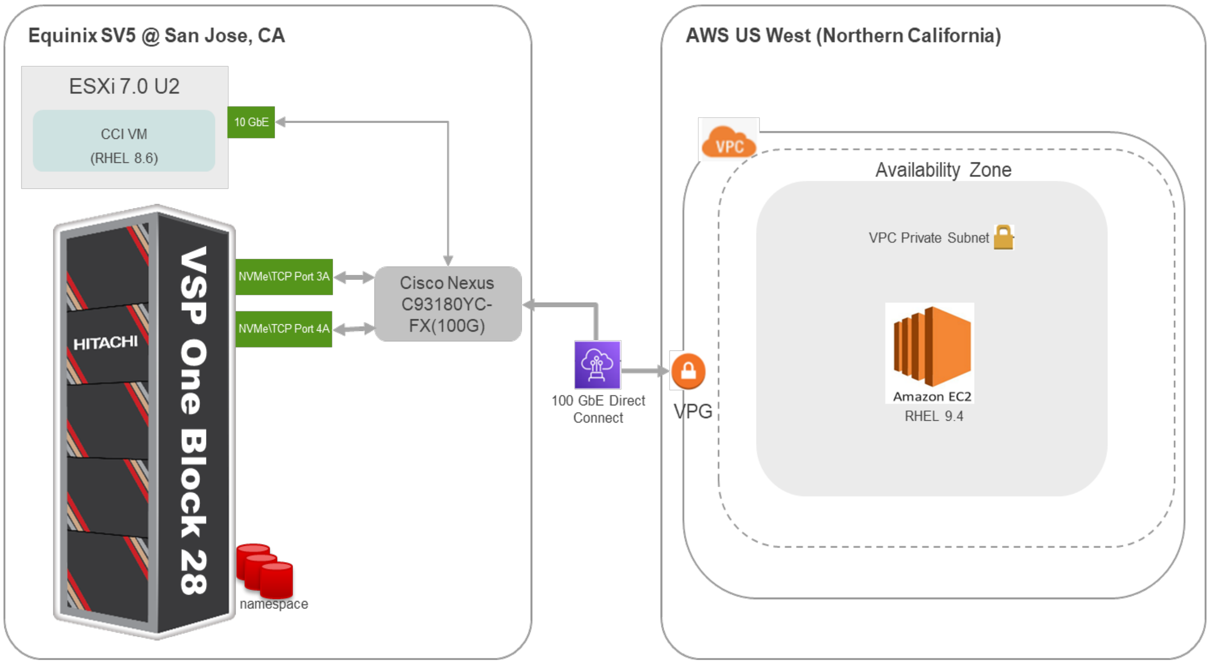

Figure 1: High Level Configuration Diagram

Hardware Requirements

The following lists the hardware used in this setup:

- NVMe/TCP host system: AWS EC2 instance (Type: c6in.16xlarge to support 100G network performance)
- Storage system: Hitachi VSP One Block 28
- Switch: Cisco Nexus C93180YC-FX
- CCI Host: RHEL 8.6 VM hosted in ESXi 7.0 Update 2 OS server

Software Requirements

The following lists the software used in this setup:

- OS install media: RHEL 9.4
- CCI Software: Command Control Interface version 01-76-03/02

AWS 100GbE Direct Connect

AWS Direct Connect is a networking service that provides a dedicated, high-speed connection between your on-premises data center and AWS. AWS Direct Connect allows you to establish private connectivity to AWS services and Amazon VPC, bypassing the public internet for more consistent performance, increased bandwidth, and reduced network costs. With 100GbE AWS direct connect, the service supports bandwidth of up to 100 gigabits per second, offering fast, reliable, and low-latency data transfer, ideal for applications with high data transfer requirements.

To set up an AWS 100GbE Direct Connect connection, complete the following steps:

1. Set up an account in AWS if do not have one. Secure your root user, enable AWS IAM Identity Center, and create an administrative user for daily tasks.

   Using the AWS Direct Connect console, submit a connection request with the port speed as 100Gbps and mention the location where you want to set up your connection. After submitting the connection request, AWS will review and send you a Letter of Authorization and Connecting Facility Assignment (LOA-CFA) which authorizes you to connect to AWS. This is needed by your colocation or network provider to set up the cross-network connection (cross-connect).

   This LOA-CFA will be sent to Equinix to request an Equinix cross connect. Equinix cross connect offers direct links to many networks and enterprises, cloud services, and content providers within Equinix IBX data centers, helping network providers to distribute content, exchange data, and deploy cloud apps, while reducing latency by connecting to various infrastructures in the data center.

2. After Equinix provisions the cross connect request, create a patch panel connection to an appropriate network switch from Equinix SV5 (where the cross connect was patched).

3. To start using your AWS Direct Connect connection, do the following on the AWS Direct Connect console:

   a. Create a virtual interface and attach it to the Direct Connect Gateway.

   b. Attach the Direct Connect gateway to a transit gateway.

   c. Configure Border Gateway Protocol (BGP) session on both AWS and the network switch to manage routing between AWS and your on-premises network.

For more information, see:

- AWS Direct Connect: https://docs.aws.amazon.com/directconnect/
- Equinix Cross Connects: https://docs.equinix.com/en-us/Content/Interconnection/Cross_Connects/xc-cross-connects.htm

Installing NVMe and Configuring IP Addresses for AWS EC2 Instance with RHEL OS

This section describes how to install and configure the NVMe package on an AWS EC2 instance running RHEL OS after an instance is created, and how to verify the hostnqn and IP addresses.

Before you begin:

- Create an AWS EC2 instance with an OS (in this setup, we used RHEL 9.4 OS) by following the instructions in the vendor installation documents.
- Verify that the network interface is detected by the OS inbox ENA driver. If required, add a network interface and install driver to the AWS EC2 instance.

1. Verify whether NVMe-CLI is already installed in the AWS EC2 instance running RHEL 9.4, as shown in the following example:

   ```
   # rpm -qa | grep nvme
   ```

2. If the NVMe-CLI package is not installed, install the package manually by creating a local repository using the installation media ISO.

   In this scenario, we installed the NVMe-CLI package because the NVMe-CLI package was not installed.

   ```
   #yum install nvme-cli
   ```

3. Verify that the hostnqn is generated by running the following command:

   ```
   # cat /etc/nvme/hostnqn
    nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-7c2314fece0f
   ```

   If the hostnqn is not present, run the following command to generate hostnqn and reverify:

   ```
   #nvme gen-hostnqn > /etc/nvme/hostnqn
   ```

4. Configure the Private IP addresses in the network interface on the EC2 instance host. You can configure this while creating an EC2 instance, or you can add a network interface with the required IP addresses after creating an EC2 instance.
5. Verify the IP addresses assigned on the ethernet interface and configure the MTU value if needed by running the following command:

   ```
   #ip link set dev eth1 mtu 9000
   ```

6. To reflect the MTU value change, restart the EC2 instance.
7. Verify the IP address by running the following command:

   ```
   # ip addr show
   2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP group default
   qlen 1000
       link/ether 06:a2:de:af:ce:bd brd ff:ff:ff:ff:ff:ff
   ```

```
    altname enp0s5
    altname ens5
    inet 10.77.23.68/26 brd 10.77.23.127 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::4a2:deff:feaf:cebd/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP group default
qlen 1000
    link/ether 06:9e:29:26:e0:c3 brd ff:ff:ff:ff:ff:ff
    altname enp0s6
    altname ens6
    inet 10.77.23.69/26 brd 10.77.23.127 scope global noprefixroute eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::49e:29ff:fe26:e0c3/64 scope link
       valid_lft forever preferred_lft forever
```

One IP address is for management access and the other one is for IO operations used in the setup.

8. Verify network connectivity to ensure that the target IP addresses are pinging successfully from the host without any packet loss:

```
# ping 172.23.30.183
PING 172.23.30.183 (172.23.30.183) 56(84) bytes of data.
64 bytes from 172.23.30.183: icmp_seq=1 ttl=61 time=52.5 ms
64 bytes from 172.23.30.183: icmp_seq=2 ttl=61 time=1.77 ms
64 bytes from 172.23.30.183: icmp_seq=3 ttl=61 time=1.81 ms
64 bytes from 172.23.30.183: icmp_seq=4 ttl=61 time=1.77 ms
64 bytes from 172.23.30.183: icmp_seq=5 ttl=61 time=1.85 ms
64 bytes from 172.23.30.183: icmp_seq=6 ttl=61 time=1.85 ms
^C
--- 172.23.30.183 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5010ms
rtt min/avg/max/mdev = 1.766/10.258/52.507/18.894 ms
```

Configuring the Storage System for NVMe/TCP

This section describes the process for configuring the Hitachi VSP One Block 20 storage NVM subsystem with RAID Manager (CCI). CCI host must be running version 01-75-03/03 or later. For detailed information on installation, configuration, and CCI commands, see the RAID Manager manuals.

Configuring CCI

To configure CCI, complete the following steps:

1. Update the CCI management server to the supported release:

```
Model   : RAID-Manager/Linux/x64
Ver&Rev: 01-76-03/02
```

2. Configure the HORCM configuration file for the NVMe storage system, as shown in the following example:

```
# cat /etc/horcm128.conf
HORCM_MON
#ip address         service        poll(10ms)   timeout(10ms)
127.0.0.1           horcm128       1000         3000

HORCM_CMD
\\.\IPCMD-172.23.30.107-31001

# cat /etc/services | grep horcm128
horcm128         49000/tcp
[root@CCI-vm etc]#

# horcmstart.sh 128
HORCM inst 128 has already been running.
```

In this example, `horcm128.conf` is the storage configuration file and 172.23.30.107 is one of the storage controller IP.

3. Log in to the storage system by running the following command:

```
# raidcom -login <username> <password> -IH128

# ps -ef | grep horcm
root        5480       1  0 Apr24 ?        00:01:09 horcmd_0128
root     3437871 3415991  0 05:52 pts/1    00:00:00 grep --color=auto horcm
```

Configuring the Storage Port

To configure the 100G TCP storage port, complete the following steps:

1. Verify the settings and port mode for the storage port using the RAID Manager command as follows:

```
# raidcom get port -key detail -I128
PORT   TYPE   ATTR   SPD  LPID  FAB  CONN  SSW   SL   Serial#  WWN      PHY_PORT PORT_MODE RSGID
SECURITY
CL3-A  NVME_TCP TAR    100G    00  N     UNKN  N       0     810138  -     -         NVME_TCP       0 -
CL4-A  NVME_TCP TAR    100G    00  N     UNKN  N       0     810138  -     -         NVME_TCP       0 -
```

By default, the port mode of 100G CHB is NVME_TCP because the default support protocol is NVMe/TCP.

2. Configure the storage ports topology.

    a. Assign network IP addresses to the 100G TCP storage ports in the required subnet ranges.

    b. Modify MTU to jumbo frame and windows size as required for the setup.

    In the following example, network IP addresses, 172.23.30.183 and 172.23.30.184, are assigned against CL3-A and CL4-A 100G storage ports. Additionally, maximum windows size and jumbo frame are enabled.

```
# raidcom modify port -port CL3-A -ipv4_address 172.23.30.183 -
ipv4_subnetmask 255.255.254.0 -ipv4_gateway_address 172.23.30.1 -I128

# raidcom modify port -port CL$-A -ipv4_address 172.23.30.184 -
ipv4_subnetmask 255.255.254.0 -ipv4_gateway_address 172.23.30.1 -I128

# raidcom modify port -port CL3-A -mtu 9000 -window_size 2048k -I128

# raidcom modify port -port CL4-A -mtu 9000 -window_size 2048k -I128
```

3. Verify the IP address configuration by running the following command:

```
# raidcom get port -port CL3-A -key opt -I128
PORT : CL3-A
TCP_OPT : IPV6_D : SACK_E : DACK_E : INS_D : VTAG_D
TCP_MTU : 9000
WSZ : 2048KB
TCP_PORT_IO_CONTROLLER : 4420
TCP_PORT_DISCOVERY_CONTROLLER : 8009
IPV4_ADDR : 172.23.30.183
IPV4_SMSK : 255.255.254.0
IPV4_GWAD : 172.23.30.1
IPV6_ADDR_INF : INV : AM : fe80::
IPV6_GADR_INF : INV : AM : ::
IPV6_GWAD_INF : INV : :: : ::
VLAN_ID : -
VP_MODE : D
IPV6_GADR2_INF : INV : AM : ::
MAC_ADDR : f8:48:97:47:e5:75
[root@CCI-vm etc]#
```

```
# raidcom get port -port CL4-A -key opt -I128
PORT : CL4-A
TCP_OPT : IPV6_D : SACK_E : DACK_E : INS_D : VTAG_D
TCP_MTU : 9000
WSZ : 2048KB
TCP_PORT_IO_CONTROLLER : 4420
TCP_PORT_DISCOVERY_CONTROLLER : 8009
IPV4_ADDR : 172.23.30.184
IPV4_SMSK : 255.255.254.0
IPV4_GWAD : 172.23.30.1
IPV6_ADDR_INF : INV : AM : fe80::
IPV6_GADR_INF : INV : AM : ::
IPV6_GWAD_INF : INV : :: : ::
VLAN_ID : -
VP_MODE : D
IPV6_GADR2_INF : INV : AM : ::
MAC_ADDR : f8:48:97:47:e4:e9
[root@CCI-vm etc]#
```

Configuring the NVM Subsystem

To configure the NVM subsystem, complete the following steps:

1. Create an NVM subsystem on the storage system with the supported host mode of the operating system using the RAID Manager command.

   a. Before creating an NVM subsystem, verify whether there are any existing NVM subsystems as follows:

   ```
   # raidcom get nvm_subsystem -key opt -I128
   ```

   In the following example, the NVM subsystem is created with NVM subsystem ID 1:

   ```
   # raidcom add nvm_subsystem -nvm_subsystem_id 1 -nvm_subsystem_name Junotcpnvme1 -
   host_mode LINUX -request_id auto -I128
   ```

   b. Verify the details of newly created NVM subsystem ID 1:

   ```
   # raidcom get nvm_subsystem -nvm_subsystem_id 1 -I128
   NVMSS_ID  RGID  NVMSS_NAME                   SECURITY T10PI    HMD          HMO_BITs
          1     0  Junotcpnvme1                 ENABLE   DISABLE  LINUX/IRIX   -
   ```

   ```
   # raidcom get nvm_subsystem -key opt -I128
   NVMSS_ID  NVMSS_NAME                   NVMSS_NQN
          1  Junotcpnvme1                 nqn.1994-04.jp.co.hitachi:nvme:storage-
   subsystem-sn.8-10138-nvmssid.00001
   ```

2. Register the 100G TCP storage ports with the NVM subsystem using the CCI raidcom command.

   In the following example, storage ports CL3-A and CL4-A ports are registered with NVM subsystem ID 1:

   ```
   # raidcom add nvm_subsystem_port -nvm_subsystem_id 1 -port CL3-A -request_id auto -I128

   # raidcom add nvm_subsystem_port -nvm_subsystem_id 1 -port CL4-A -request_id auto -I128

   # raidcom get nvm_subsystem_port -nvm_subsystem_id 1 -I128
   PORT   NVMSS_ID  NVMSS_NAME
   CL3-A         1  Junotcpnvme1
   CL4-A         1  Junotcpnvme1
   ```

3. Configure namespace security settings.

   a. Verify the Host NQN and note it for configuring the Host NQN settings in the NVM subsystem.

   b. For a Linux system, you can retrieve the Host NQN using the following command:

   ```
   # cat /etc/nvme/hostnqn
   nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-7c2314fece0f
   ```

   c. Configure the Host NQN to allow access to the NVM subsystem using the RAID Manager command as follows:

   ```
   # raidcom add host_nqn -nvm_subsystem_id 1 -host_nqn nqn.2014-
   08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-7c2314fece0f -request_id auto -I128
   ```

```
# raidcom get host_nqn -nvm_subsystem_id 1 -I128
NVMSS_ID  NVMSS_NAME                    HOST_NQN
       1  Junotcpnvme1                  nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-
a7ad-7171-7c2314fece0f
```

4. Create a namespace by registering the LDEV with the NVM subsystem. After registering the LDEV with the NVM subsystem, a namespace with a unique Namespace ID is created for that LDEV.

   a. Create new LDEVs if there is no free LDEV.

      In the following example, an LDEV of LDEV ID 01:20 is created and formatted from an already existing pool ID 1:

      ```
      # raidcom add ldev -pool 1 -ldev_id 01:20 -capacity 10G -IH128
      ```

      ```
      # raidcom initialize ldev -ldev_id 01:20 -operation fmt -IH128
      ```

      Starting with SVOS 10.4.1, you must create an LDEV (DRS Volume) from a pool as a **mandatory step** for storage provisioning on VSP One Block 20 series storage systems.

      Run the following commands to create a DRS volume:

      Create a pool:

      ```
      raidcom add dp_pool -pool_id <pool_id> -pool_name <pool_name> -ldev_id
      <pool_volume_id> -IH<horcm_instance>
      ```

      Create virtual LDEV on the pool:

      ```
      raidcom add ldev -pool <pool_id> -ldev_id <virtual_volume_id> -capacity
      <virtual_Volume_capacity> -capacity_saving deduplication_compression -drs -request_id
      auto -IH<horcm_instance>
      ```

      For more information on LDEV specification or latest recommendation related to storage provisioning, see the storage provision guide for the specific SVOS level:

      https://docs.hitachivantara.com/r/en-us/svos/10.4.x/mk-23vsp1b012/planning-for-portconnections/nvme/tcp-ports/requirements-for-volumes-used-for-storing-user-data

   b. Create a namespace for the newly created LDEVs and register it with storage NVM subsystem ID as follows:

      ```
      #raidcom add namespace -nvm_subsystem_id 1 -ns_id auto -ldev_id 01:20 -request_id auto
      -I128
      ```

   c. Verify that the namespaces are created for the LDEVs in the storage subsystem:

      ```
      # raidcom get namespace -nvm_subsystem_id 1 -I128
      NVMSS_ID  NVMSS_NAME                    NSID  LDEVID  CAPACITY(BLK)
             1  Junotcpnvme1                     1     288       20971520
      ```

5. Configure the namespace path settings.

   Register the Host NQN to the namespace ID to allow host access to the namespace. Configure the Host NQN-Namespace Path by running the RAID manager command, as shown in the following example:

```
# raidcom add namespace_path -nvm_subsystem_id 1 -ns_id 1 -host_nqn nqn.2014-
08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-7c2314fece0f -request_id auto -I128
```

6. Verify the namespace path information by running the following RAID Manager command:

```
# raidcom get namespace_path -nvm_subsystem_id 1 -I128
NVMSS_ID  NVMSS_NAME                  NSID  LDEV#  HOST_NQN
       1  Junotcpnvme1                   1    288  nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-
a7ad-7171-7c2314fece0f
```

7. Verify the host NQN login status on the storage ports:

```
# raidcom get port -port CL3-A -key login_host_nqn -I128
PORT     LOGIN_STATUS   HOST_NQN
CL3-A    LOGIN         nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-
7c2314fece0f
```

```
# raidcom get port -port CL4-A -key login_host_nqn -I128
PORT     LOGIN_STATUS   HOST_NQN
CL4-A    LOGIN         nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-
7c2314fece0f
```

Configuring the Host for Target Discovery and Connect

This section describes the process for configuring the NVMe target discovery and connect from an AWS EC2 instance running RHEL to enable communication with the Hitachi VSP One Block 20 storage system.

Before configuring NVMe target discovery, verify the following:

- NVMe-CLI package is installed on the EC2 instance running RHEL OS.
- Physical network connectivity is established between the AWS 100G direct connect endpoint and the storage port through the 100G Port of the Network Switch.
- Network IP addresses are assigned to the NIC ports on the EC2 instance.
- Hitachi VSP One Block 20 storage system is configured, namespaces for the LDEVs in the storage NVM subsystem are created, and the namespace path is defined with the Host NQN.

NVMe Target Discovery and Target Connect

1. Load the ‘nvme_tcp’ module if it is not already loaded:

```
# modprobe nvme_tcp
```

2. Discover the NVMe controllers by running the following NVMe discovery command:

   Syntax:

```
nvme discover --transport=tcp --traddr=<target controller IP> --host-traddr=<Host Netrowk
IP address> --trsvcid=8009
```

Note: Port 8009 is the default TCP port for NVMe/TCP discovery controllers.

Example:

```
# nvme discover --transport=tcp --traddr=172.23.30.183 --host-traddr=10.77.23.69 --
trsvcid=8009
Discovery Log Number of Records 1, Generation counter 5
=====Discovery Log Entry 0======
trtype:  tcp
adrfam:  ipv4
subtype: nvme subsystem
treq:    not specified, sq flow control disable supported
portid:  0
trsvcid: 4420
subnqn:  nqn.1994-04.jp.co.hitachi:nvme:storage-subsystem-sn.8-10138-nvmssid.00001
traddr:  172.23.30.183
eflags:  none
sectype: none
```

```
# nvme discover --transport=tcp --traddr=172.23.30.184 --host-traddr=10.77.23.69 --
trsvcid=8009
Discovery Log Number of Records 12, Generation counter 5
=====Discovery Log Entry 0======
trtype:  tcp
adrfam:  ipv4
subtype: nvme subsystem
treq:    not specified, sq flow control disable supported
portid:  1
trsvcid: 4420
subnqn:  nqn.1994-04.jp.co.hitachi:nvme:storage-subsystem-sn.8-10138-nvmssid.00001
traddr:  172.23.30.184
eflags:  none
sectype: none
```

3. To discover the controllers, update the `/etc/nvme/discovery.conf` file by adding the parameters. You can include additional parameters for testing.

```
# cat /etc/nvme/discovery.conf
```

```
# Used for extracting default parameters for discovery
#
# Example:
# --transport=<trtype> --traddr=<traddr> --trsvcid=<trsvcid> --host-traddr=<host-traddr>
--host-iface=<host-iface>
--transport=tcp --traddr=172.23.30.183 --host-traddr=10.77.23.69 --trsvcid=8009 --ctrl-
loss-tmo=-1
--transport=tcp --traddr=172.23.30.184 --host-traddr=10.77.23.69 --trsvcid=8009 --ctrl-
loss-tmo=-1
```

Note: The additional parameter "--ctrl-loss-tmo=-1" is added to the discovery.conf file because in the event of a path failure, the NVM subsystem attempts to reconnect to the failed controller for a specified duration managed by the "ctrl-loss-tmo" option of the nvme connect command. The default value of ctrl-loss-tmo is 600 seconds, after which the failed path does not automatically reconnect. Setting the ctrl-loss-tmo option to -1 ensures that the NVM subsystem continuously attempts to reconnect indefinitely.

4. Connect to all the discovered controllers by running the following command:

```
#nvme connect-all
```

5. To ensure that all controllers are connected automatically during bootup, enable the `nvmf-autoconnect` service by running the following commands:

```
# systemctl enable nvmf-autoconnect.service
```

```
# systemctl start nvmf-autoconnect.service
```

6. List all the NVMe namespaces that are connected to the EC2 instance running RHEL:

```
# nvme list
Node              Generic               SN           Model
Namespace  Usage                       Format          FW Rev
------------- -------------- --------------------- -------------------------- -----------
--------- --------
/dev/nvme0n1   /dev/ng0n1     vol07125f4e52b92e280  Amazon Elastic Block Store   0x1
75.16  GB /  75.16  GB    512   B +  0 B   2.0
/dev/nvme2n1   /dev/ng2n1     8-10138-00001         HITACHI SVOS-RF-System       0x1
10.17  GB /  10.74  GB    512   B +  0 B   A3020140
```

Note: `/dev/nvme0n1` is the EBS volume of the EC2 instance where the RHEL 9.4 OS is installed.

7. List the available paths connected to the NVM subsystem:

```
# nvme list-subsys
nvme-subsys2 - NQN=nqn.1994-04.jp.co.hitachi:nvme:storage-subsystem-sn.8-10138-
nvmssid.00001
               hostnqn=nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-
7c2314fece0f
               iopolicy=numa
\
 +- nvme2 tcp
traddr=172.23.30.183,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
 +- nvme14 tcp
traddr=172.23.30.184,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
```

8. List the available paths connected to each device in the NVM subsystem by specifying the device node name:

```
#nvme list-subsys /dev/nvme2n1
nvme-subsys2 - NQN=nqn.1994-04.jp.co.hitachi:nvme:storage-subsystem-sn.8-10138-
nvmssid.00001
               hostnqn=nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-
7c2314fece0f
               iopolicy=numa
\
 +- nvme14 tcp
traddr=172.23.30.184,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
optimized
 +- nvme2 tcp
traddr=172.23.30.183,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
optimized
```

Configuring Multipathing

This section describes how to configure NVMe native multipathing for Hitachi VSP One Block 20 storage NVMe namespaces on RHEL 9.4 in an EC2 instance.

Configuring NVMe Native Multipath

NVMe multipathing is enabled by default in Red Hat Enterprise Linux 9. If it is not enabled, complete the following steps:

1. Enable NVMe multipathing by setting the 'nvme_core.multipath=Y' option for the 'nvme-core' module.

    a. Verify the current kernel and identify the 'vmlinuz' kernel file in the /boot directory.

    b. Run the grubby command:

    Syntax:

    ```
    grubby --args=nvme_core.multipath=Y --update-kernel /boot/<vmlinuz-file>
    ```

    Example:

    ```
    grubby --args=nvme_core.multipath=Y --update-kernel /boot/vmlinuz-5.14.0-
    427.20.1.el9_4.x86_64
    ```

2. Verify whether DM-Multipath is installed (in this scenario, DM-Multipath is not installed). If DM-Multipath is installed, disable and remove any `/etc/multipath.conf` files by running the following commands:

    ```
    # rpm -qa | grep multipath
    # systemctl disable multipathd
    # systemctl stop multipathd
    # systemctl status multipathd
    # multipath -F
    # rm /etc/multipath.conf
    ```

3. If NVMe multipathing is enabled manually, recreate the `'initramfs'` file and restart the system:

    ```
    # dracut -f -v
    # reboot
    ```

4. Verify that native NVMe multipath is enabled. If the output is 'Y', native NVMe multipath is enabled.

    ```
    # cat /sys/module/nvme_core/parameters/multipath
    Y
    ```

5. Create a udev rule for the VSP One Block 20 storage NVM subsystem to enable round-robin load balancing for NVMe native multipathing on RHEL 9.4. Create a `71-nvme-iopolicyhitachi-SVOS.rules` file under the `/lib/udev/rules.d` directory:

```
# cd /lib/udev/rules.d
# pwd
/lib/udev/rules.d

# cat /lib/udev/rules.d/71-nvme-iopolicy-hitachi-SVOS.rules
# Enable round-robin for hitachi SVOS ACTION=="add", SUBSYSTEM=="nvme-subsystem",
ATTR{model}=="HITACHI SVOS-RF-System", ATTR{iopolicy}="round-robin"
```

6. Reboot the system:

```
# reboot
```

7. Verify the NVM subsystem:

```
# nvme list-subsys
nvme-subsys2 - NQN=nqn.1994-04.jp.co.hitachi:nvme:storage-subsystem-sn.8-10138-
nvmssid.00001
               hostnqn=nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-
7c2314fece0f
               iopolicy=round-robin
\
 +- nvme2 tcp
traddr=172.23.30.183,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
 +- nvme14 tcp
traddr=172.23.30.184,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live

# nvme list-subsys /dev/nvme2n1
nvme-subsys2 - NQN=nqn.1994-04.jp.co.hitachi:nvme:storage-subsystem-sn.8-10138-
nvmssid.00001
               hostnqn=nqn.2014-08.org.nvmexpress:uuid:ec27080e-c341-a7ad-7171-
7c2314fece0f
               iopolicy=round-robin
\
 +- nvme14 tcp
traddr=172.23.30.184,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
optimized
 +- nvme2 tcp
traddr=172.23.30.183,trsvcid=4420,host_traddr=10.77.23.69,src_addr=10.77.23.69 live
optimized
```

8. Verify the effective IO policy for Hitachi VSP One Block 20 series storage and that the NVM subsystem is detected:

```
# cat /sys/class/nvme-subsystem/nvme-subsys*/iopolicy
round-robin

# cat /sys/class/nvme-subsystem/nvme-subsys*/model
HITACHI SVOS-RF-System
```

Configuring NVMe/TCP I/O Timeout on RHEL in AWS EC2

This section describes how to configure timeout value for I/O operations of the NVMe devices for AWS EC2 instance running RHEL 9.4.

Depending on the Linux version, the default timeout value might already be set at the maximum allowed value. For AWS EC2 instance with Red Hat Linux 9.4 version (ami-0c5ebd68eb61ff68d), the default I/O timeout for NVMe devices is set to '4294967295' seconds. Because of the high timeout value, I/O hang may occur for some storage hardware failure scenarios in NVMe/TCP configuration. For uninterrupted operation in the event of a storage hardware failure, you must configure the I/O timeout value to a low value (for example, '60' seconds).

The optimal I/O timeout value for NVMe core is '60' seconds according to VSP guidelines.

Configuring the I/O Operation Timeout Value

1. Verify the default I/O time out value for the RHEL host:

```
#cat /sys/module/nvme_core/parameters/io_timeout
4294967295
```

2. If the I/O timeout value is very high, change the I/O timeout value in the kernel line by setting the '`nvme_core.io_timeout=60`' option for the 'nvme-core' module.

3. After identifying the 'vmlinuz' kernel file in the /boot directory, run the following command:

   Syntax:

```
# grubby --args=nvme_core.io_timeout=<I/O timeout value> --update-kernel /boot/<vmlinuz-file>
```

   Example:

```
# grubby --args=nvme_core.io_timeout=60 --update-kernel /boot/ vmlinuz-5.14.0-427.20.1.el9_4.x86_64
```

   Note: This command affects both existing and future NVMe devices.

4. Reboot the system:

```
# reboot
```

5. Verify that the I/O time out value changed to the required value:

```
#cat /sys/module/nvme_core/parameters/io_timeout
60
```

Hitachi Vantara

Corporate Headquarters 2535 Augustine Drive
Santa Clara, CA 95054 USA www.HitachiVantara.com community.HitachiVantara.com

Regional Contact Information
Americas: +1 866 374 5822 or info@hitachivantara.com
Europe, Middle East and Africa: +44 (0) 1753 618000 or info.emea@hitachivantara.com
Asia Pacific: +852 3189 7900 or info.marketing.apac@hitachivantara.com