Technical Report

NetApp E-Series and NVMe over Fabrics Support

Nonvolatile Memory Express over InfiniBand and RoCE on E-Series Systems

Abdel Sadek, Chuck Nichols, Joey Parnell, Scott Terrill, Steve Schremmer, Amine Bennani, Tish Best | NetApp
March 2019 | TR-4766

Abstract

The new NetApp® E-Series EF570 and E5700 storage systems with SANtricity® 11.50 support the new Nonvolatile Memory over Fabrics (NVMe-oF) protocol using either InfiniBand or RDMA over Converged Ethernet (RoCE) connections. This report provides technical details for the implementation, benefits, and limitations of this system. It also compares SCSI and NVMe-oF in structure and performance. The comparison involves NVMe/IB versus using iSCSI Extensions for RDMA (iSER) over InfiniBand, NVMe/IB versus Fibre Channel, and NVMe/RoCE versus iSCSI.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

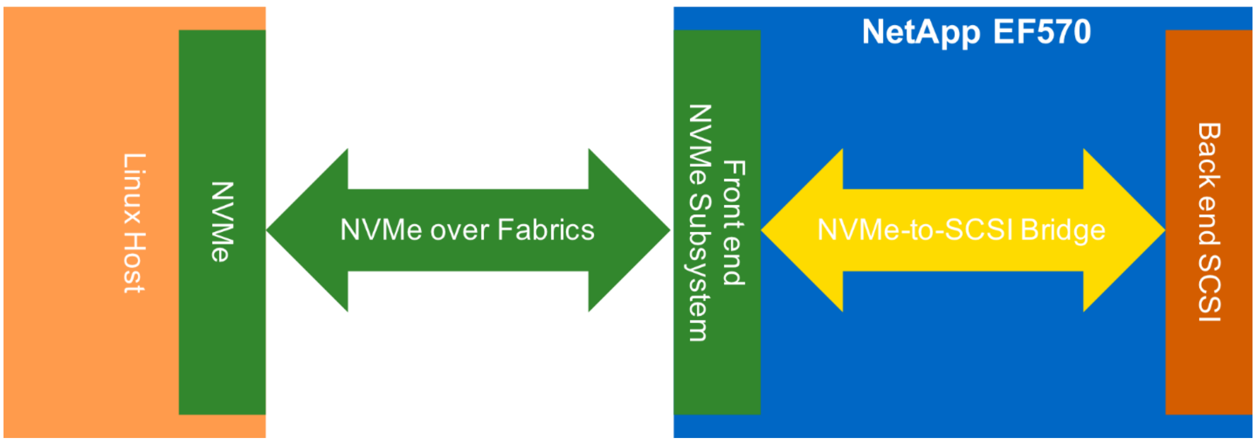
1 Introduction

Nonvolatile Memory Express (NVMe) has become the industry standard interface for PCIe solid-state disks (SSDs). With a streamlined protocol and command set and fewer clock cycles per I/O, NVMe supports up to 64K queues and up to 64K commands per queue. These attributes make it more efficient than SCSI-based protocols like SAS and SATA.

The introduction of NVMe over Fabrics (NVMe-oF) makes NVMe more scalable without affecting the low latency and small overhead that are characteristic of the interface.

Most implementations on the market today focus on adding NVMe-enabled drives to the back-end storage while keeping the front end to the host SCSI based. NetApp E-Series storage systems have taken a different approach. With the E-Series, NVMe-oF is supported from the host to the front end of the NetApp EF570 all-flash array or the E5700 hybrid array, while the back end is still SCSI-based with SAS drives.

Figure 1) NVMe-oF front end on E-Series systems.



2 NVMe-oF Ecosystem

NetApp is a promoter of the nvmexpress.org committee, which manages the NVMe and NVMe-oF specifications. In addition, NetApp actively collaborates with different partners at the engineering level, including operating system vendors like Red Hat and SuSE, along with hardware vendors like Mellanox. The E-Series Interoperability team is also working with the University of New Hampshire for potential certification at NVMe Plugfest events.

3 Transport Layers: InfiniBand and RoCE

The NVMexpress.org specifications outline support for NVMe-oF over remote direct memory access (RDMA) and Fibre Channel (FC). The RDMA-based protocols can be either InfiniBand (IB) or RDMA over Converged Ethernet (RoCE).

The NetApp E-Series implementation supports both NVMe/IB and NVMe/RoCE for the following reasons:

- InfiniBand and RoCE have RDMA built into them.
- The E-Series already has a long history of supporting other protocols over RDMA (SCSI-based), like iSCSI Extensions for RDMA (iSER) and SCSI RDMA Protocol (SRP).
- The same hardware on EF570 and E5700 that runs iSER or SRP can run NVMe/IB or NVMe/RoCE, although not at the same time.

- All three protocols (iSER, SRP, and NVMe/IB) can coexist on the same fabric and even on the same InfiniBand host channel adapter (HCA) port on the host side. This capability allows customers with existing fabrics running iSER and/or SRP to connect the EF570 or E5700 running NVMe/IB to the same fabric.
- Both iSCSI and NVMe/RoCE can coexist on the same fabric on the host side.
- All InfiniBand components in the fabric (NetApp EF570 and E5700 storage systems, switches, and HCAs) can negotiate the speed down as needed: Enhanced Data Rate (EDR) 100Gbps; Fourteen Data Rate (FDR) 56Gbps; or Quad Data Rate (QDR) 40Gbps. A lower speed makes it easier to connect to legacy components.
- The NetApp EF570 and E5700 support 100Gbps, 50Gbps, 40Gbps, 25Gbps, and 10Gbps speeds for NVMe/RoCE.
- The NetApp EF570 and E5700 support NVMe/RoCE v2 (which is routable), and they are also backward compatible with RoCE v1.

4 NVMe and NVMe-oF: Protocols and Concepts

4.1 Command Protocol Compared with the Transport Protocol

This document uses the following definitions:

- **Command protocol.** The command set used for moving data to and from the storage array and managing storage entities on that array.
- **Transport protocol.** The underlying physical connection and the protocol used to carry the command protocol's commands to and from the storage array.

Since its inception, E-Series storage software was designed to provide SCSI command protocol storage functionality. Over the years, this software has been extended, architecturally and functionally, to support other transport protocols. However, in all cases, it has remained fundamentally a SCSI product. Therefore, it has historically been unnecessary to distinguish between the concept of the supported command protocol and the concept of the underlying transport protocol. That is, because E-Series has traditionally been a SCSI product, the fact that it supports a Fibre Channel host/fabric connect implies that it supports the FCP SCSI protocol carried by the Fibre Channel transport protocol.

Table 1 lists the transport and command protocol combinations historically supported by E-Series storage products. Note that, despite the variations in underlying transport mechanisms, SCSI has been the command protocol in use in every case.

Table 1) Historical E-Series transport and command protocol combinations.

Transport Protocol	Command Protocol
Parallel SCSI bus	SCSI
Fibre Channel	Fibre Channel SCSI Protocol (FCP)
Serial-attached SCSI	Serial SCSI Protocol (SSP)
Ethernet	Internet SCSI Protocol (iSCSI)
InfiniBand	SCSI RDMA Protocol (SRP)
InfiniBand	iSCSI extensions for RDMA (iSER)
InfiniBand	NVMe over Fabrics (NVMe/IB)
RDMA over Converged Ethernet	NVMe over Fabrics (NVMe/RoCE)

The E-Series storage software now supports a command protocol other than SCSI: NVMe over Fabrics. The 11.40 release of the E-Series software supports the InfiniBand NVMe transport protocol, and the 11.50 release also supports the RoCE v2 NVMe transport protocol.

4.2 NVMe Concepts

NVMe

NVMe is a specification-defined, register-level interface for applications (through OS-supplied file systems and drivers) to communicate with nonvolatile memory data storage through a PCI Express (PCIe) connection. This interface is used when the storage devices reside in the same physical enclosure, and the host OS and application can be directly connected through PCIe, such as within servers or laptop computers.

NVMe-oF

NVMe-oF is a specification-defined extension to NVMe that enables NVMe-based communication over interconnects other than PCIe. This interface makes it possible to connect “external” storage enclosures to a server, either directly or through a switch, while still using NVMe as the fundamental communication mechanism.

The NVMe-oF protocol can be carried over multiple different physical interconnects, including transports such as RoCE, InfiniBand, and Fibre Channel. Regardless of the interconnect, NVMe-oF is transport agnostic. Server and storage communicate through NVMe-oF, independent of the underlying transport that is used to carry the NVMe-oF protocol. For the 11.50 release of E-Series software, the InfiniBand and RoCE transport protocols are both supported.

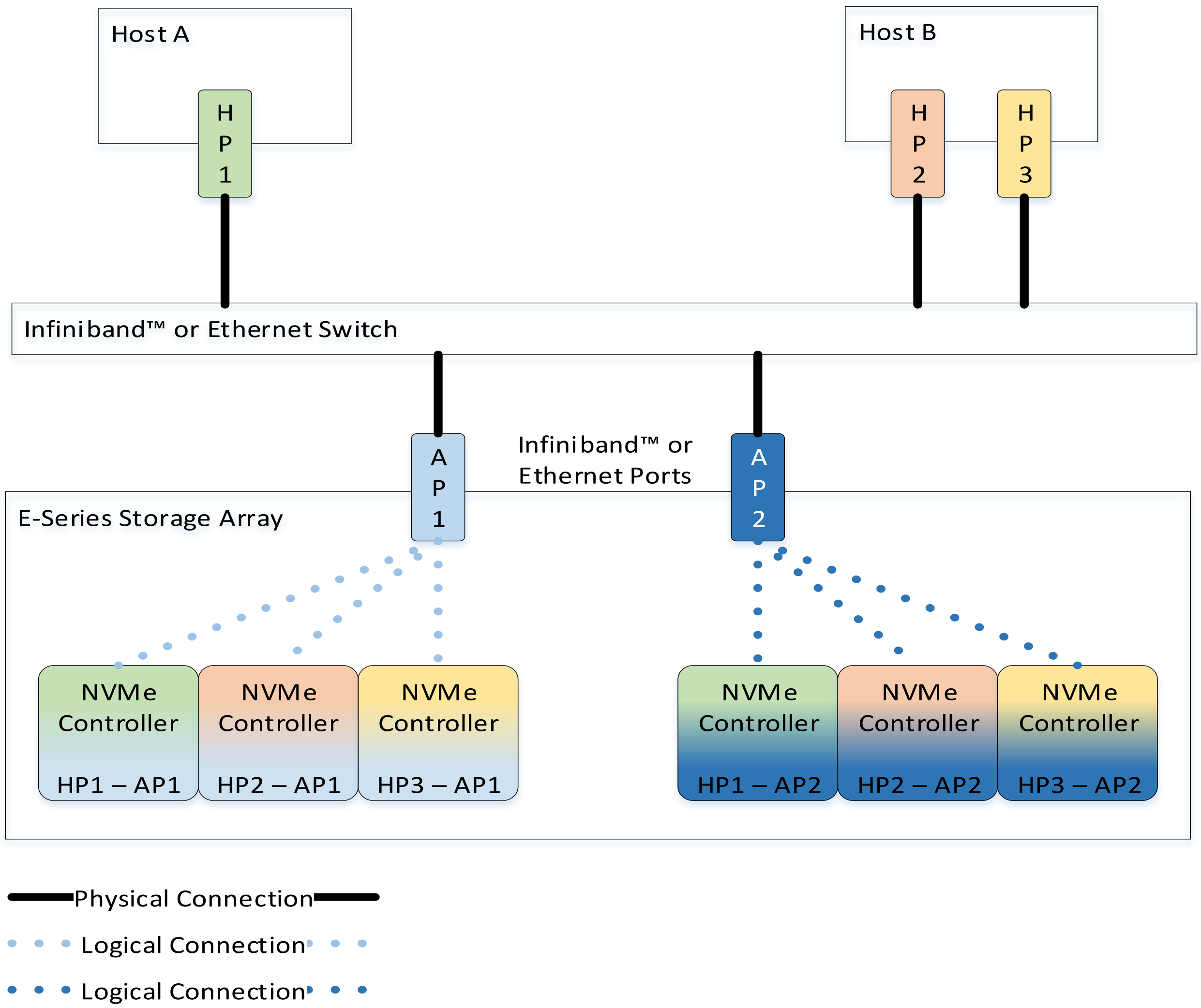
Controller

The term controller has a very specific meaning in the context of NVMe. An NVMe controller is a concrete entity defined by the specification as “a PCI Express function that implements NVM Express.” Essentially, a controller is the device that presents a set of hardware registers that can be accessed with PCIe for the purpose of storing nonvolatile data.

The NVMe-oF standard extends the definition of a controller, however, specifying that “a controller is associated with exactly one host at a time.” Thus, in the context of NVMe-oF, a controller becomes an abstract entity that represents the relationship between a single host port and a single storage array port. In SCSI terms, this is analogous to the concept of an initiator port-target port nexus. When an NVMe-oF connection is established between an initiator port and a target port, an NVMe controller is created to represent that connection.

Figure 2 depicts the logical NVMe controller constructs for a typical NVMe-oF topology. Host A presents one InfiniBand port (HP1) connected to a switch, and Host B presents two ports (HP2 and HP3), also connected to the same switch. The E-Series controller presents two InfiniBand ports, AP1 and AP2, both connected to the same switch as the host ports. NVMe-oF connections are established from each host port to each array port. Therefore E-Series software creates a logical NVMe controller for each connection. Thus, the figure depicts six NVMe controllers for the sample configuration, with the host-to-array port association described as HPx-APx, or HP3-AP2 in this case.

Figure 2) NVMe-oF host/array topology with logical NVMe controllers.



Namespace

An NVMe namespace is defined as "a quantity of nonvolatile memory that can be formatted into logical blocks." This definition is virtually identical to the SCSI concept of a logical unit. Thus, volumes created on an E-Series storage array that have historically been presented to SCSI interfaces as logical units are presented to NVMe-oF interfaces as namespaces.

Namespace ID

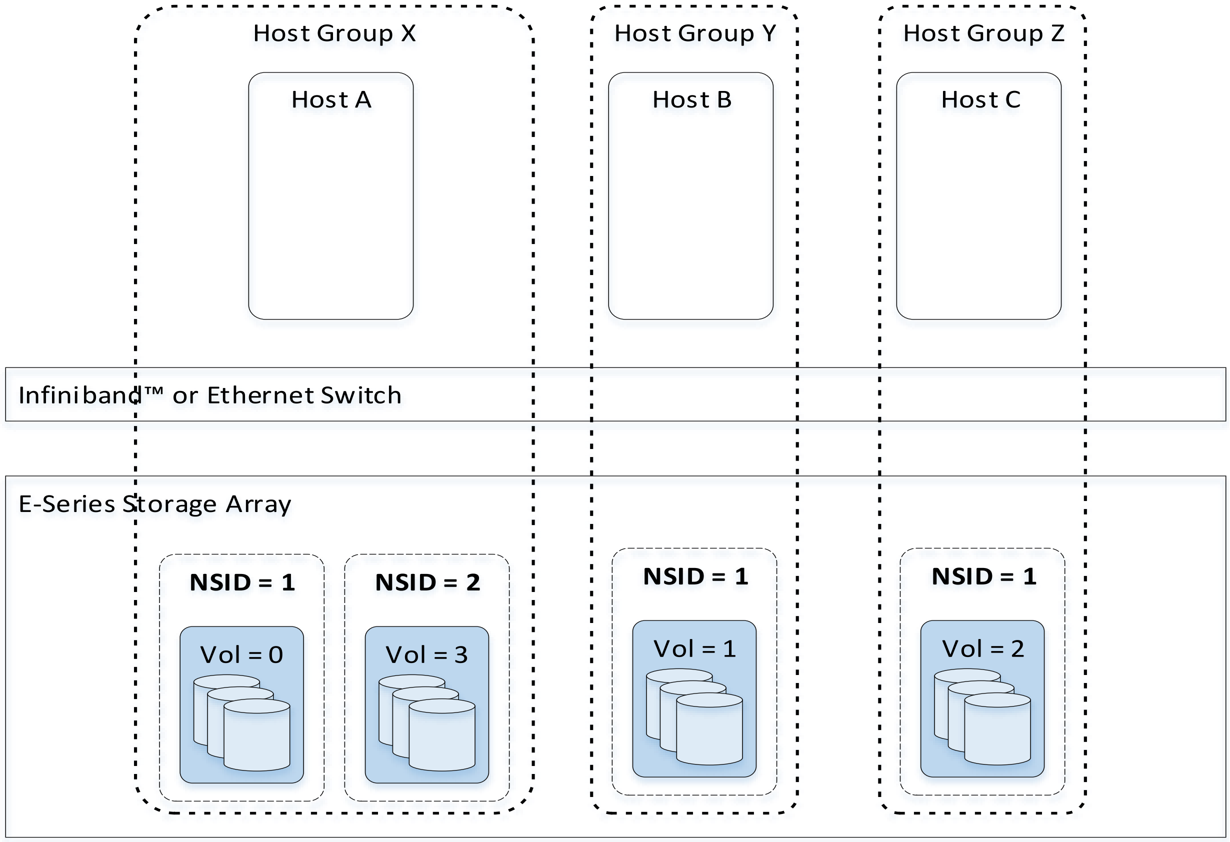
A namespace ID (NSID) is an identifier used by a controller to provide access to a namespace. This is nearly equivalent to a logical unit number (LUN) in SCSI. There are two exceptions. First, an NSID cannot have a value of 0. Second, an NSID of FFFFFFFFh is a broadcast value that is used to specify all namespaces.

The accessibility of a volume by a host is configured from the management interfaces, along with setting the namespace ID for that host or host group. As with SCSI, a logical volume can be mapped to only a single host group at a time, and a given host group cannot have any duplicate NSIDs.

Figure 3 shows the logical partitioning of storage to multiple hosts. Logical Volumes 0 and 3 are assigned to Host Group X with NSIDs of 1 and 2, respectively (recalling that 0 is not a valid NSID). Logical Volume

1 is assigned to Host Group Y as NSID 1, and Logical Volume 2 is assigned to Host Group Z, also as NSID 1.

Figure 3) Namespace ID mapping to host groups.



Queue Pairs

Like some other RDMA-based protocols, NVMe and NVMe-oF communication between devices relies on the concept of a queue pair (QP). A QP is the combination of a submission queue (SQ) and a completion queue (CQ). The host (initiator) places commands into an SQ that is read by the storage array (target). The target places completion information relating to a received command into the CQ associated with the SQ on which the command was received. For NVMe-oF, there must be a 1:1 relationship between submission queues and completion queues; that is, every SQ must have a single, unique CQ associated with only that SQ.

When an NVMe controller is created, a minimum of two QPs are created. The first QP (using queue ID 0) is referred to as the admin queue; all remaining queues are referred to as I/O queues. The admin queue is used to process the admin command set (see “Admin Commands” in section 4.3, “Command Sets”), while the I/O queues are used to process the NVM command set (see “NVM Commands” in section 4.3).

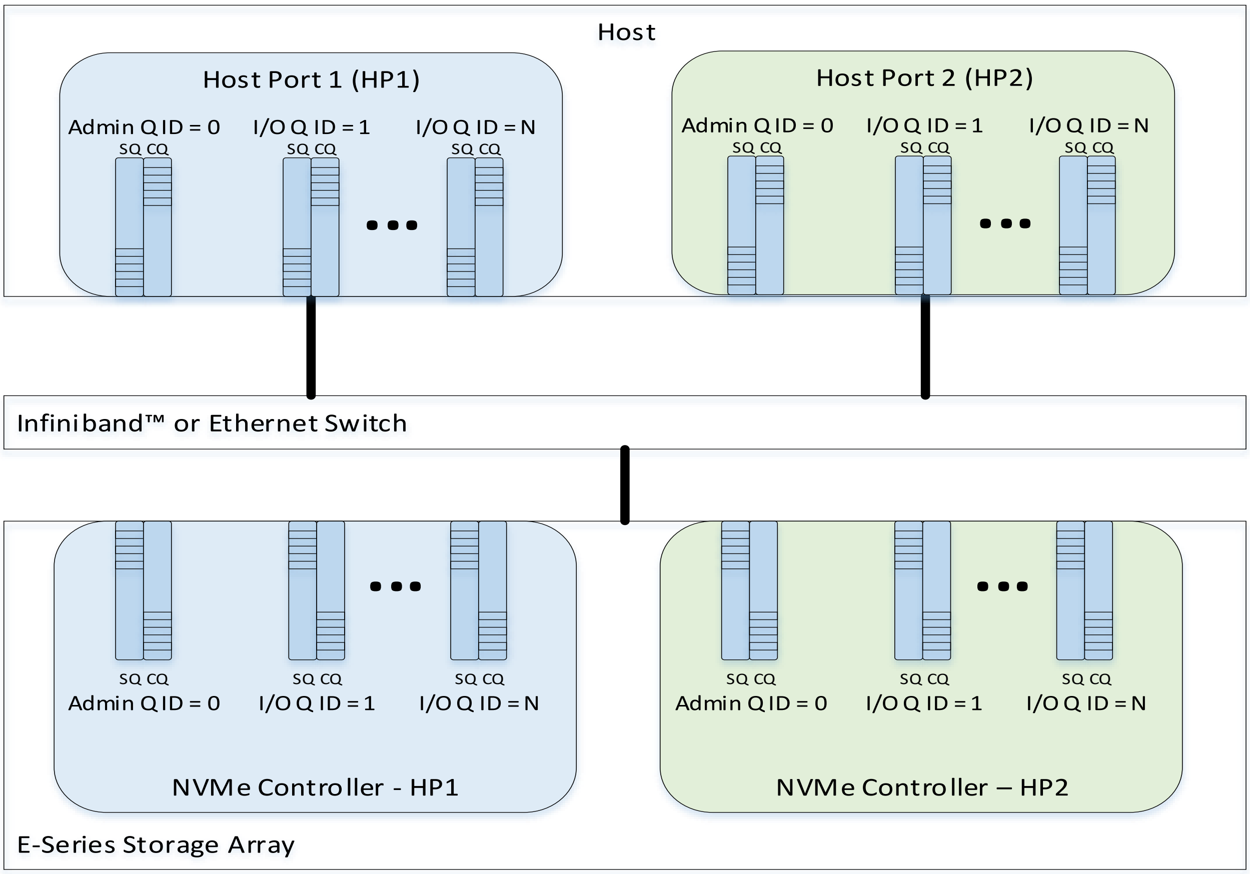
The number of I/O queues created by the controller is negotiated by the initiator and target:

1. The initiator requests some number of I/O queues.
2. The target responds with the number of I/O queues that can be created on that controller.
3. The initiator issues I/O queue creation requests for the number of queues indicated by the target.

The 11.50 release of the E-Series SANtricity OS software supports four I/O queues per controller. This number was increased from one in the 11.40 release.

Figure 4 illustrates the queue pair concept applied to NVMe controllers. Recall that the NVMe controller is a logical construct within the E-Series storage software. It represents the combination of the host port and the array port that forms the NVMe-oF connection between the host and the storage array. When an NVMe-oF connection is established, the host creates an admin queue (with a Queue ID of 0) and then negotiates with the array to create N I/O queues (with Queue IDs of 1 through N). Even though the host queue pairs and the storage array queue pairs are not part of the same physical memory system, NVMe-oF allows them to be a logically "shared" memory queue pair, with command and control structures being shared through RDMA between the host and storage array. Thus, when the host puts a new command in a slot on its submission queue for I/O Queue 1, that command is placed, through RDMA, in the same slot in I/O Queue 1 for the associated NVMe controller on the storage array.

Figure 4) NVMe controller queue pairs.



4.3 Command Sets

This section is a brief overview of the NVMe and NVMe-oF commands supported by the E-Series storage array software. For details of specific commands, refer to the NVMe and NVMe-oF specifications referenced at the end of this document, as well as the E-Series NVMe host interface software interface specification.

Admin Commands

Admin commands are received and completed through the controller admin QP. These commands are used, among other things, to configure and monitor the status of NVMe controllers connected to a host. They are analogous to SCSI non-read/write commands such as SCSI INQUIRY, MODE SELECT/SENSE, and LOG SELECT/SENSE, among others.

Table 2) E-Series supported admin commands.

Command	Specification	Mandatory or Optional
Get Log Page	NVMe	M
Identify	NVMe	M
Abort	NVMe	M
Set Features	NVMe	M
Get Features	NVMe	M
Asynchronous Event Request	NVMe	M
Keep Alive	NVMe	O

NVM Commands

NVM commands are received and completed through the controller I/O QP. These commands are used for operations such as data movement and namespace access and are analogous to SCSI commands such as READ, WRITE, SYNC CACHE, PERSISTENT RESERVE IN, and so on.

Table 3) E-Series supported NVM commands.

Command	Specification	Mandatory or Optional
Flush	NVMe	M
Write	NVMe	M
Read	NVMe	M
Reservation Register	NVMe	O
Reservation Report	NVMe	O
Reservation Acquire	NVMe	O
Reservation Release	NVMe	O

As noted in Table 3, in addition to the mandatory NVM commands, E-Series arrays using the NVMe-oF host interface support the optional reservation feature. This feature allows a host to reserve a namespace, which prevents other hosts from writing to the namespace (reads might also be restricted). This feature is commonly used by clustering packages to make sure that nodes within the cluster do not accidentally access the same namespace at the same time.

Those familiar with SCSI persistent reservations might find this feature to be very similar, although there are a number of differences. In SCSI, the registrant is tied to one and only one initiator-target-LUN (I_T_L) nexus, whereas in NVMe-oF, the registrant is tied to a namespace and host ID. Multiple controllers can have the same host ID, and all controllers using the same host ID are treated the same.

Another difference is that the “persist through power loss” setting can be set outside of a register command, and a register has the option of specifying “no change” in addition to on and off.

NVMe-oF does not have a “specify initiator” or “all target port” concept. Using the same host ID for all connections from a given host essentially solves the same problem. It is worth noting that NVMe-oF standard 1.2 has an “ignore existing key” parameter for every reservation command, but we have chosen to take guidance from 1.3, which has dropped this parameter for all but the register command. The actual format of the commands and parameter data is noticeably different from SCSI. We recommend looking at the NVM Express standard or the NVM Express SCSI translation guide to understand how to send valid reservation commands to the array.

Fabrics Commands

Fabrics commands (see Table 4) are used to create queue pairs and initialize NVMe controllers. They are somewhat analogous to protocol-specific initialization commands such as the Fibre Channel port login.

Table 4) E-Series supported fabrics commands.

Command	Specification	Mandatory/Optional
Property Set	NVMe-oF	Mandatory
Connect	NVMe-oF	Mandatory
Property Get	NVMe-oF	Mandatory

4.4 Host Driver and Tools

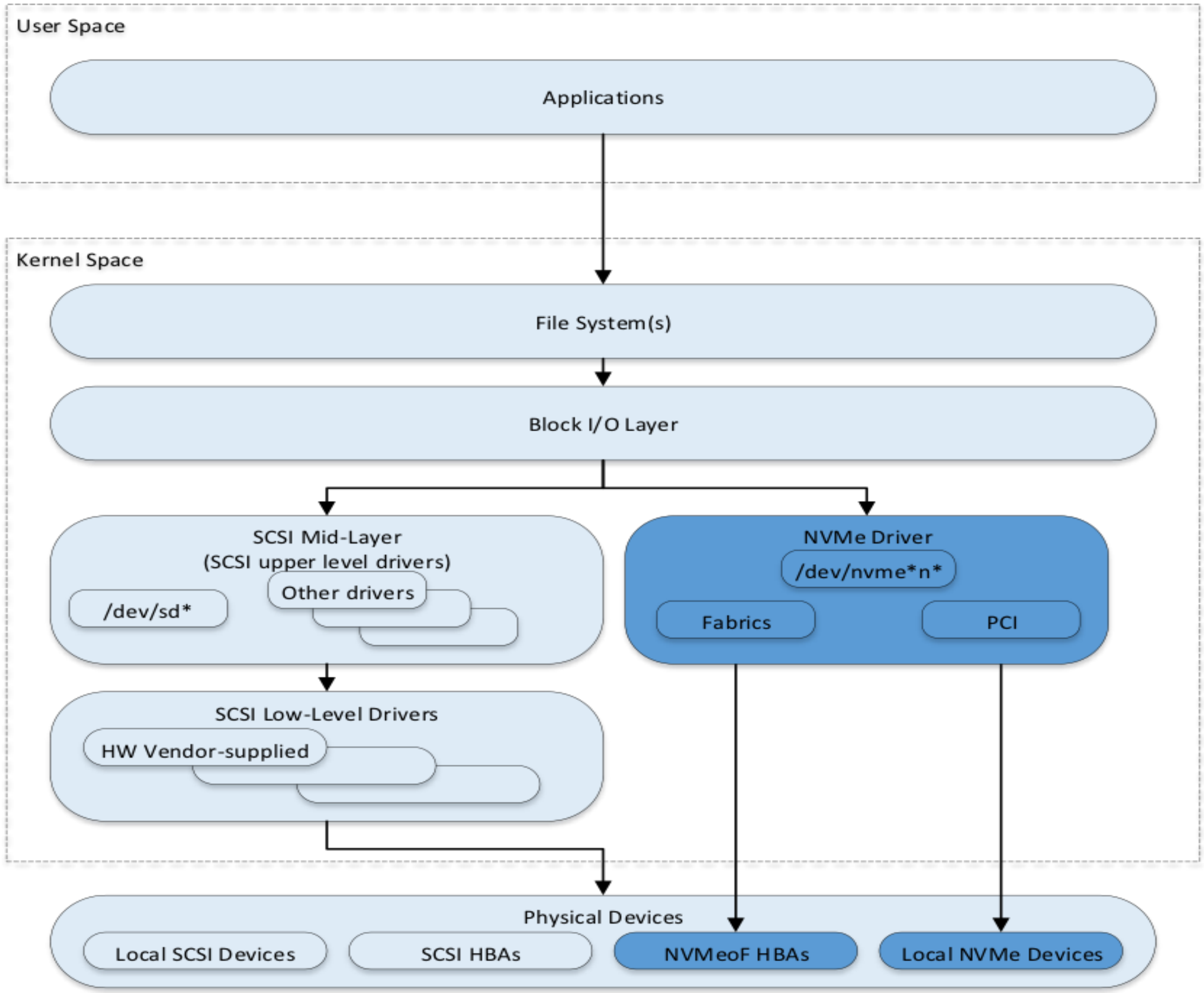
Driver Stack

One of the advertised advantages of NVMe (and NVMe-oF) relative to SCSI is that it can support lower-latency I/O. This is not only because the devices are faster, but also because of some inherent advantages in the host OS driver stack. Therefore I/O spends less time getting from the application to the storage, which reduces response times.

Figure 5 shows a simplified view of the Linux OS driver stack. Like the SCSI driver, the NVMe driver sits below the block I/O layer. The NVMe driver, however, is not split into upper and lower levels. The driver presents the NVMe devices to the block I/O system and contains drivers to support both PCIe-attached devices and fabrics-attached devices. The fabrics portion of the driver contains the transport-specific code to handle operations such as InfiniBand-based RDMA.

Another advantage of using an NVMe driver is that they are designed with fast, nonvolatile memory storage devices in mind; that is, they are optimized to work well with low-latency storage. The SCSI driver, on the other hand, was originally created and used with rotating media-based storage such as spinning disk drives. For those devices, the bulk of the I/O response time lay with the device itself (for example, rotational seek times). Therefore, the driver itself did not necessarily have to be highly optimized for execution-time performance. Additionally, the SCSI driver stack was designed to support multiple different types of SCSI devices such as disks, tapes, printers, and so on. Therefore, there is some built-in overhead in the driver for handling I/O to various types of devices that is unnecessary and not present in the NVMe driver.

Figure 5) Linux OS driver structure.



Multipathing and Failover

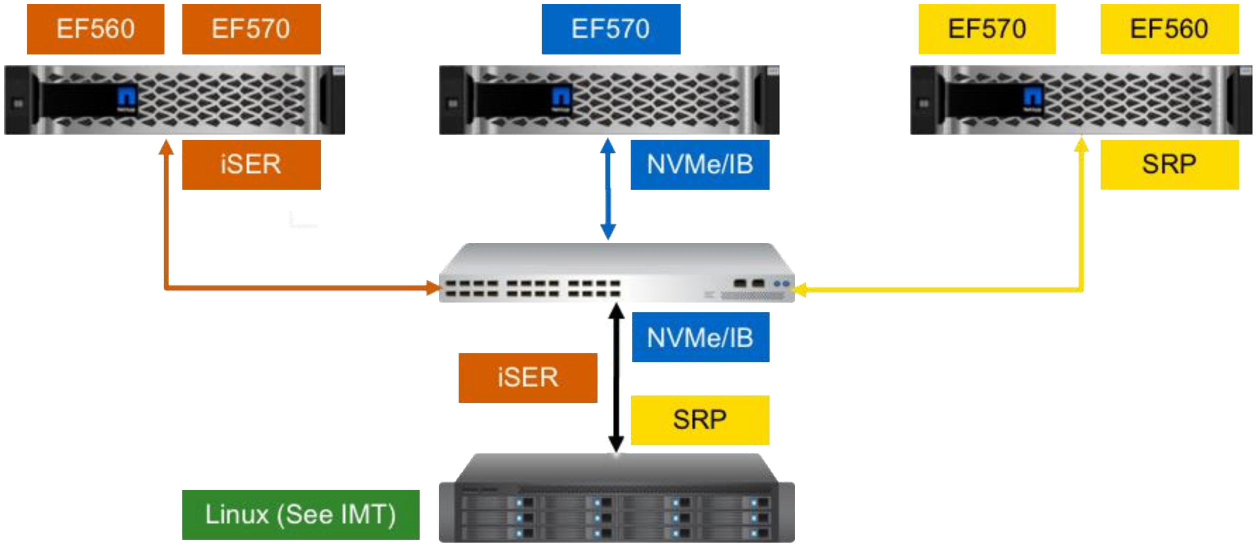
Currently, multipathing and failover functionality is provided by the Linux Device Mapper Multipath (DM-MP) module. Multiple paths to the namespaces on an E-Series array are automatically configured into a single device. NetApp provides an NVMe prioritizer package that works with DM-MP to ensure that I/O is sent down the optimal path to the E-Series namespaces and to assist with failover and failback operations. The NetApp prioritizer uses a vendor-unique method to determine the path information, because the NVMe specification does not provide this capability.

In the near future, the path information will be provided using the new NVMe Asynchronous Namespace Access (ANA) feature. NetApp worked with the NVMe community to define this functionality, and the ANA specification was recently released by the NVMe organization. NetApp is working with the Linux community to finalize the ANA implementation in the OS. E-Series arrays have already incorporated support for the ANA standard, and this support can be enabled once host-side implementation is available.

Coexistence Among NVMe/IB, iSER, and SRP

On the host side, all the protocols that E-Series devices support on InfiniBand (SRP, iSER, and NVMe/IB) can run on the same host channel adapter at the same time but to different E-Series targets (Figure 6).

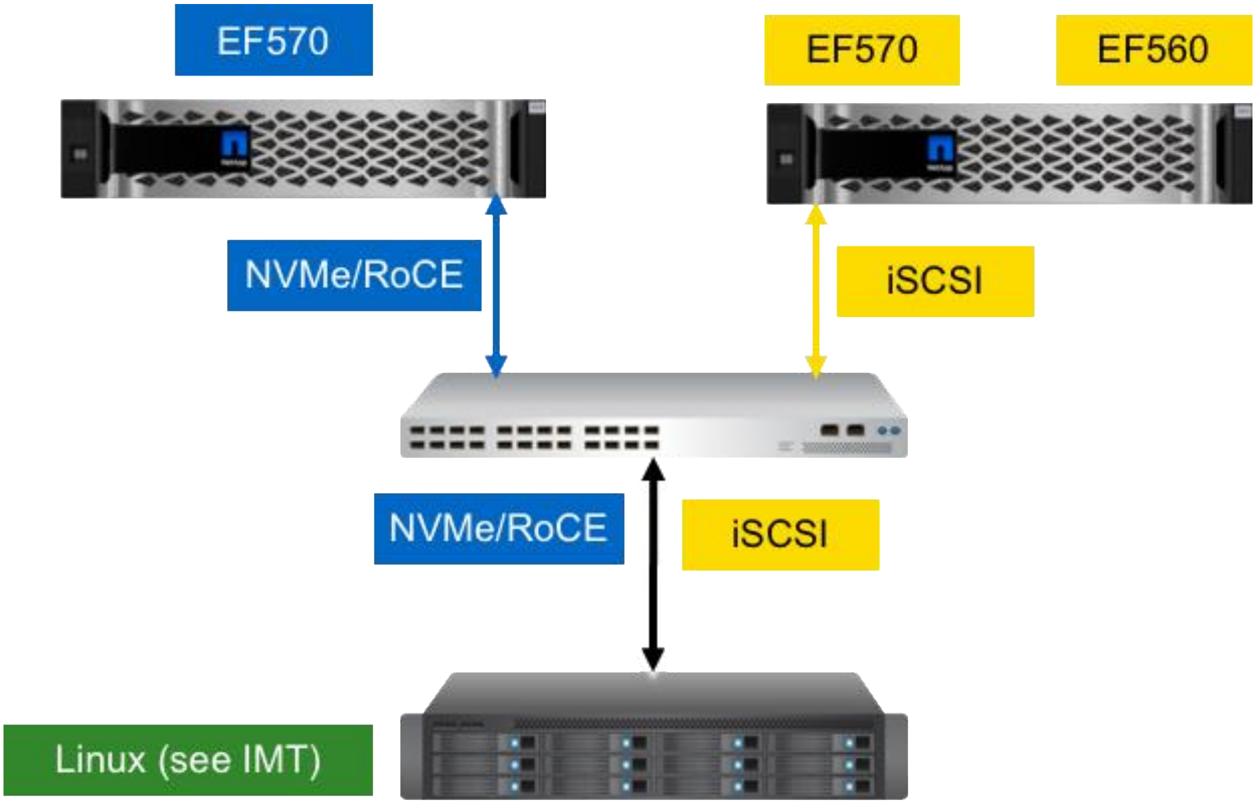
Figure 6) Coexistence among NVMe/IB, iSER, and SRP on the host side.



Coexistence Between NVMe/RoCE and iSCSI

On the host side, iSCSI and NVMe/RoCE can run on the same network adapter, provided that it supports RDMA at the same time, but to different E-Series targets (Figure 7).

Figure 7) Coexistence between NVMe/RoCE and iSCSI.



NVMe CLI

The NVMe CLI is a command line utility that provides management tools for NVMe devices on a Linux host. The utility can list NVMe namespaces that are configured as block devices on the host and can provide details about those namespaces. For NVMe-oF, it has subcommands for discovering and connecting to NVMe controllers on the fabric. Table 5 lists some useful commands. For more information, see the Linux man pages by running the `man nvme` command from the Linux shell.

Table 5) Some useful NVMe CLI commands.

Command	Description
nvme list	Lists information about each NVMe namespace configured as a block device on the system
nvme discover	Discovers NVMe controllers on the fabric
nvme connect	Connects to an NVMe controller on the fabric
nvme connect-all	Discovers and connects to multiple NVMe controllers on the fabric by using settings contained in `/etc/nvme/discovery.conf`

4.5 Release Limitations

Table 6) Limitations.

Features/Limitations	Host	E-Series	In the NVMe-oF Specification
Multipathing/failover (DM-MP)	Yes*	Yes	No
Multipathing/failover (NVMe with ANA)	No*	Yes	Yes
Nondisruptive firmware upgrade	Yes	Yes	N/A
Persistent reservations	Yes	Yes	Yes
T10PI	No	Yes	Yes
Remote mirroring	N/A	No	No
Copy services	N/A	No	No
Thin provisioning	Yes	No	Yes

* Refer to the Interoperability Matrix Tool (IMT) for host operating system support.

4.6 FAQ

Are regular network interface cards (NICs) supported for NVMe/RoCE?

Answer: No. The cards must support RDMA. They are called rNICs and are widely available.

Is direct connect supported, or is a switch required?

Answer: Both direct connect and fabric connection through a switch are supported for NVMe/IB and NVMe/RoCE.

Do E-Series controllers support direct connect to multiple hosts using a 4x25GbE by 100GbE cable for NVMe/RoCE?

Answer: No.

Q: What are the different form factors for different speeds?

Answer: NVMe/IB: QSFP28 for 100Gbps, 56Gbps, and 40Gbps.

Answer: NVMe/RoCE: QSFP28 for 100Gbps and 50Gbps; QSFP+ for 40Gbps; SFP28 for 25Gbps; and SFP+ for 10Gbps.

Q: What protocols do the EF570 and E5700 support with the 100Gbps InfiniBand host interface card?

Answer: The SRP, iSER, NVMe/IB, and NVMe/RoCE protocols are supported, but not at the same time. Migrating between protocols is software based, and no hardware change is required.

Q: Do E-Series systems support regular RoCE, which runs SCSI?

Answer: No. Only the NVMe protocol is supported to run over RoCE, not SCSI.

5 Conclusion

NVMe-oF is still evolving, and NetApp is leading the way. With the EF570/E5700 release, NetApp is the first to market with NVMe over Fabrics enterprise storage. The efficient SANtricity OS with back-end SCSI keeps cost low and performance high, with very low latency.

References

The following references were used in this technical report:

- NVMe over Fabrics Specifications
  http://www.nvmexpress.org/resources/specifications/
- Linux Storage Stack
  https://www.thomas-krenn.com/en/wiki/Linux_Storage_Stack_Diagram
- NetApp Interoperability Matrix (IMT)
  https://mysupport.netapp.com/matrix/#welcome
- E-Series NVMe-oF Express Guide
  https://docs.netapp.com/ess-11/index.jsp

Version History

Version	Date	Document Version History
Version 1.0	March 2019	Initial draft.

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

TR-4766-0319

NetApp