Intel® Virtual RAID on CPU (Intel® VROC) Driver for ESXi*

User Guide

Revision 009

September 2025

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis. You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

All product plans and roadmaps are subject to change without notice.

The products described may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at intel.com.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Altering clock frequency, voltage, or memory interface speeds may void any product warranties and reduce stability, security, performance, and life of the processor and other components. Intel has not validated processor running memory above Plan-Of-Record (POR) speed. DRAM/DIMM devices should support desired speed, check with DRAM/DIMM vendors for details. System manufacturers are responsible for all validation and assume the risk of any stability, security, performance, or other functional issues resulting from such alterations.

Contents

1 Overview 5
1.1 Reference OEM Platform Documentation 5
1.2 Terms and Acronyms 5

2 Product Requirements 7
2.1 Platform and VMD Requirements 7
2.2 Supported RAID Options 7
2.3 Hot Plug Requirements 8
2.4 RAID Spare Drive and Rebuild Configuration Recommendations 8

3 Product Installation/Upgrade 10
3.1 ESXi* VROC Driver Download Instructions and Supported Versions 10
3.1.1 Supported ESXi* Versions Based on Intel® Xeon® Platform 11
3.2 ESXi* VROC Driver Upgrade Procedure 11
3.2.1 Installation of the ESXi* VMD Outbox (async) Driver 11

4 Intel® VROC and LED Management Tool 13
4.1 Disk and RAID Command Options 13
4.2 RAID Volume Management 15
4.3 LED Management 17
4.4 NVMe Firmware Update Management 19

5 VMware* ESXi Commands – General Reference 21
5.1 List/Remove Software 21
5.2 Disk Listing 21
5.3 Disk Management 22
5.4 Saving Core Dump Example 22

6 Injecting Outbox ESXi* VROC Driver into ESXi* Bootable Image 23
6.1 Tools Needed 23
6.2 Software Needed and Injection Procedure Summary 23

7 Appendix 25
7.1 Expanded Intel® VROC and LED Management Tool Command Output Options 25
7.2 Custom Development of User Space Tools to Send NVMe Admin Commands to NVMe SSDs Behind Intel® VMD 26

Figures

Figure 2-1. Graphical Representation of Boot Options 9

Tables

Table 1-1. Glossary of Terms and Acronyms 5

Revision History

Revision	Description	Revision Date
001	Initial Release	November 2022
002	Updated Chapters 3, 4, 5 and 6	April 2023
003	Updated Sections 2.2, 2.3, 2.4 and 4.1	August 2023
004	Updated Sections 1.2, 2.2, 4.1 and 4.2	October 2023
005	Updated Multiple Sections	July 2024
006	Updated Sections 3, 4	September 2024
007	Updated Sections 2, 3, 4 and 7	December 2024
008	Updated Section 2.1	May 2025
009	Updated Section 7.2	Sept 2025

§§

1 Overview

This document provides guidance for the Intel® VROC Driver for ESXi*, referred to as the ESXi* VROC driver.

The ESXi* VROC driver supports ESXi* versions 7.0U3d and 8.0x. This document assumes reader familiarity with the ESXi* Operating System/Hypervisor and does not provide detailed instructions for installing ESXi* or copying files within an ESXi* environment.

The ESXi* VROC driver is designed to work with the Intel® VMD hardware logic implemented on Intel® Xeon® Scalable Processors. Intel® VMD is much like an HBA controller for NVMe* SSDs and adds robust management for NVMe* SSDs in the data center such as Hot Plug, LED management, and error handling, in addition to being a dependency for Intel® VROC configuration.

VMware* ESXi 6.5 and 6.7 reached their end-of-life, as announced by VMware*. No support is available for these versions, and it is recommended that customers upgrade to ESXi* version 7.0U3 or 8.0x.

1.1 Reference OEM Platform Documentation

Refer to your OEM for a full list of available feature sets. If any of the information in this document conflicts with the support information provided by the platform OEM, the platform documentation and configurations take precedence.

Customers should always contact the place of purchase or system/software manufacturer with support questions about their specific hardware or software configuration.

1.2 Terms and Acronyms

Table 1-1. Glossary of Terms and Acronyms

Term	Definition
NVMe	Non-Volatile Memory Express
ISO	International Standards Organization
SSD	Solid State Drive
VMD	Volume Management Device
IOVP	I/O Vendor Partner
GA	General Availability
vSAN	Virtual Storage Area Network
LED	Light Emitting Diode

Term	Definition
ESXi	Elastic Sky X integrated
OS	Operating System
OEM	Original Equipment Manufacturer
VROC	Virtual RAID on CPU
HII	Human Interface Infrastructure
PEM	Power Entry Module
BKC	Best Known Configuration
NS	Namespace
VCG	VMware Compatibility Guide

§§

2 Product Requirements

2.1 Platform and VMD Requirements

Platform and VMD requirements to use Intel® Virtual RAID on CPU (Intel® VROC) RAID volumes:

- Generations 1-6 Intel® Xeon® Scalable Processor.
- Intel® NVMe* SSDs which comply with NVMe specifications attached to a Hot Swappable Backplane (HSBP) for Surprise Hot Plug management.
- Intel® VMD supports a limit of 128 NVMe drive namespaces accessible to host software.
- 48 NVMe* SSDs is the maximum for total devices supported on a platform.
- A licensed Intel® VROC key (hardware, software, On-demand, as applicable) is required on the system to enable VROC-configured RAID volumes.
- Intel® VROC UEFI drivers should be integrated with the system BIOS and consist of the following UEFI driver set:
  - VMDVROC_1.efi
  - VMDVROC_2.efi
  - With the above UEFI drivers loaded, the BIOS will display a HII menu within the BIOS setup menu as "Intel® Virtual RAID on CPU". BIOS menu paths and options may differ depending on BIOS vendor.
- Intel® VMD must be enabled in the BIOS. Check with your BIOS vendor for instructions on enabling Intel® VMD in the BIOS setup menu.
- "Intel® Virtual RAID on CPU" is the BIOS menu option for creating Intel® VROC RAID volumes and associated Intel proprietary metadata for RAID support in the ESXi* Operating System.

2.2 Supported RAID Options

The "Intel® Virtual RAID on CPU" BIOS menu is commonly used to create RAID volume(s), although volumes may also be created using the Intel® VROC and LED Management CLI Tool (See Section 4). Currently supported RAID configurations are:

- RAID 1 boot volume.
- RAID 1 data volume.
- Only 1 volume is supported on a given array of drives. Matrix RAID is not supported.
- RAID 0/10/5 are not currently supported.
- A boot datastore and data datastore may coexist on the same VMD controller. Example: Connect 4 drives behind one Intel® VMD Domain/Controller. Create two RAID 1 volumes. Use one volume for the operating system and the other for data.
- vSAN use case exception: based on vSAN certification requirements, a boot device and data device may not be attached to the same storage controller.

- Additional scenario clarifications:
  - Multiple RAID 1 volumes are supported for NVMe drives behind Intel® VMD. Intel verifies functionality of two volumes per domain/controller. The maximum volume limit has not been tested as of this writing.
  - A disk can only be used in one RAID volume. If part of the disk is used in a RAID volume, the rest of the disk cannot be used elsewhere (i.e.: no Matrix RAID).
  - Each LUN or volume must contain only one VMFS datastore.

2.3 Hot Plug Requirements

Hot plug for a VMD passthrough disk may be performed when the disk is assigned to a VM as a virtual disk and the VM is powered down. In this case, Intel recommends removing the datastore before hot plugging the device.

Intel does not recommend that customers perform hot plug for VMD passthrough disk if the disk is already assigned to a VM as a virtual disk and when VMs are running.

The above guidance complies with VMware's recommendations that IO to the disk be stopped and the datastore removed or unmounted prior to hot plug actions.

Depending on the validation platform of choice, Intel® VMD must be enabled on the surprise hot plug capable lanes of the platform. For surprise hot plug to work correctly, some platforms allow a jumper to be set to disable ACPI hot plug. PCIe* hot plug is required to support Intel® VMD.

RAID 1 full volume hot plug, where both RAID disks are physically removed from a system, is not a supported scenario.

2.4 RAID Spare Drive and Rebuild Configuration Recommendations

Options for drive replacement and RAID rebuild when using RAID 1 volumes are the following:

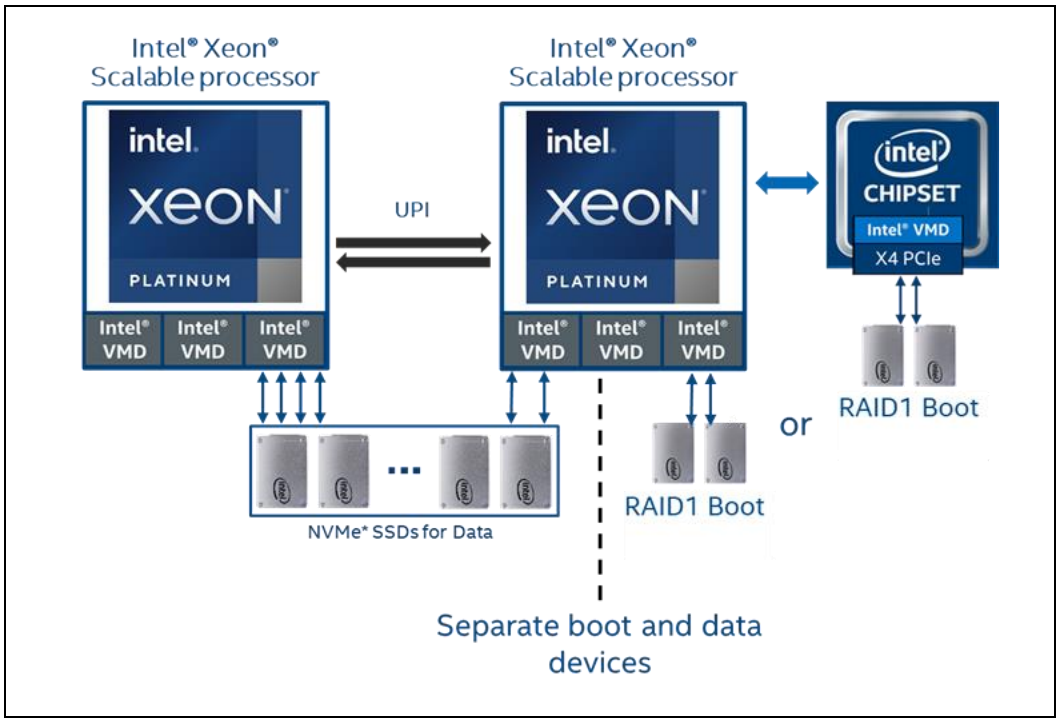
- When configuring the ESXi* boot drive(s) as a RAID 1, Intel recommends a third drive be placed in the system and be designated as an online spare. In case of failure of a RAID 1 member drive, automatic rebuild to the spare drive occurs, making it an active member without further administrator actions.
- If no spare drive has been designated in the system, and a member RAID 1 drive fails, recommended actions are to unplug the failed drive from the system (use Locate LED to assist as needed) and insert a clean drive of capacity at least equal to the RAID volume size.
- The newly inserted drive does not auto rebuild into the RAID 1. It must be manually selected before a rebuild begins, for example: **`vmd rebuildstart vmhba0 -v 59 -d 1`** (based on volume details provided by the *intel-vmdr-cli* tool. This tool is discussed in greater depth in Section 4).

If system slot availability does not allow for a separate slot to insert a spare drive, i.e.: all slots are occupied, another option for auto-rebuild is to provide an external (on the shelf) spare drive. This option may simplify administrative drive replacement if immediate insertion and volume rebuild is needed. Test prior to using in a production environment. Recommended sequence of actions is the following:

1. ESXi* installed and the latest driver/tool installed.

2. Assign a third disk as a spare in BIOS.
3. Boot to ESXi*.
4. With the user tool, confirm and locate the disk that was designated as spare (LED locate).
5. Unplug the designated spare (this disk has been marked in BIOS Intel VROC settings as a spare drive and the slot it occupies is now available for another disk).
6. If a member of the RAID 1 fails, unplug the failed drive, and insert the external spare drive into the vacated slot.
7. Rebuild starts immediately.

Figure 2-1. Graphical Representation of Boot Options



§§

3 Product Installation/Upgrade

3.1 ESXi* VROC Driver Download Instructions and Supported Versions

An inbox ESXi* VROC driver is provided with ESXi* releases 7.0x/8.0x/9.0x. As of this writing, the ESXi* version 7.0x inbox driver is a 2.x level driver, version 8.0x inbox driver is a 3.x level driver, and version 9.0x inbox driver is a 9.x level driver. The inbox driver is production quality and fully functional at the time of ESXi* GA release.

To address improvements/fixes identified for the inbox ESXi* VROC driver following ESXi* 7.0x and 8.0x GA releases, outbox or async drivers have been provided by Intel at the links below.

For VMware* ESXi* 9.0 and later, only an inbox Intel® VROC driver is available. As of VMware* ESXi* 9.0, async/add-in drivers are no longer provided. For issues potentially related to ESXi 9.0x Intel® VROC drivers, contact your Intel representative for resolution.

Note: The ESXi* 2.x inbox driver may be upgraded to a 3.x or 9.x driver depending on platform version. However, it is not recommended to downgrade a 3.x or 9.x driver to a 2.x driver.

Production ESXi* VROC drivers are VMware* certified and can be downloaded from Intel's public web site as listed below.

How to download VMware* Intel® ESXi* VROC Production IOVP Signed/Certified Drivers

1. For Intel® VROC Driver for ESXi* 8.x, access via the link https://www.intel.com/content/www/us/en/download/784751/intel-vroc-driver-and-management-tool-for-vmware-esxi-8-x.html
2. For Intel® VROC Driver for ESXi* 7.x, access via the link https://www.intel.com/content/www/us/en/download/784752/intel-vroc-driver-and-management-tool-for-vmware-esxi-7-x.html

The links above also include downloads for certified driver **Release Notes**, along with the companion **Intel® VROC and LED Management Tool for VMware* ESXi***. See Section 4 for additional details on this tool.

3.1.1 Supported ESXi* Versions Based on Intel® Xeon® Platform

For questions related to ESXi* version support based on Intel® Xeon® platform, please refer to the following: Intel® Virtual RAID on CPU (Intel® VROC) Operating Systems Support List. See *VMware* ESXi* Operating Systems* section.

To assist with confirming the correct ESXi* driver for a given ESXi* version or platform, VMD Device ID for each Intel® Xeon® platform follows.

- VMD Device ID 28c0: 3rd Gen Intel® Xeon® Scalable Processors and later
- VMD Device ID 201d: Prior to 3rd Gen Intel® Xeon® Scalable Processors

See VMware's Compatibility Guide (VCG) for a full listing of certified VROC drivers for each ESXi* release.

3.2 ESXi* VROC Driver Upgrade Procedure

As mentioned previously, VMware* ESXi has an integrated inbox ESXi* VROC driver. The **`esxcli software vib list |grep vmd`** command displays the *iavmd* driver listed as in the example below:

```
[root@localhost:~] esxcli software vib list |grep vmd
iavmd                          3.0.0.1010-9vmw.802.0.0.22380479      VMW     VMwareCertified
lsuv2-intelv2-nvme-vmd-plugin  2.7.2173-2vmw.802.0.0.22380479        VMware  VMwareCertified
```

3.2.1 Installation of the ESXi* VMD Outbox (async) Driver

To update or install an async driver, VMware* recommends using the following component installation command syntax:

```
esxcli software component apply -d <path_to_component.zip> --no-sig-check
```

For example:

```
esxcli software component apply -d /tmp/INT-esx-8.0.0-Intel-Volume-Mgmt-Device-
9.0.0.1006-1OEM.800.1.0.20613240.zip --no-sig-check
```

```
Installation Result
   Message: The update completed successfully, but the system needs to be rebooted for the changes to be effective.
   Components Installed: Intel-Volume-Mgmt-Device_9.0.0.1006-1OEM.800.1.0.20613240
   Components Removed: Intel-Volume-Mgmt-Device_3.0.0.1010-9vmw.802.0.0.22380479
   Components Skipped:
   Reboot Required: true
   DPU Results:
```

Note: As a change to previous ESXi* VROC driver update practice, it is no longer recommended to perform a VIB file standalone installation for production use. This installation method, which requires a **`-f`** or **`--force`** option (starting with ESXi* 8.0), lowers the driver security acceptance level below the ESXi* host level and may introduce vulnerabilities into ESXi* operations:

```
Not recommended for production use:  esxcli software vib install -d /tmp/[xxxxx].vib -f -
-no-sig-check
```

Note: **`--no-sig-check`** is used to load an unsigned driver, usually for test purposes. Production drivers, however, are signed/certified and do not require this parameter.

Reboot the host to activate the updated driver. Use the **`esxcli software vib list |grep vmd`** command to confirm the newly updated *iavmd* driver version.

```
[root@localhost:~] esxcli software vib list | grep vmd
iavmd                          9.0.0.1006-1OEM.800.1.0.20613240      INT     VMwareCertified
lsuv2-intelv2-nvme-vmd-plugin  2.7.2173-2vmw.802.0.0.22380479        VMware  VMwareCertified
```

§§

4 Intel® VROC and LED Management Tool

As a companion to the Intel® VROC driver for ESXi* 7.0x and 8.0x, Intel offers a command line utility for RAID/non-RAID volume, LED, and NVMe firmware management. The utility is named Intel® VROC and LED Management Tool, using binary files named *intel-vmdr-cli* (ESXi* 8.0x) or *intel-vmdr-user* (ESXi* 7.0x). It is commonly referred to as the user or CLI tool.

The Intel® VROC and LED Management Tool for VMware* ESXi* has been discontinued for VMware* ESXi* 9.0 and later. To manage RAID configurations for VMware* ESXi* 9.0 and later, use PreOS/UEFI VROC* menu options.

The CLI tool is developed alongside a specific ESXi* VROC driver release to ensure compatibility. Use of the latest production ESXi* VROC driver along with its paired CLI tool is strongly recommended. Refer to the links below (same as in Section 3.1) for CLI tool download along with compatible ESXi* VROC Driver-to-CLI tool mapping:

1. For Intel® VROC LED Management Tool for ESXi* 8.x, access via the link https://www.intel.com/content/www/us/en/download/784751/intel-vroc-driver-and-management-tool-for-vmware-esxi-8-x.html
2. For Intel® VROC LED Management Tool for ESXi* 7.x, access via the link https://www.intel.com/content/www/us/en/download/784752/intel-vroc-driver-and-management-tool-for-vmware-esxi-7-x.html

The CLI tool is installed using the command syntax shown below (**`--no-sig-check`** option is required only when installing a pre-production or unsigned release).

```
esxcli software component apply -d <path_to_component.zip> --no-sig-check
```

Note: No reboot is required to activate the tool for ESXi* 7.0. However, reboot is required to activate the tool for ESXi* 8.0 as displayed in the following.

```
Installation Result
   Message: The update completed successfully, but the system needs to be rebooted for the changes to be effective.
   Components Installed: intel-vmdr-cli_9.0.0.2329-1OEM.800.1.0.20613240
   Components Removed:
   Components Skipped:
   Reboot Required: true
   DPU Results:
```

After installing the *intel-vmdr-cli* (ESXi* 8.0x) or *intel-vmdr-user* (ESXi* 7.0x) tool, it will be accessible at the following ESXi* host paths and CLI names, depending on the ESXi* version used:

- For ESXi* 8.0x: **`/opt/intelvmdrcli/bin/intel-vmdr-cli`**
- For ESXi* 7.0x: **`/opt/intel/bin/intel-vmdr-user`**

4.1 Disk and RAID Command Options

General Syntax (See Section 7 Appendix for expanded command output listings):

```
intel-vmdr-cli {command} {adapter name} {options}
```

Note: ESXi* 8.0u2 CLI tool command examples are presented in this user guide.

Available RAID level options are dependent on the Intel VROC license installed on the system. See your Intel representative for assistance in matching the appropriate license for RAID requirements.

To view the Intel VROC license currently installed in an ESXi* system:

```
intel-vmdr-cli getlicenseinfo

./intel-vmdr-cli getlicenseinfo
Intel(R) VROC Premium SKU (Premium License)
RAID Level Support: 1
RAID Support for 3rd Party SSD: Yes
```

Note: **`./intel-vmdr-cli`** as a standalone command (with no options) displays an extensive list of commands and sample syntax. Examples given in this document are based on ESXi* 8.0U2 and follow this command line guidance:

```
./intel-vmdr-cli

Management Interface Version: 3000000

Usage:

intel-vmdr-cli raidctrllist
   intel-vmdr-cli vmdctrllist
   intel-vmdr-cli disklist
   intel-vmdr-cli volumelist
   intel-vmdr-cli getlicenseinfo
   intel-vmdr-cli setled [RAIDCtrlName] -d[TargetID] -l[off, identify]
      eg: intel-vmdr-cli setled vmhba2 -d 1 -l identify

   intel-vmdr-cli getled [RAIDCtrlName] -d[TargetID]
      eg: intel-vmdr-cli getled vmhba2 -d 1

   intel-vmdr-cli createvol [RAIDCtrlName] -b[StripSize] -d[TargetIDs] -l[RAIDLevel] -
n[VolumeName] -s[SourceDiskID (Optional)] -c[VolumeSize]
      eg(Volume Creation): intel-vmdr-cli createvol vmhba2 -b 64 -d 0,1 -l 1 -n
testvolume -c 5120

      eg(RAID Migration): intel-vmdr-cli createvol vmhba2 -b 64 -d 0,1 -l 1 -n testvolume
-s 1 -c 5120

   intel-vmdr-cli volinitialize [RAIDCtrlName] -v[volumeID]
      eg: intel-vmdr-cli volinitialize vmhba2 -v 59

   intel-vmdr-cli deletevol [RAIDCtrlName] -d[TargetID]
      eg: intel-vmdr-cli deletevol vmhba2 -d 127

   intel-vmdr-cli verifyvol [RAIDCCtrlName] -d[TargetID] -t[VerifyType]
      eg: intel-vmdr-cli verifyvol vmhba2 -d 59 -t 0

   intel-vmdr-cli getvolinfo [RAIDCtrlName] -d[TargetID]
      eg: intel-vmdr-cli getvolinfo vmhba2 -d 59

   intel-vmdr-cli getdiskinfo [RAIDCtrlName] -d[TargetID]
      eg: intel-vmdr-cli getdiskinfo vmhba2 -d 1

   intel-vmdr-cli markpassthrough [RAIDCtrlName] -d[TargetID]
      eg: intel-vmdr-cli markpassthrough vmhba2 -d 1
```

```
   intel-vmdr-cli rebuildstart [RAIDCtrlName] -v[VolumeID] -d[TargetID]
      eg: intel-vmdr-cli rebuildstart vmhba2 -v 59 -d 1

   intel-vmdr-cli markdiskspare [RAIDCtrlName] -s[SpareDiskID]
      eg: intel-vmdr-cli markdiskspare vmhba2 -s 1

   intel-vmdr-cli nvmeidctrl [RAIDCtrlName] -d[TargetID]
      eg: intel-vmdr-cli nvmeidctrl vmhba2 -d 1

   intel-vmdr-cli nvmegetlogpage [RAIDCtrlName] -d[TargetID] -i[LogPageId] -l[LogLength]
-n[NamespaceID]
      eg: intel-vmdr-cli nvmegetlogpage vmhba2 -d 1 -i 2 -l 512 -n 4294967295

   intel-vmdr-cli nvmefwdownload [RAIDCtrlName] -d[TargetID] -f[FirmwarePath]
      eg: intel-vmdr-cli nvmefwdownload vmhba2 -d 1 -f "/tmp/your_firmware_image.bin"

   intel-vmdr-cli nvmefwcommit [RAIDCtrlName] -d[TargetID] -s[FirmwareSlot] -
a[FirmwareAction]
      eg: intel-vmdr-cli nvmefwcommit vmhba2 -d 1 -s 1 -a 0


Description:
   VMDCtrlName   VMD Controller name = vmhba[X-x]
   RAIDCtrlName  RAID Controller name = vmhba[X]
   -d            TargetID of Disk/Volume
   -b            Strip size in KB
   -n            Name of volume
                 NVMe namespace Id (for nvmegetlogpage)
   -c            Volume size
   -v            Volume ID
   -l            State of the led[off, identify] (for setled)
                 RAID level (for createvol)
                 NVMe log page length (for nvmegetlogpage)
   -s            Source disk of volume(Applicable only for Migration)
                 Spare disk ID (for markdiskspare)
                 NVMe SSD firmware slot Id (for nvmefwcommit)
   -i            NVMe log page Id (for nvmegetlogpage)
   -f            NVMe firmware image path (for nvmefwdownload)
   -a            NVMe firmware commit action (for nvmefwcommit)
   -t            Type of volume verification (for verifyvol)
```

Many command options exist for viewing, creating, deleting, or analyzing disk and RAID volumes as listed below. Several of these are presented in the next section to demonstrate syntax and expected results.

4.2 RAID Volume Management

To view RAID-configured volumes:

```
intel-vmdr-cli volumelist
```

This displays any RAID volumes in the system. In our case, we have no RAID volumes currently configured, as shown below.

```
./intel-vmdr-cli volumelist
Volume not found
```

Volume Creation

Referring to results of the *./intel-vmdr-cli disklist* command, VMD controller 0 (vmhba0) currently contains 4 PASSTHROUGH drives. PASSTHROUGH drives in this example are connected to a VMD controller but are not yet in a RAID configuration.

```
./intel-vmdr-cli disklist

Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 0     Serial#: "BTLJ85110CKX4P0DGN
"         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 1     Serial#: "BTLJ85110C514P0DGN
"         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 2     Serial#: "BTLJ74550A904P0DGN
"         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 3     Serial#: "PHLJ832307RK4P0DGN
"         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX01"   Controller: vmhba1      TargetId: 0     Serial#: "BTLJ7244051S1P0FGN
"         State: PASSTHROUGH
```

To create a separate RAID 1 data volume using two of the available PASSTHROUGH drives (TargetId 0 and 1), with name of *datavol*, and size of *5120*MB:

```
intel-vmdr-cli createvol vmhba0 -b 64 -d 0,1 -l 1 -n datavol -c 5120

./intel-vmdr-cli createvol vmhba0 -b 64 -d 0,1 -l 1 -n datavol -c 5120
SUCCESS: VOLUME CREATED
```

The two targeted ID's now appear as MEMBER drives in the newly created volume:

```
./intel-vmdr-cli disklist
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 0     Serial#: "BTLJ85110CKX4P0DGN
"         State: MEMBER
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 1     Serial#: "BTLJ85110C514P0DGN
"         State: MEMBER
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 2     Serial#: "BTLJ74550A904P0DGN
"         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 3     Serial#: "PHLJ832307RK4P0DGN
"         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX01"   Controller: vmhba1      TargetId: 0     Serial#: "BTLJ7244051S1P0FGN
"         State: PASSTHROUGH
```

The newly created volume can be listed to confirm its TargetId and health State, in this case NORMAL. TargetID will be used in later commands to check volume info.

```
./intel-vmdr-cli volumelist
Volume Name: "datavol"  Controller: vmhba0      TargetId: 59    State: NORMAL
```

Note: ESXi* volumes are not initialized by default upon creation.

```
./intel-vmdr-cli getvolinfo vmhba0 -d 59
Volume state is NORMAL
Volume block size in bytes: 512
Volume initialized: No
Background Operation: None
Volume Name: datavol
Volume RAID level: 1
Volume stripe size in KB: 64
Volume size in MB: 5120
Volume disk count: 2
Volume member disk id: 0
Volume member disk id: 1
Volume failed disk count: 0
Volume offline disk count: 0
```

Manual initialization is required. For new volumes, initialization is performed quickly:

```
intel-vmdr-cli volinitialize vmhba0 -v 59
```

```
./intel-vmdr-cli volinitialize vmhba0 -v 59
SUCCESS: VOLUME INITIALIZATION STARTED
```

To manually start verification on a RAID volume, use the specified volume's parameters:

```
intel-vmdr-cli verifyvol vmhba0 -d 59 -t 0
```

```
./intel-vmdr-cli verifyvol vmhba0 -d 59 -t 0
SUCCESS: Volume verification process started.
Please use getvolinfo command to track verification progress.
```

Volume verification may be time consuming, depending on size of the volume. To check verification progress and display volume characteristics:

```
intel-vmdr-cli getvolinfo vmhba0 -d 59
```

```
./intel-vmdr-cli getvolinfo vmhba0 -d 59
Volume state is NORMAL
Volume block size in bytes: 512
Volume initialized: Yes
Background Operation: Verification
Background Operation Progress: 32.7%
Volume Name: datavol
Volume RAID level: 1
Volume stripe size in KB: 64
Volume size in MB: 5120
Volume disk count: 2
Volume member disk id: 0
Volume member disk id: 1
Volume failed disk count: 0
Volume offline disk count: 0
```

Volume Deletion

To remove an existing volume, specify the volume to be deleted:

```
intel-vmdr-cli deletevol vmhba0 -d 59
```

```
./intel-vmdr-cli deletevol vmhba0 -d 59
SUCCESS: VOLUME DELETED
```

4.3 LED Management

For LED Management, NVMe* devices should be attached to a system-integrated hot swappable backplane, switch, or connectors on the motherboard.

For help in running the CLI tool to test LED management, enter the following for syntax assistance:

```
intel-vmdr-cli setled
```

```
./intel-vmdr-cli setled
Usage:
   intel-vmdr-cli setled [RAIDCtrlName] -d[TargetID] -l[off, identify]
      eg: intel-vmdr-cli setled vmhba2 -d 1 -l identify
```

```
Description:
   VMDCtrlName   RAID Controller name = vmhba[X]
   -d            TargetID of Disk
   -l            State of the led[off, identify]
```

Example for setting LED state:

Options:

- **[-l <STATE>]** - LED state (off, identify)
- **[-d <NUM>]** - Target disk

Note: The CLI tool supports only **off** and **identify** states.

To find the disk number and target number, the **disklist** command is used:

```
intel-vmdr-cli disklist
```

```
./intel-vmdr-cli disklist
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 0     Serial#:
"BTLJ85110CKX4P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 1     Serial#:
"BTLJ85110C514P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 2     Serial#:
"BTLJ74550A904P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 3     Serial#:
"PHLJ832307RK4P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX01"   Controller: vmhba1      TargetId: 0     Serial#:
"BTLJ7244051S1P0FGN  "         State: PASSTHROUGH
```

The number after **TargetID:** is the disk number. In the example shown, controller *vmhba0* has 4 disks assigned, *vmhba0*-[0-3].

In the displays below, syntax is included to set LED's to **identify** and **off**.

Confirm current LED state on device *vmhba0-0* (disk number 0) of *vmhba0* (**off**):

```
intel-vmdr-cli getled vmhba0 -d 0
```

```
./intel-vmdr-cli getled vmhba0 -d 0
rp_get_led Called
Targeting Disk:0
LED state:off
```

Set LED state on the same drive to **identify**:

```
intel-vmdr-cli setled vmhba0 -d 0 -l identify
```

```
./intel-vmdr-cli setled vmhba0 -d 0 -l identify
Targeting Disk:0 with LED:identify
Request to Set LED on disk has completed.
```

Return LED state to **off**:

```
intel-vmdr-cli setled vmhba0 -d 0 -l off
```

```
./intel-vmdr-cli setled vmhba0 -d 0 -l off
Targeting Disk:0 with LED:off
Request to Set LED on disk has completed.
```

In this example, set LED status on device *vmhba0-1,* (disk number 1) to **off**:

```
intel-vmdr-cli setled vmhba0 -d 1 -l off

./intel-vmdr-cli setled vmhba0 -d 1 -l off
Targeting Disk:1 with LED:off
Request to Set LED on disk has completed.
```

Confirm the status register is correctly set to **off**:

```
intel-vmdr-cli getled vmhba0 -d 1

./intel-vmdr-cli getled vmhba0 -d 1
rp_get_led Called
Targeting Disk:1
LED state:off
```

4.4 NVMe Firmware Update Management

To update firmware on NVMe drives using the CLI tool, example commands and syntax follow.

Confirm Drive Details and Firmware Version

```
./intel-vmdr-cli disklist
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 0     Serial#:
"BTLJ85110CKX4P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 1     Serial#:
"BTLJ85110C514P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 2     Serial#:
"BTLJ74550A904P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX04"   Controller: vmhba0      TargetId: 3     Serial#:
"PHLJ832307RK4P0DGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPF21Q01"   Controller: vmhba1      TargetId: 0     Serial#:
"PHAL029300DA1P6MGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPF21Q01"   Controller: vmhba1      TargetId: 1     Serial#:
"PHAL0293009C1P6MGN  "         State: PASSTHROUGH
Disk Name: "INTEL SSDPE2KX01"   Controller: vmhba1      TargetId: 2     Serial#:
"BTLJ7244051S1P0FGN  "         State: PASSTHROUGH
```

Note: Global namespace ID is used here. Specific NVMe drive NS may also be used.

```
./intel-vmdr-cli  nvmegetlogpage vmhba1 -d 1 -i 3  -l 512 -n oxffffffff
Active Firmware Info           : 0x1
Firmware Revision for Slot 1 : (L0310100)
```

Obtain and Place Firmware Image

```
/tmp/L0310350.bin
```

Firmware Update – Stage 1 - Download

```
./intel-vmdr-cli nvmefwdownload vmhba1 -d 1 -f "/tmp/L0310350.bin"
Firmware download completed successfully
```

Firmware Update – Stage 2 - Commit

```
./intel-vmdr-cli nvmefwcommit vmhba1 -d 1 -s 1 -a 1
Firmware commit completed successfully, but it may require system reboot
```

Note: “-a” parameter may require “0” or other value due to drive/system type.

Reboot and Verify Updated Firmware Version

```
./intel-vmdr-cli  nvmegetlogpage vmhba1 -d 1 -i 3  -l 512 -n oxffffffff
Active Firmware Info           : 0x11
```

```
Firmware Revision for Slot 1 : (L0310350)
```

§§

5 VMware* ESXi Commands – General Reference

The following miscellaneous commands are provided for general reference and troubleshooting.

5.1 List/Remove Software

ESXi* version:

```
vmware -vl
```

List of installed drivers:

```
esxcli software vib list
```

Installed ESXi* VROC driver:

```
esxcli software vib list | grep iavmd
```

To remove driver:

```
esxcli software vib remove -n <driver name>
```

5.2 Disk Listing

List disks:

```
esxcfg-mpath -L
```

Count number of disks:

```
ls /vmfs/devices/disks/ | grep NVMe* | grep : -vc
```

List disks with controllers:

```
esxcfg-scsidevs -A
```

List controllers:

```
esxcfg-scsidevs -a
```

List all datastores present:

```
esxcli storage vmfs extent list
```

If no datastores show up, run this command:

```
esxcli storage vmfs snapshot list
```

5.3 Disk Management

Note: The **diskID** (**-d**) obtained via the **esxcfg-scsidevs -A** command follows the format:
t10.NVMe____INTEL_SSDPE2KX040T8_____BTLJ85110CKX4P0DGN__00000001

To remotely detach/disable a disk:

```
esxcli storage core device set --state=off -d
t10.NVMe____INTEL_SSDPE2KX040T8_____BTLJ85110CKX4P0DGN__00000001

To see results of the above command:

tail -f /var/log/vmkernel.log

2024-04-25T00:00:06.638Z In(182) vmkernel: cpu40:1051783 opID=c17c268d)ScsiDevice: 1831:
Device t10.NVMe____INTEL_SSDPE2KX040T8_____BTLJ85110CKX4P0DGN__00000001
has been turned off administratively.
```

To remotely attach/enable a disk:

```
esxcli storage core device set --state=on -d
t10.NVMe____INTEL_SSDPE2KX040T8_____BTLJ85110CKX4P0DGN__00000001

2024-04-25T00:02:27.611Z In(182) vmkernel: cpu2:1051751 opID=e3749acb)ScsiDevice: 1838:
Device t10.NVMe____INTEL_SSDPE2KX040T8_____BTLJ85110CKX4P0DGN__00000001
has been turned on administratively.
```

To format namespaces:

```
/vmfs/volumes/share/VMware/Tools/NVMe*cli/NVMe*-cli intel-NVMe*0 format 1
```

5.4 Saving Core Dump Example

```
partedUtil setptbl
/vmfs/devices/disks/t10.NVMe*____INTEL_SSDPEDMD400G4_CVFT534200N4400BGN__00000001 gpt "1
128 5000000 9D27538040AD11DBBF97000C2911D1B8 0"
# esxcli system coredump partition set --
partition=t10.NVMe*____INTEL_SSDPEDMD400G4_CVFT534200N4400BGN__00000001:1
# esxcli system coredump partition set --enable=true
# esxcli system coredump partition list
# vsish -e set /reliability/crashMe/Panic 1
# esxcfg-dumppart --copy –devname
/vmfs/devices/disks/t10.NVMe*____INTEL_SSDPECME016T4_CVF85504008C1P6BGN2D100000001:9
```

§§

6 Injecting Outbox ESXi* VROC Driver into ESXi* Bootable Image

6.1 Tools Needed

Download the Windows* *VMware.PowerCLI* tool from VMware*:
https://www.powershellgallery.com/packages/VMware.PowerCLI/11.4.0.14413515

6.2 Software Needed and Injection Procedure Summary

The VMware* ESXi driver installation package should include a .zip depot file. The following is used for this example:

- **VMware-ESXi-7.0U3d-19482537-depot.zip**

1. Log in to a vCenter Server or ESX host:

   Connect-VIServer (may have trust certificate issue)
   Connect-VIServer -server 10.2.61.32

Note: Do a **Get-PowerCLIConfiguration**. If it says **unset**, do a **Set-PowerCLIConfiguration -InvalidCertificateAction Ignore -Confirm:$false**.

Note: To find out what commands are available, type: **Get-VICommand**.

Note: To show searchable help for all PowerCLI commands, type: **Get-PowerCLIHelp**.

2. Once you have connected, display all virtual machines by typing: **Get-VM**.
3. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Add-EsxSoftwareDepot VMware-ESXi-7.0U3d-19482537-depot.zip**
4. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Get-ESXImageProfile**

```
Name                           Vendor          Last Modified           Acceptance Level
ESXi-7.0U3d-19482537-standard  VMware, Inc.*   3/29/2022               PartnerSupported
ESXi-7.0U3sd-19482531-no-tools VMware, Inc.*   3/11/2022               PartnerSupported
ESXi-7.0U3sd-19482531-standard VMware, Inc.*   3/29/2022               PartnerSupported
ESXi-7.0U3d-19482537-no-tools  VMware, Inc.*   3/11/2022               PartnerSupported
```

5. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> New-EsxImageProfile -CloneProfile "ESXi-7.0U3d-19482537-standard" -Name "profile" -Vendor "Intel"**

```
Name                           Vendor          Last Modified           Acceptance Level
profile                        Intel           3/29/2022               PartnerSupported
```

6. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Remove-EsxSoftwarePackage -ImageProfile profile -SoftwarePackage iavmd**

Name	Vendor	Last Modified	Acceptance Level
profile	Intel	6/22/2022	PartnerSupported

7. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Get-EsxImageProfile -Name profile | Select-Object -ExpandProperty Viblist | Sort-Object**

Name	Version	Vendor	Creation Date
esxio-combiner	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
esx-ui	1.36.0-18403931	VMware*	7/30/2021 6:2...
esx-update	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
esx-xserver	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
gc	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
i40en	1.11.1.31-1vmw.703.0.20.191...	VMware*	1/11/2022 11:...
icen	1.4.1.7-1vmw.703.0.20.19193900	VMware*	1/11/2022 11:...

8. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Get-EsxSoftwarePackage -PackageUrl iavmd-3.0.0.1034-1OEM.700.1.0.15843807.x86_64.vib**

Name	Version	Vendor	Creation Date
iavmd	3.0.0.1034-1OEM.700.1.0.158	Intel	4/2/2022 4:02...

9. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Add-EsxSoftwarePackage -ImageProfile profile -SoftwarePackage iavmd**

Name	Vendor	Last Modified	Acceptance Level
profile	Intel	6/22/2022	PartnerSupported

10. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Get-EsxImageProfile -Name profile | Select-Object -ExpandProperty Viblist | Sort-Object**

Name	Version	Vendor	Creation Date
esxio-combiner	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
esx-ui	1.36.0-18403931	VMware*	7/30/2021 6:2...
esx-update	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
esx-xserver	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
gc	7.0.3-0.35.19482537	VMware*	3/11/2022 2:1...
i40en	1.11.1.31-1vmw.703.0.20.191...	VMware*	1/11/2022 11:...
iavmd	**3.0.0.1034-1OEM.700.1.0.158...**	**Intel®**	**4/2/2022 4:02...**
icen	1.4.1.7-1vmw.703.0.20.19193900	VMware*	1/11/2022 11:...

11. Run the command: **C:\Program Files (x86)\VMware\Infrastructure\vSphere PowerCLI> Export-EsxImageProfile -ImageProfile "profile" -ExportToIso -FilePath ESXI_7.0.u3d_19482537_vmd_3.0.0.1034.iso -NoSignatureCheck**

7 Appendix

7.1 Expanded Intel® VROC and LED Management Tool Command Output Options

The following are provided to show available listings when using the Intel® VROC and LED Management Tool for managing Intel® VROC RAID volumes in an ESXi* environment.

Command: `intel-vmdr-cli disklist`

Disk state options are:

`CREATE`	disk is properly created, but is not functional yet
`PASSTHROUGH`	disk is passthru drive (not part of a RAID volume) and available to OS
`MEMBER`	disk is a member of RAID volume, and is hidden from OS
`SPARE`	disk is a spare drive, and is hidden from OS
`INCOMPATIBLE`	disk is incompatible due to license
`RAID UNSUPPORTED`	disk is not supported by the driver (e.g. block size)
`RAID Migration Target`	disk is a migration target
`UNKNOWN`	disk is in an unknown state

Command: `intel-vmdr-cli volumelist`

RAID volume state options are:

`CREATE`	volume is properly created and has disks assigned to it, but not yet functional
`NORMAL`	volume is functional, and all member disks are in a good condition
`DEGRADED`	volume is degraded, which means that one or more disks has failed, and RAID redundancy is compromised
`FAILED`	volume is failed, which means that it cannot be used to read from or write to
`STOPPED`	volume is stopped, which means that it is not currently operational

7.2 Custom Development of User Space Tools to Send NVMe Admin Commands to NVMe SSDs Behind Intel® VMD

The Intel VROC® VMD management interface supports the ability to receive NVMe Admin commands through its NVMe management interface layer and send these commands directly to NVMe SSDs behind Intel® VMD. Intel has implemented this feature using Broadcom's Daemon Software Development Kit (DSDK), access via the link https://tap.broadcom.com/docs/-/sdks/Daemon-Software-Development-Kit-DSDK/9.0.0.0.

The DSDK is intended to assist partners with developing custom utilities.

Note: To obtain access to the above link, contact your Broadcom account representative to enable access to the development tools available via the link.

To access iavmd's NVMe management interface, it is recommended to follow the VMware documentation for implementing user space tools.

The following code snippet represents the iavmd driver signature for management callback. For a user level program to interface with the iavmd driver's callback interface to send NVMe Admin opcodes, the function **`nvme_MgmtAdminCmds`** must be accessed in the user space tool. The second function in this code snippet **`nvme_MgmtRescan`** does not need to be implemented, but the signature is required.

Code Snippet 1

```
static vmk_MgmtCallbackInfo nvmeMgmtCbs[] = {
   {
      .location      = VMK_MGMT_CALLBACK_KERNEL,
      .callback      = nvme_MgmtAdminCmds,
      .synchronous   = 1,
      .numParms      = 2,
      .parmSizes     = {sizeof(NvmeAdminCmd), sizeof(vmk_uint32)},
      .parmTypes     = {VMK_MGMT_PARMTYPE_IN, VMK_MGMT_PARMTYPE_OUT},
      .callbackId    = NVME_MGMT_CALLBACK_ID,
   },
   {
      .location      = VMK_MGMT_CALLBACK_KERNEL,
      .callback      = nvme_MgmtRescan,
      .synchronous   = 1,
      .numParms      = 2,
      .parmSizes     = {sizeof(vmk_uint32), sizeof(vmk_uint32)},
      .parmTypes     = {VMK_MGMT_PARMTYPE_IN, VMK_MGMT_PARMTYPE_IN},
      .callbackId    = NVME_MGMT_CALLBACK_ID + 1,
   },
};
```

The following code snippet is provided to aid in development by demonstrating what is happening in the iavmd driver when the function **`nvme_MgmtAdminCmds`** is called.

Code Snippet 2

```
VMK_ReturnStatus nvme_MgmtAdminCmds(vmk_MgmtCookies *cookies,
      vmk_MgmtEnvelope *envelope, NvmeAdminCmd *cmd, vmk_uint32 *cmd_status)
{
   NvmeCommonCommand common;
   VMK_ReturnStatus status;
   vmk_IOA ioa = 0;
   int result;
   vmk_uint8 *response = NULL;
   NvmeAdapter *adapter = (NvmeAdapter *)cookies->handleCookie;
   vmk_uint64 cReset;

   nvme_Log(NVME_DEBUG, "%s: adapter:%p\n", __func__, adapter);

   cReset = vmk_AtomicRead64(&adapter->CtrlResetInProgress);
   if(cReset)
      return VMK_BUSY;

   if(!isAdapterValid(adapter))
      return VMK_FAILURE;

   if(cmd == NULL)
      return VMK_FAILURE;

   response = nvme_AllocDma(adapter, 0x1000, &ioa);
   if(!response)
      return VMK_NO_MEMORY;

   vmk_Memset(&common, 0x0, sizeof(common));
   common.opcode = cmd->opcode;
   common.flags = cmd->flags;
   common.nsid = cmd->nsid;
   common.cdw2[0] = cmd->cdw2;
   common.cdw2[1] = cmd->cdw3;
   common.cdw10[0] = cmd->cdw10;
   common.cdw10[1] = cmd->cdw11;
   common.cdw10[2] = cmd->cdw12;
   common.cdw10[3] = cmd->cdw13;
   common.cdw10[4] = cmd->cdw14;
   common.cdw10[5] = cmd->cdw15;
   common.prp1 = ioa;

   if (cmd->data_len) {
      if(cmd->data_len > 0x1000){
         nvme_FreeDma(adapter, 0x1000, ioa, response);
         return VMK_BAD_PARAM;
      }
      if(cmd->opcode & 1) {
         if (envelope)
            vmk_CopyFromUser((vmk_VA)response, (vmk_VA)cmd->addr, cmd->data_len);
         else
            vmk_Memcpy((void*)response, (void*)cmd->addr, cmd->data_len);
      }
   }
```

```
    if(!cmd->timeout_ms)
       cmd->timeout_ms = 10000;

    status = nvme_SubmitSyncRequest(adapter->adminQueue, (NvmeCommand *)&common,
                  &result);
    if(status != VMK_OK)
       result = -1;

    if (cmd->data_len) {
       if ((status == VMK_OK) && !(cmd->opcode & 1)) {
          if (envelope) {
             if (vmk_CopyToUser((vmk_VA)cmd->addr, (vmk_VA)response, cmd->data_len))
             status = VMK_FAILURE;
          } else {
             vmk_Memcpy((void*)cmd->addr, (void*)response, cmd->data_len);
          }
       }
    }
    nvme_FreeDma(adapter, 0x1000, ioa, response);
    vmk_Memcpy(cmd_status, &result, sizeof(vmk_uint32));
    return status;
}
```

§§