

Network File System Version 4  
Internet-Draft January 1, 2020  
Intended status: Informational  
Expires: January 6, 2021

S. Faibish  
D. Black  
Dell EMC  
C. Hellwig  
July 6, 2020

Using the Parallel NFS (pNFS) SCSI/NVMe Layout  
draft-faibish-nfsv4-scsi-nvme-layout-00

## Abstract

This document explains how to use the Parallel Network File System (pNFS) SCSI NVMe Layout Types with transports using the NVMe over Fabrics protocols. This draft picks the previous SCSI over NVMe draft of C. Hellwig and extended it to support all the types of transport protocols supported by NVMe transport over fabrics additional to the SCSI transport protocol introduced in pNFS SCSI Layout. The proposed transport protocols include support for several transports and fabrics and support the RDMA transports.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of Internet-Draft Shadow Directories can be accessed at <https://www.ietf.org/standards/ids/internet-draft-mirror-sites/>.

This Internet-Draft will expire on January 6, 2021.

## Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](https://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">1.1.</a>	Conventions Used in This Document . . . . .	<a href="#">2</a>
<a href="#">1.2.</a>	General Definitions . . . . .	<a href="#">2</a>
<a href="#">2.</a>	SCSI Layout mapping to NVMe . . . . .	<a href="#">3</a>
<a href="#">2.1.</a>	Volume Identification . . . . .	<a href="#">7</a>
<a href="#">2.2.</a>	Client Fencing . . . . .	<a href="#">7</a>
<a href="#">2.2.1.</a>	Reservation Key Generation . . . . .	<a href="#">8</a>
<a href="#">2.2.2.</a>	MDS Registration and Reservation . . . . .	<a href="#">8</a>
<a href="#">2.2.3.</a>	Client Registration . . . . .	<a href="#">8</a>
<a href="#">2.2.4.</a>	Fencing Action . . . . .	<a href="#">8</a>
<a href="#">2.2.5.</a>	Client Recovery after a Fence Action . . . . .	<a href="#">9</a>
<a href="#">2.3.</a>	Volatile write caches . . . . .	<a href="#">10</a>
<a href="#">3.</a>	Security Considerations . . . . .	<a href="#">10</a>
<a href="#">4.</a>	IANA Considerations . . . . .	<a href="#">10</a>
<a href="#">5.</a>	Normative References . . . . .	<a href="#">11</a>
	Author's Address . . . . .	<a href="#">11</a>

## [1.](#) Introduction

The pNFS Small Computer System Interface (SCSI) layout [[RFC8154](#)] is layout type that allows NFS clients to directly perform I/O to block storage devices while bypassing the MDS. It is specified by using concepts from the SCSI protocol family for the data path to the storage devices. This documents explains how to access PCI Express, RDMA or Fibre Channel devices using the NVM Express protocol [[NVME](#)] using the SCSI layout. This document does not amend the pNFS SCSI layout document in any way, instead of explains how to map the SCSI constructs used in the pNFS SCSI layout document to NVMe concepts

using the NVMe SCSI translation reference.

### [1.1.](#) Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

Faibish

Expires January 6, 2021

[Page 2]

---

Internet-Draft

pNFS SCSI/NVMe Layout over Fabrics

July 2020

### [1.2.](#) General Definitions

The following definitions are provided for the purpose of providing an appropriate context for the reader.

**Client** The "client" is the entity that accesses the NFS server's resources. The client may be an application that contains the logic to access the NFS server directly. The client may also be the traditional operating system client that provides remote file system services for a set of applications.

**Server/Controller** The "server" is the entity responsible for coordinating client access to a set of file systems and is identified by a server owner.

## [2.](#) SCSI Layout mapping to NVMe

The SCSI layout definition [[RFC8154](#)] only references few SCSI specific concepts directly.

**NVM Express** [[NVME](#)] Base Specification revision 1.4 and prior revisions define a register level interface for host software to communicate with a non-volatile memory subsystem over PCI Express (NVMe over PCIe). This specification defines extensions to NVMe that enable operation over other interconnects (NVMe over Fabrics). The NVM Express Base Specification revision 1.4 is referred to as the NVMe Base specification.

The goal for this draft is to enable an implementer who is familiar with the pNFS SCSI layout ([RFC 8154](#)) and the NVMe standards (both NVMe-oF 1.1 and NVMe 1.4) to implement the pNFS SCSI layout over NVMe-oF. The mapping of extensions defined in this document refers to a specific NVMe Transport defined in an NVMe Transport binding specification. This document refers to

NVMe Transport binding specification for FC, RDMA and TCP [[RFC7525](#)]. The NVMe Transport binding specification for Fibre Channel is defined in INCITS 540 Fibre Channel - Non-Volatile Memory Express [[FC-NVMe](#)].

NVMe over Fabrics has the following differences from the NVMe Base specification used with SCSI:

- There is a one-to-one mapping between I/O Submission Queues and I/O Completion Queues. NVMe over Fabrics does not support multiple I/O Submission Queues being mapped to a single I/O Completion Queue;

Faibish

Expires January 6, 2021

[Page 3]

---

Internet-Draft

pNFS SCSI/NVMe Layout over Fabrics

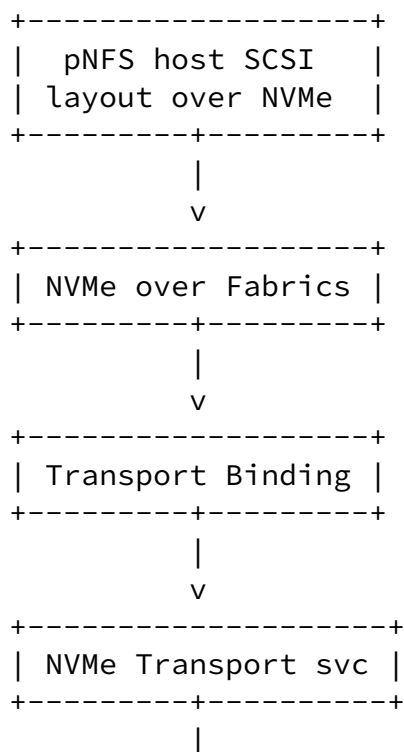
July 2020

- NVMe over Fabrics does not define an interrupt mechanism that allows a controller to generate a host interrupt. It is the responsibility of the host fabric interface (e.g., Host Bus Adapter) to generate host interrupts;
- NVMe over Fabrics does not use the Create I/O Completion Queue, Create I/O Submission Queue, Delete I/O Completion Queue, and Delete I/O Submission Queue commands. NVMe over Fabrics does not use the Admin Submission Queue Base Address (ASQ), Admin Completion Queue Base Address (ACQ), and Admin Queue Attributes (AQA) properties (i.e., registers in PCI Express). Queues are created using the Connect command;
- NVMe over Fabrics uses the Disconnect command to delete an I/O Submission Queue and corresponding I/O Completion Queue;
- Metadata, if supported, shall be transferred as a contiguous part of the logical block. NVMe over Fabrics does not support transferring metadata from a separate buffer;
- NVMe over Fabrics does not support PRPs but requires use of SGLs for Admin, I/O, and Fabrics commands. This differs from NVMe over PCIe where SGLs are not supported for Admin commands and are optional for I/O commands;
- NVMe over Fabrics does not support Completion Queue flow control. This requires that the host ensures there are available Completion Queue slots before submitting new commands; and
- NVMe over Fabrics allows Submission Queue flow control to be disabled if the host and controller agree to disable it. If Submission Queue flow control is disabled, the host is required

to ensure that there are available Submission Queue slots before submitting new commands.

NVMe over Fabrics requires the underlying NVMe Transport to provide reliable NVMe command and data delivery. An NVMe Transport is an abstract protocol layer independent of any physical interconnect properties. An NVMe Transport may expose a memory model, a message model, or a combination of the two. A memory model is one in which commands, responses and data are transferred between fabric nodes by performing explicit memory read and write operations while a message model is one in which only messages containing command capsules, response capsules, and data are sent between fabric nodes.

The only memory model NVMe Transport supported by NVMe [\[NVME\]](#) is PCI Express, as defined in the NVMe Base specification. While differences exist between NVMe over Fabrics and NVMe over PCIe implementations, both implement the same architecture and command sets. But NVMe SCSI Translation reference is only using the NVMe over Fabrics not the memory model. NVMe over Fabrics utilizes the protocol layering shown in Figure 1. The native fabric communication services and the Fabric Protocol and Physical Fabric layers in Figure 1 are outside the scope of this specification.



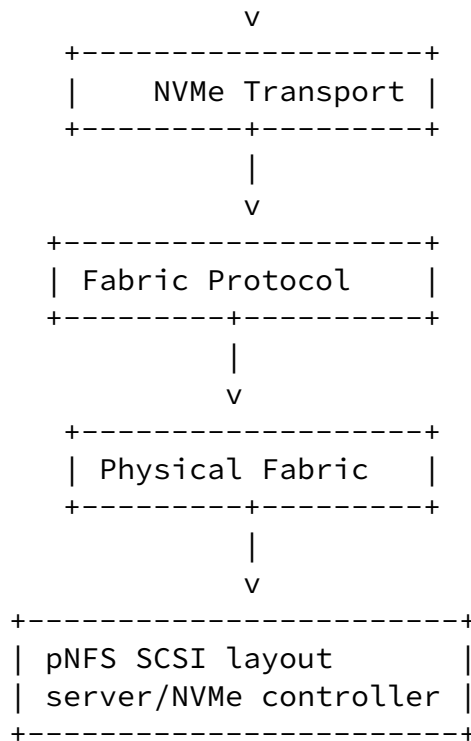
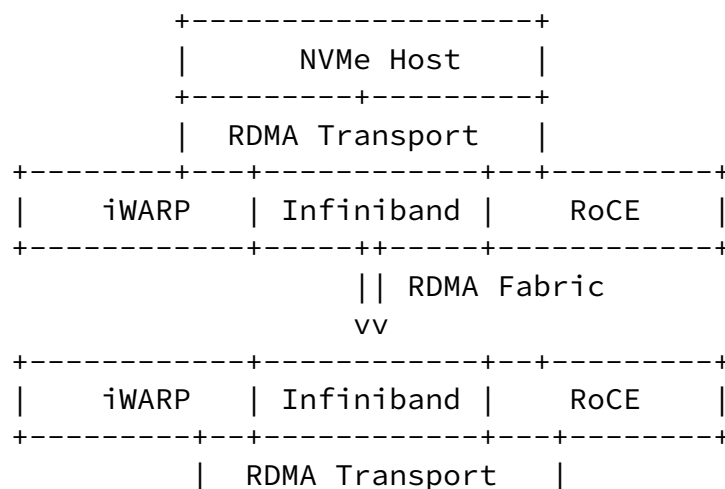


Figure 1: pNFS SCSI over NVMe over Fabrics Layering

An NVM subsystem port may support multiple NVMe Transports if more than one NVMe Transport binding specifications exist for the underlying fabric (e.g., an NVM subsystem port identified by a Port ID may support both iWARP and RoCE). This draft is also defining NVMe binding implementation that uses the Transport type of RDMA Transport. The RDMA Transport is RDMA Provider agnostic. The diagram in Figure 2 illustrates the layering of the RDMA Transport and common RDMA providers (iWARP, InfiniBand, and RoCE) within the host and NVM subsystem.



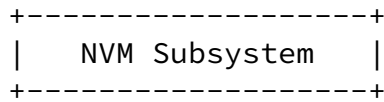


Figure 2: RDMA Transport Protocol Layers

NVMe over Fabrics allows multiple hosts to connect to different controllers in the NVM subsystem through the same port. All other aspects of NVMe over Fabrics multi-path I/O and namespace sharing are equivalent to that defined in the NVMe Base specification.

An association is established between a host and a controller when the host connects to a controller's Admin Queue using the Fabrics Connect command. Within the Connect command, the host specifies the Host NQN, NVM Subsystem NQN, Host Identifier, and may request a specific Controller ID or may request a connection to any available controller. The host is the pNFS client and the controller is the NFSv4 server. The pNFS clients connect to the server using different network protocols and different transports excluding PCIe direct connection. While an association exists between a host and a controller, only that host may establish connections with I/O Queues of that controller.

NVMe over Fabrics supports both fabric secure channel and NVMe in-band authentication. An NVM subsystem may require a host to use fabric secure channel, NVMe in-band authentication, or both. The Discovery Service indicates if fabric secure channel shall be used for an NVM subsystem. The Connect response indicates if NVMe in-band authentication shall be used with that controller. For SCSI over NVMe over Fabrics only the in-band authentication model will be used as the fabric secure channel is only supported with PCIe transport memory model not supported by SCSI layout protocol.

The pNFS SCSI layout uses the Device Identification VPD page (page code 0x83) from [\[SPC4\]](#) to identify the devices used by a layout. There are several ways to build SCSI Device

## [2.1.](#) Volume Identification

Identification descriptors from NVMe Identify data included in the Controller Identify Attributes specific to NVMe over Fabrics specified in the Identify Controller fields in Section 4.1 of

[[NVMEoF](#)]. This document uses a subset of this information to identify LUs backing pNFS SCSI layouts.

To be used as storage devices for the pNFS SCSI layout, NVMe devices MUST support the EUI-64 [[RFC8154](#)] value in the Identify Namespace data, as the methods based on the Serial Number for legacy devices might not be suitable for unique addressing needs and thus MUST NOT be used. UUID identification can be added by using a large enough enum value to avoid conflict with whatever T10 might do in a future version of the SCSI [[SBC3](#)] standard (the underlying SCSI field in SPC is 4 bits, so an enum value of 32 in this draft MUST be used). For NVMe, these identifiers need to be obtained via the Namespace Identification Descriptors in NVMe 1.4 (returned by the Identify command with the CNS field set to 03h).

## [2.2](#). Client Fencing

The SCSI layout uses Persistent Reservations to provide client fencing. For this both the MDS and the Clients have to register a key with the storage device, and the MDS has to create a reservation on the storage device. The pNFS SCSI protocol implements fencing using persistent reservations (PRs), similar to the fencing method used by existing shared disk file systems. To allow fencing individual systems, each system MUST use a unique persistent reservation key. The following is a full mapping of the required PR IN and PR OUT SCSI command to NVMe commands which MUST be used when using NVMe devices as storage devices for the pNFS SCSI layout.

### [2.2.1](#). Reservation Key Generation

Prior to establishing a reservation on a namespace, a host shall become a registrant of that namespace by registering a reservation key. This reservation key may be used by the host as a means of identifying the registrant (host), authenticating the registrant, and preempting a failed or uncooperative registrant. This document assigns the burden to generate unique keys to the MDS, which MUST generate a key for itself before exporting a volume and a key for each client that accesses SCSI layout volumes.

One important difference between SCSI Persistent Reservations and NVMe Reservations is that NVMe reservation keys always apply to all controllers used by a host (as indicated by the NVMe Host Identifier)



This behavior is somewhat similar to setting the ALL\_TG\_PT bit when registering a SCSI Reservation key, but actually guaranteed to work reliably.

### [2.2.2. MDS Registration and Reservation](#)

Before returning a PNFS SCSI VOLUME\_BASE volume to the client, the MDS needs to prepare the volume for fencing using NVMeReservations. Registering

a reservation key with a namespace creates an association between a host and a namespace. A host that is a registrant of a namespace may use any controller with which that host is associated (i.e., that has the same Host Identifier, refer to [\[NVME\]](#) section-5.21.1.26) to access that namespace as a registrant.

### [2.2.3. Client Registration](#)

#### [2.2.3.1 SCSI client](#)

Before performing the first I/O to a device returned from a GETDEVICEINFO operation, the client will register the reservation key returned by the MDS with the storage device by issuing a "PERSISTENT RESERVE OUT" command with a service action of REGISTER with the "SERVICE ACTION RESERVATION KEY" set to the reservation key.

#### [2.2.3.2 NVMe Client](#)

A client registers a reservation key by executing a Reservation Register command (refer to [\[NVME\]](#) [section 6.11](#)) on the namespace with the Reservation Register Action (RREGA) field cleared to 000b (i.e., Register Reservation Key) and supplying a reservation key in the New Reservation Key (NRKEY) field. A client that is a registrant of a namespace may register the same reservation key value multiple times with the namespace on the same or different controllers. There are no restrictions on the reservation key value used by hosts with different Host Identifiers.

### [2.2.4. Fencing Action](#)

#### [2.2.4.1 SCSI client](#)

In case of a non-responding client, the MDS fences the client by issuing a "PERSISTENT RESERVE OUT" command with the service action set to "PREEMPT" or "PREEMPT AND ABORT", the "RESERVATION KEY" field set to the server's reservation key, the service action "RESERVATION KEY" field set to the reservation key associated with the non-responding client, and the "TYPE" field set to 8h (Exclusive Access - Registrants Only).

#### [2.2.4.2](#) NVMe Client

A host that is a registrant may preempt a reservation and/or registration by executing a Reservation Acquire command (refer to [section 6.10](#)), setting the Reservation Acquire Action (RACQA) field to 001b (Preempt), and supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field. The CRKEY value shall match that used by the registrant to register with the namespace. If the CRKEY value does not match, then the command is aborted with status Reservation Conflict.

If the PRKEY field value does not match that of the current reservation holder and is equal to 0h, then the command is aborted with status Invalid Field in Command. A reservation preempted notification occurs on all controllers in the NVM subsystem that are associated with hosts that have their registrations removed as a result of actions taken in this section except those associated with the host that issued the Reservation Release command. After the MDS preempts a client, all client I/O to the LU fails. The client SHOULD at this point return any layout that refers to the device ID that points to the LU.

#### [2.2.5](#). Client Recovery after a Fence Action

A client that detects a NVMe status codes (I/O error) on the storage devices MUST commit all layouts that use the storage device through the MDS, return all outstanding layouts for the device, forget the device ID, and unregister the reservation key.

Future GETDEVICEINFO calls MAY refer to the storage device again, in which case the client will perform a new registration based on the key provided. If a reservation holder attempts to obtain a reservation of a different type on a namespace for which that host already is the reservation holder, then the command is aborted with status Reservation Conflict. It is not an error if a reservation holder attempts to obtain a reservation of the same type on a namespace for which that host already is the reservation holder.

NVMe over Fabrics [[NVMeoF](#)] utilizes the same controller architecture as that defined in the NVMe Base specification [[NVME](#)].

This includes using Submission and Completion Queues to execute commands between a host and a controller. Section 8.20 of [\[NVME\]](#) base specifications describes the relationship between a controller (MDS) and a namespace associated to the Clients. In a static controller model used by SCSI layout, controllers that may be allocated to a particular Client may have different state at the time the association is established.

### [2.3.](#) Volatile write caches

The Volatile Write Cache Enable (WCE) bit (i.e., bit 00) in the Volatile Write Cache Feature (Feature Identifier 06h) is the Write Cache Enable field in the NVMe Get Features command, see Section-5.21.1.6 of [\[NVME\]](#). If a write cache is enable on a NVMe device used as a storage device for the pNFS SCSI layout, the MDS MUST ensure to use the NVMe FLUSH command to flush the volatile write cache. If there is no volatile write cache on the server, then attempts to access this NVMe Feature cause errors. The Get Features command specifying the Volatile Write Cache feature identifier is expected to fail with Invalid Field in Command status.

## [3.](#) Security Considerations

Since no protocol changes are proposed here, no security considerations apply. But the protocol is assuming that NVMe Authentication commands are implemented in the NVMe Security Protocol as the format of the data to be transferred is dependent on the Security Protocol. Authentication Receive/Response commands return the appropriate data corresponding to an Authentication Send command as defined by the rules of the Security Protocol. As the current draft is only supporting NVMe over fabric In-band protocol the Authentication requirements for security commands are based on the security protocol indicated by the SECP field in the command and DO NOT require authentication when used for NVMe in-band authentication. When used for other

purposes, in-band authentication of the commands is required.

#### [4.](#) IANA Considerations

The document does not require any actions by IANA.

Faibish Expires January 6, 2021 [Page 10]

---

Internet-Draft pNFS SCSI/NVMe Layout over Fabrics July 2020

#### [5.](#) Normative References

[NVME] NVM Express, Inc., "NVM Express Revision 1.4", June 10, 2019.

[NVMeoF] "NVM Express over Fabrics Revision 1.1", July 26, 2019

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.

[RFC8154] Hellwig, C., "Parallel NFS (pNFS) Small Computer System Interface (SCSI) Layout", May 2017.

[SBC3] INCITS Technical Committee T10, "SCSI Block Commands-3", ANSI INCITS INCITS 514-2014, ISO/IEC 14776-323, 2014.

[SPC4] INCITS Technical Committee T10, "SCSI Primary Commands-4", ANSI INCITS 513-2015, 2015.

[FC-NVMe] INCITS Technical Committee T10, "Fibre Channel - Non-Volatile Memory Express", ANSI INCITS 540, 2018.

[RFC7525] Sheffer, Y., "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)" also known as [BCP 195](#).

Author's Address

Sorin Faibish  
Dell EMC

228 South Street  
Hopkinton, MA 01774  
United States of America

Phone: +1 508-249-5745  
Email: faibish.sorin@dell.com

David Black  
Dell EMC  
176 South Street  
Hopkinton, MA 01748  
United States of America

Phone: +1 774-350-9323  
Email: david.black@dell.com

Christoph Hellwig  
Email: hch@lst.de