Network Appliance, Inc
OpenVMS Support
Guide to Best Practices

For OpenVMS on HP Alphaserver Platforms

Version 1.1
March 2007

This page is intentionally blank.

OpenVMS Support
Guide to Best Practices

NetApp®

Table of Contents

Table of Figures

This page is intentionally blank.

OpenVMS Support
Guide to Best Practices

NetApp®

1 Preface

1.1 Purpose

This document is intended as a guide for setting up and using the Network Appliance storage system to provide disk storage for the OpenVMS operating system in a Fibre Cchannel SAN environment.

1.2 Conventions

The following conventions are used in this guide:

Convention	Description
Ctrl/x	A sequence such as Ctrl/x indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
[Return]	In examples, a key name enclosed in brackets indicates that you press a key on the keyboard. (In text, a key name is not enclosed in a box.)
...	A horizontal ellipsis in examples indicates one of the following: • Additional optional arguments in a statement have been omitted. • The preceding item or items can be repeated one or more times. • Additional parameters, values, or other information can be entered.
• • •	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.
< >	Angle brackets indicate a placeholder for variable information in a command. The angle brackets should not be entered as part of the command.
( )	In command format descriptions, parentheses indicate that you must enclose choices in parentheses if you specify more than one.

Convention	Description
[ ]	In command format descriptions, brackets indicate optional choices. You can choose one or more items or no items. Do not type the brackets on the command line. However, you must include the brackets in the syntax for OpenVMS directory specifications and for a substring specification in an assignment statement.
|	In command format descriptions, vertical bars separate choices within brackets or braces. Within brackets, the choices are optional; within braces, at least one choice is required. Do not type the vertical bars on the command line.
{ }	In command format descriptions, braces indicate required choices; you must choose at least one of the items listed. Do not type the braces on the command line.
bold text	This typeface represents the introduction of a new term. It also represents the name of an argument, an attribute, or a reason.
italic text	Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (internal error number), in command lines (/PRODUCER=*name*), and in command parameters in text (where *dd* represents the predefined code for the device type).
bold italic	This typeface indicates a keyword or literal which must be entered exactly as shown.
UPPERCASE TEXT	Uppercase text indicates the name of a routine, the name of a file, or the abbreviation for a system privilege.
-	A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Non-decimal radixes—binary, octal, or hexadecimal—are explicitly indicated.

1.3 Definition of terms

This document contains terms which may have different meanings depending on the context in which they are used. For example, volume has different meanings when used from the NetApp appliance perspective and when used from the OpenVMS perspective. This section will define those terms and the context for which they are used.

Term	Definition
Console	Universally refers to a serial connection port giving direct, control access to a device (i.e.: NetApp appliance, host server, Fibre Channel switch, etc).
Filer	The NetApp storage appliance running the Data ONTAP software. OpenVMS documentation often refers to this as the "Disk Storage Controller", however in this context "NetApp Storage" will be used, as this storage appliance is the target of this documentation.
LUN	Logical Unit Number. Always used in the context of the NetApp storage appliance. OpenVMS uses a "Vendor Unique" Identifier (UDID) associated with the LUN for its device naming.
Volume	This term will always be used from the OpenVMS context. Although volume has significant meaning to Data ONTAP, this term is most frequently used by OpenVMS to mean any disk storage device initialized as either an ODS-2 or ODS-5 OpenVMS file system.
Disk	Often used to describe the physical hardware device upon which physical storage arrays are based. From an OpenVMS context, this term is interchangeable with *volume* and denotes the basic storage unit on which an ODS-2 or ODS-5 file system has been placed. **In this document, the disk is understood to be a storage entity (LUN presented by NetApp storage), identified to OpenVMS by its UDID, and referenced on the filer by the LUN.** The OpenVMS context will be used throughout this document.
WWNN, WWPN, and WWID	The industry standard conventions for uniquely identifying Fibre Channel entities. WWID (World Wide ID) is the term most frequently used in the OpenVMS environment to specify the World Wide Node Name or World Wide Port Name of a Fibre Channel device.
gbs	A fibre channel port speed in gigabits per second.

Term	Definition
cluster	When used in reference to the NetApp storage appliance, indicates the pair of filers operating cooperatively as one storage control unit. When used in the OpenVMS context, it refers to two (2) or more OpenVMS hosts sharing resources, and functioning together to appear as one system.

1.4 Document Sections

This document is separated into the following sections:
Section 1 (this section) – Document preface
Section 2 – NetApp Storage Appliance prerequisites and setup
Section 3 – Storage LUN provisioning
Section 4 – OpenVMS Host setup
Section 5 – SAN fabric zoning
Section 6 – Host Multipathing control
Section 7 – Troubleshooting

2 NetApp storage appliance prerequisites and setup

This section presents the prerequisites and proper setup of the Network Appliance storage system.

- Any NetApp filer capable of supporting fibre-channel protocol in a Fibre Channel fabric
- Fibre channel speeds of 1gbs, 2gbs, and 4gbs, depending on filer and fibre channel switch models
- Data ONTAP 7.2.1P1D9 and later
- Single Image cluster failover mode (SSI)

2.1 Supported NetApp filers

OpenVMS storage may be provisioned on any NetApp filer capable of supporting fibre-channel protocol in a Fibre Channel fabric. (Use of Fibre Channel Arbitrated Loops and Direct Connect to communicate between host initiators and storage targets is not supported for OpenVMS). Fibre channel speeds of 1gbs, 2gbs, and 4gbs are supported depending on filer and fibre channel switch models.

See the Fibre Channel Configuration Guide at NOW: Fibre Channel Configuration Guide

Note: OpenVMS running on HP Alphaserver models support only 1gbs and 2gbs fibre channel HBA links. For SAN fabric configurations where the NetApp filer fibre channel port speed does not match the host speed, the differential is handled by the Fibre Channel switch.

2.2 NetApp software requirements

Support for OpenVMS was first introduced in Data ONTAP version 7.2.1. Testing of this Data ONTAP release revealed a bug which has been fixed in a subsequent patch release. This patch release is Data ONTAP version 7.2.1P1D9, and can be found on the Network Appliance "NOW" web site. The bug-fix has been incorporated into the code streams of the future releases of "gordonbiersch" (Data ONTAP version 7.2.2) and "ironcity" (Data ONTAP 7.3.x).

- Data ONTAP 7.2.1P1D9 and later

2.3 Data ONTAP cluster failover mode

NetApp filers running the appropriate version of Data ONTAP must be configured for Single Image cluster failover mode (SSI). Although OpenVMS disk storage can be presented while in other cluster failover modes, multipathing capabilities will be severely hampered should a mode be used which does not present all of the fibre channel ports as being active (ACTIVE/ACTIVE) when the OpenVMS host server first discovers its disk storage devices.

This page is intentionally blank.

3 Storage LUN provisioning

This section describes hot to create storage for the OpenVMS environment.

3.1 Aggregate and FlexVol requirements

The underlying storage aggregate on each NetApp filer **head** is created using the standard Data ONTAP command for creating data storage aggregates. NetApp recommends that the aggregate be created using a raid type of "raid_dp" (RAID 4 Double Parity) for additional data protection. If an already existing aggregate is to be used, ensure it is of sufficient size to accommodate the FlexVol volume(s) that hold the provisioned LUNs.

FlexVol volumes that hold OpenVMS LUNs must be created with a space reservation policy that guarantees that the provisioned LUNs will be 100% writable by the OpenVMS operating system. Allow for the proper amount of space necessary if LUN snapshots are to be taken.

Should a portion of a LUN become unwritable due to space reservation reasons and drop offline, the OpenVMS operating system will consider this to be a device error. This may have detrimental effects on applications using this storage. If this were to happen to a SAN booted system disk, then a system hang or complete operating system failure could occur.

3.2 Create LUNs

The OpenVMS operating system does not identify its disk volumes using only the underlying LUN ID presented by the storage sub-system. Since OpenVMS multipathing pre-existed SCSI standard unique identifiers, a special Vendor Unique ID was chosen as the method to identify each LUN as a unique entity. The Unique ID, also called the User Defined Identifier (UDID), is embedded in the LUN definition.

The SCSI "Report Device Identifier" command (0xA3) and a SCSI "Vendor Unique" command (0xEC) are used to retrieve this identifier. UDID values of 1-9999 are used as identifiers. Values of 10000-32767 are possible but should not be used for the OpenVMS operating system (unexpected operation of those devices may result). SCSI command (0xA3) is used primarily by the operating system to fetch the UDID value. SCSI command (0xEC) is used by the Alphaserver bios to fetch this same value when setting up for SAN boot or operating system Crash Dump files located on SAN storage.

When creating or modifying LUNs for OpenVMS storage provisioning, you must complete the following steps.

Note: The number of steps required to define or modify an OpenVMS LUN may be reduced in future releases of the Data ONTAP software.

3.2.1 Place filer in diagnostice mode

Use the ***priv set diag*** command to place the filer into diagnostic mode. This allows access to the "dev_id" attribute associated with the LUN and the Vendor Unique Identifier (UDID) stored as that attribute. The command syntax is:

filer> ***priv set diag*** **[return]**

For example:

```
ntapfiler13> priv set diag
Warning: These diagnostic commands are for use by Network Appliance
        personnel only.
ntapfiler13*>
```

Figure 3-1 Privileged Diagnostic Mode

Once you are in this mode, you will have access to the special attributes associated with the LUN you wish to modify.

3.2.2 Create LUN

Use the standard Data ONTAP ***lun create*** command. You may also use the LUN setup procedure, if desired. The ***lun create*** command has the following syntax:

lun create -s <**size**> ***-t*** <**type**> [***-o noreserve***] <**lun_path**> **[return]**

- <**size**> must be large enough to accommodate the storage required by the OpenVMS host.
- <**type**> must be "***solaris***". This requirement may change in the future, should a specific LUN type be designated for the OpenVMS operating system. For now, the ***solaris*** LUN type has been designated for OpenVMS use.

NetApp strongly recommends that the option –o noreserve not be taken. (See the explanation about space reservation in Section 3.1).

3.2.3 Supplying Vendor Unique ID (UDID)

Supply or change the Vendor Unique ID (UDID) by executing the steps below:

1. Take the created (or existing) LUN offline.
   - Issue the ***lun offline*** <**lun_path**> command.
2. Modify the LUN attribute "dev_id" to the desired value.
   - Issue the ***lun attribute set*** <**lun_path**> ***dev_id*** <**value**> command.
3. Placing the modified LUN back online.
   - Issue the ***lun online*** <**lun_path**> command.

Note: The filer must be in diagnostic mode to access the attribute.

The following example assigns the value of 25 to the LUN attribute “dev_id”:

```
ntapfiler13*> lun offline /vol/ovms1/u25
Thu Feb 8 09:27:16 EST [ntapfiler13: lun.offline:warning]: lun /vol/ovms1/u25
has been taken offline
ntapfiler13*> lun attribute set /vol/ovms1/u25 dev_id 25
ntapfiler13*> lun online /vol/ovms1/u25
```

Figure 3-2 Set the "dev_id" attribute

3.3 Create IGROUPS and map LUNS for OpenVMS.

You must create an IGROUP on each filer head that is serving LUNs to the OpenVMS operating system.

It is through the IGROUP that provisioned LUNs are presented to the OpenVMS operating system. Each IGROUP will contain the WWPN port names (initiators) from all of the OpenVMS host servers accessing the storage. OpenVMS is not sensitive to multiple initiators present in the same fabric zone, so no special hard zoning or separate IGROUPS need be created. All OpenVMS cluster initiator ports can be contained in the same IGROUP.

3.3.1 LUN 0 (zero) – Special case

OpenVMS uses LUN 0 (zero) as a “Command Console LUN” (CCL). The operating system queries the storage sub-system through this CCL for all information about all of the other provisioned LUNs. Due to this activity, the following rules should be observed:

- You must provide one (1) minimally sized “dummy” LUN mapped to LUN 0 on each filer pair (not each filer head). Since this LUN is never written, the WAFL in Data ONTAP will never actually use any disk space for this LUN (other than minimal overhead), therefore the “-o noreserve” option may be used when creating this LUN. (An attempt to map a second LUN to LUN number 0 will result in an error when the NetApp filers are in Single Instance (SSI) cluster failover mode).
- A Vendor Unique Identifier (UDID) should not be assigned to this “dummy” LUN.
- This “dummy” LUN must be mapped as LUN 0 in the IGROUP on the filer head where it is defined.

3.3.2 Mapping Data LUNs

All provisioned data LUNs can be mapped to the filer pair IGROUP(s) using values between 1 and 4095. Any LUNs mapped to values greater than 255 will not be visible to the bios routines on the Alphaserver, but are visible to the OpenVMS operating system. Any LUNs which are intended to be used for SAN booting or Crash Dump disks must be mapped in the range from 1 to 255. As per Data ONTAP rules, only one occurrence of a particular LUN number can be mapped to an IGROUP across the filer pair (this is governed by the rule that all IGROUPs containing the same initiators are considered to be the same IGROUP). A warning will be displayed if you try to duplicate a LUN mapping to the same LUN number across the filer pair (while using Single Instance (SSI) cluster failover mode).

3.3.3 Obtaining the Initiator WWPN port names.

The initiator WWPN port names required to create the OpenVMS IGROUP(s) can be obtained from the Alphaserver hosts. The procedures to obtain this data are presented in Section 4.

3.3.4 LUNs with duplicate UDID values

NetApp storage appliances make it possible to present LUNs to many separate hosts and host clusters. This is also true for OpenVMS. Because of this, it is possible to have LUNs provisioned for separate entities which coincidentally have the same UDID values. The following guidelines prevent accidental conflicts which might arise due to these duplications:

- No OpenVMS cluster (or standalone system) can have more than one (1) disk device with the same name. Hence, only one (1) UDID of a particular value can be presented to that cluster (or standalone host). (See section 4.2).
- OpenVMS clusters (or standalone hosts) must always be segregated by separate IGROUP(s). Initiators from other clusters (or standalone hosts) must not overlap.
- If OpenVMS clusters (or standalone hosts) must occasionally have access to the same disk resources, then those LUNs should be mapped to the multiple IGROUP(s) as needed. Again, do not violate the rule about duplicate UDID values in the same IGROUP.

4 Set up OpenVMS host

This section addresses setting up the OpenVMS host servers to access NetApp storage, the Alphaserver bios, SAN booting, and Crash Dump storage. If your site is not using NetApp provisioned LUNs for SAN booting or Crash Dumps, proceed to Section 4.2.

Note: Interactions with the host server bios commands can only occur while the operating system is down.

4.1 Alphaserver bios, SAN boot, and Crash Dump disks

Before you can SAN boot the operating system from a NetApp provisioned LUN, you must first setup the Alphaserver bios to recognize the LUN from which you will boot. The LUN must contain a valid OpenVMS system disk volume. Booting the operating system from the host server bios level requires that one (1) or more valid paths to the LUN be known by the host server. This path information is stored in special environment variables (saved in non-volatile ram) on the host. The actual path information is unique from server to server as it is based upon which PCI-X slot the HBA (Host Bus Adapter) cards are located in. (i.e.: you cannot just copy this environment data to another server and expect it to work). There is a special bios utility program called WWIDMGR which can be used to discover the paths to the LUNs you wish to boot from. The User's Manual for this utility can be found at WWIDMGR User Manual.

4.1.1 Initialize host server

Prior to using the WWIDMGR utility, you must initialize the host server using the following command:

P00>>> ***init*** **[return]**
Initializing…

•

•

•

(device and bus initialization info displayed)
P00>>>

You can display some basic information about the HBA cards installed by using the command (as shown in this example):

```
P00>>> show device pg
pga0.0.0.8.1          PGA0          WWN 1000-0000-c939-1b27
pgb0.0.0.8.0          PGB0          WWN 1000-0000-c934-064a
P00>>>
```

Figure 4-1 Alphaserver bios Show Device

Note: The WWN format is displayed as hyphenated groups of four (4) characters instead of the NetApp convention of colon separated groups of two (2) characters.

Once initialized, the host is in a state ready to process WWIDMGR commands. All WWIDMGR commands begin with the literal ***wwidmgr***.

4.1.2 Get HBA status

Issue the following WWIDMGR command to show the status of the HBAs installed:

```
P00>>> wwidmgr -show adapter
item adapter                WWN                    Cur. Topo Next Topo
[ 0] pga0.0.0.8.1           2000-0000-c939-1b27    FABRIC    FABRIC
[ 1] pgb0.0.0.8.0           2000-0000-c934-064a    FABRIC    FABRIC
[9999] All of the above.
P00>>>
```

Figure 4-2 WWIDMGR Show Adapter

The Cur Topo and Next Topo columns should display FABRIC. If not, then use the following command to properly set your HBA cards to use FABRIC topology.

Note: OpenVMS does not support fibre channel arbitrated loops (FCAL) or point to point (PTP) to communicate between initiators and targets:

P00>>> ***wwidmgr –set adapter –item*** <**item_number**> ***-topo fabric*** **[return]**

- **<item_number>** is the HBA adapter of interest from the **wwidmgr –show adapter** command, or 9999 for all of the adapters.

You must re-initialize the server for this change to take effect.

4.1.3 Discover available LUNs

To have the Alphaserver bios discover the LUNs available for SAN booting, use the following command:

P00>>> ***wwidmgr –show wwid*** **[return]**

Output from this command may look similar to this example:

```
[0] UDID:21 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-3069 (ev:wwid0)
[1] UDID:-1 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-3032 (ev:none)
[2] UDID:22 WWID:01000010:60a9-8000-486e-5338-575a-2d34-5737-7152 (ev:none)
[3] UDID:23 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-314d (ev:none)
[4] UDID:24 WWID:01000010:60a9-8000-486e-5338-575a-2d34-5737-7241 (ev:none)
[5] UDID:26 WWID:01000010:60a9-8000-486e-5338-575a-2d34-5737-7271 (ev:none)
```

Figure 4-3 WWIDMGR Show WWID

In this example, five (5) available LUNs are displayed.

Note: The number appearing within brackets ([ ]) in the left column is a menu item number and not the LUN.

Example Item [1] shows a UDID value of -1 (minus one). This is normal for any LUNs discovered which do not have a UDID assigned. (In this case, this is LUN 0, the "dummy LUN").

Of interest is the information in parenthesis indicating whether the path is currently stored in an environmental variable (in non-volatile ram). Item [0], (UDID: 21), in this example shows its path being stored in environmental variable wwid0. The others indicate that their paths are not stored.

For SAN booting, the path must be stored in an environmental variable. Up to four (4) wwid environmental variables (wwid0 to wwid3) are defined in the server bios. There are also eight (8) environmental variables named N1 through N8 which contain WWPN path names to the target ports where LUNs have been discovered.

4.1.4 Clear environmental variables

To clear any or all of the environmental variables (wwid and N), use the following command (Section 3.5 of the WWIDMGR Manual):

P00>>> ***wwidmgr –clear*** <**item_to_clear**> **[return]**

- <**item_to_clear**> can be any of the following: **wwid0, wwid1, wwid2, wwid3**, **N1** to **N8**, or **all**.

4.1.5 Select LUNs for use

To select the desired LUN for SAN booting, use the "quickset" feature in WWIDMGR. (Section 2.4 of the WWIDMGR Manual).

You should not use the "quickset" feature if the following conditions exist:

- Multiple SAN boot disks on different NetApp filer pairs.
- Crash Dump disk on a filer pair other than the SAN boot disk.

Consult the WWIDMGR User's Manual for procedures where the LUN devices do not share a common path (Section 3.6 of the WWIDMGR Manual).

To use the "quickset" feature, issue the following command:

P00>>> ***wwidmgr –quickset -udid*** <**udid_value**> **[return]**

The example below demonstrates the use of the "quickset" feature. It is based on five (5) LUNs being presented by a NetApp filer pair:

First show the available LUNs:

```
P00>>> wwidmgr -show wwid
[0] UDID:21 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-3069 (ev:none)
[1] UDID:-1 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-3032 (ev:none)
[2] UDID:22 WWID:01000010:60a9-8000-486e-5338-575a-2d34-5737-7152 (ev:none)
[3] UDID:23 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-314d (ev:none)
[4] UDID:24 WWID:01000010:60a9-8000-486e-5338-575a-2d34-5737-7241 (ev:none)
[5] UDID:25 WWID:01000010:60a9-8000-486e-5336-675a-2d34-5736-3238 (ev:none)
P00>>>
```

Figure 4-4 Show WWID prior to -quickset

Next, select the desired LUN using the UDID value (in this example UDID 21 is used):

```
P00>>> wwidmgr -quickset -udid 21

Disk assignment and reach ability after next initialization:

60a9-8000-486e-5336-675a-2d34-5736-3069
via adapter: via fc nport: connected:
dga21.1001.0.8.1 pga0.0.0.8.1 500a-0985-8719-2bd4 Yes
dga21.1002.0.8.1 pga0.0.0.8.1 500a-0985-9719-2bd4 Yes
dgb21.1003.0.8.0 pgb0.0.0.8.0 500a-0986-9719-2bd4 Yes
dgb21.1004.0.8.0 pgb0.0.0.8.0 500a-0986-8719-2bd4 Yes
P00>>>
```

Figure 4-5 WWIDMGR -quickset

Warning: There is a variant of this command where a menu item can be selected instead of the UDID value. This variant should not be used and is not supported. This is because it establishes a path based upon a non-OpenVMS compatible unit identification value. Attempts to boot from this path will result in an OpenVMS BUGCHECK (crash).

To use the LUN device as selected, the Alphaserver must be re-initialized (you must do this anyway, after using WWIDMGR and prior to booting).

4.1.6 Check other boot-related environmental variables

Several other environmental variables should be checked and/or modified prior to booting the OpenVMS operating system. These variables are related to the OS booting function (and/or the dump disk access, if applicable). These variable names (and functions) are:

- **bootdef_dev** – This variable will contain one or more device name/paths pointing to the boot device. The devices are specified as a space separated list. Up to four (4) paths may be specified. To examine the variable, issue the command:

  P00>>> ***show bootdef_dev*** **[return]**

Here is an example showing two paths to the same boot device (UDID 21):

```
P00>>> show bootdef_dev
bootdef_dev                    dga21.1001.0.8.1 dgb21.1003.0.8.0
P00>>>
```

Figure 4-6 Show bootdef_dev

To set this variable (if not set correctly by the "quickset" feature), use the following command:

P00>>> ***set bootdef_dev*** <**path_list**>**[return]**

Here is an example of setting the bootdef_dev variable:

```
P00>>> set bootdef_dev dga21.1001.0.8.1 dgb21.1003.0.8.0
P00>>>
```

Figure 4-7 Set bootdef_dev

- **boot_osflags** – This variable contains a value pair which denotes the OpenVMS root path to boot from and whether the boot should be automatic or conversational. The first number in the coma-separated pair is the operating system root, and the second number is either zero (0) for automatic or one (1) for conversational. To examine this variable, issue the command:

  P00>>> ***show boot_osflags*** **[return]**

Here is an example showing flags which specify a boot from root 0 automatically (non-conversational).

```
P00>>> show boot_osflags
boot_osflags 0,0
P00>>>
```

Figure 4-8 Show boot_osflags

To set the boot_osflags, use the command:

P00>>> ***set boot_osflags*** <**os_root,conversation_flag**>**[return]**

Here is an example setting the operating system root to 0 and a conversational boot:

```
P00>>> set boot_osflags 0,1
P00>>>
```

Figure 4-9 Set boot_osflags

- **dump_dev** – This variable may contain one (1) or more paths to the dump device (only if a SAN dump device is used). The device paths are similar to the paths used in the ***bootdef_dev*** variable. To examine this variable, issue the command:

  P00>>> ***show dump_dev*****[return]**

An example of this command is:

```
P00>>> show dump_dev
dump_dev                        dga21.1001.0.8.1 dgb21.1003.0.8.0
P00>>>
```

Figure 4-10 Show dump_dev

To set one or more paths into the variable, issue the command:

P00>>> ***set dump_dev*** <**path_list**>**[return]**

An example of setting this variable to two (2) paths:

```
P00>>> set dump_dev dga21.1001.0.8.1 dgb21.1003.0.8.0
P00>>>
```

Figure 4-11 Set dump_dev

NetApp recommends that no more than two (2) paths be specified, each on a separate HBA port.

4.2 Storage devices from the OpenVMS perspective

The OpenVMS operating system has been in existence since the early 1980s. This operating system has a long history of "world-class" features, fault tolerance, extreme security, and unparalleled uptimes. Originally owned and developed by Digital Equipment Corporation, for their VAX family of computer platforms, this operating system migrated to the Alpha 64 bit architecture in the early 1990's. This operating system has recently been ported for use on the Intel Itanium64 platform. Hewlett Packard (HP) is the current owner and licensor of the OpenVMS operating system, having acquired it through its merger with Compaq (which purchased DEC).

Although much work has progressed over the years to keep storage solutions for OpenVMS current with today's technology, much of the operating system's native file system retains a structure derived from the earliest days of the operating system. This section will explore, in depth, how the OpenVMS host connects to, and uses mass (disk) storage (especially in the fibre channel SAN area.

4.2.1 OpenVMS device naming conventions

The OpenVMS operating system was designed to be clustered into groups of two (2) or more systems coupled, via hardware and software, in such a manner to allow the direct sharing of

multiple resources (mostly disk and tape storage). This design required a standard convention to uniquely identify each shared resource. This standard not only encompasses the hardware links used to interconnect the systems, but also how network communications should occur between OpenVMS systems whether clustered or not. Even if an OpenVMS site is not using DECnet as its primary networking protocol, a DECnet Phase IV node name must be defined for each OpenVMS host.

4.2.1.1 Host node names

This host name forms the basis for creating device names that are shared throughout the cluster environment. OpenVMS node names are alphanumeric strings of one (1) to six (6) characters (the letters A-Z and digits 0-9, beginning with an alphabetic letter). Each member of an OpenVMS cluster must have a unique name. Any directly attached device (i.e.: SCSI disks, SCSI attached tape drives, etc) that may be shared to other members of the cluster must refer to that device as **nodename$device_id**, where **nodename** is the host to which the device is attached and **device_id** is the physical device in **ddcu:** name format. The **ddcu:** format is described below.

The following example of a device shows SCSI disk device DKA0: on node FRACK:

FRACK$DKA0

4.2.1.2 Device classes, controllers, and units

The **ddcu:** device name format is based on a coding method where the **dd** portion of the name identifies the device class (i.e.: SCSI disk, fibre channel disk, SCSI tape, network adapter, terminal device, etc…). Typical device classes include the following (this not an alliinclusive list):

- DK – SCSI disk
- DG – Fibre Channel disk
- DV – Floppy drive
- DQ – CD-ROM drive
- TK – SCSI tape
- GK – SCSI media robot
- FG – Fibre Channel HBA
- PG – Fibre Channel Port
- OP – System or Operator consoles
- EW and EI – Ethernet network adapters
- TX, TT, RT, FT – Terminal or Pseudo-Terminal devices
- BG – TCP or UDP network socket device

The **c** portion of the name identifies the associated controller driving the device. This will be identified by a letter (A-Z) which denotes the particular card and its placement in the host computer's bus (typically PCI-X in Alphaserver models). At the time of this writing, this format limits OpenVMS to the use of up to 26 adapters of a given class (for example, up to 26 Fibre Channel HBAs).

The **u** portion of the name specifies the associated port or unit number in the controller card. This value is usually zero (0), but may include values greater than zero (>0) if the device driver requires it (segmented protocols on a network adapter, for example).

Most devices with multiple physical port capabilities are presented to the operating system as separate devices. For example, the two (2) port network interface card (NIC) in the Alphaserver is seen as two devices (EIA0: and EIB0:).

4.2.1.3 Device allocation classes

On OpenVMS clusters, especially large clusters, it is easier to segregate devices by allocation class than by node names. To this end, the system generation parameter ALLOCLASS was created. When OpenVMS boots, and a non-zero value is assigned to the ALLOCLASS parameter, then that ALLOCLASS is used instead of the node name. Valid values for ALLOCLASS range from 0 – 255. A value of zero (0) means that no ALLOCLASS names will be used (using the node name instead). Values of 1 or greater (up to 255) will substitute the ALLOCLASS in the form of $alloclass$ddcu:. For example:

> Node FRACK has ALLOCLASS set to zero (0), the SCSI disk device DKA0: will appear and be shared as FRACK$DKA0:
>
> Node FRACK has ALLOCLASS set to five (5), the SCSI disk device DKA0: will appear and be shared as 5DKA0:

Allocation classes one (1 - disk) and two (2 - tape) are reserved for devices connected via Fibre Channel SANs. Regardless of how you define the ALLOCLASS parameters on your systems, the automatic assignment of ALLOWCLASS 1 and 2 will occur when external storage devices are fibre channel connected. Each OpenVMS node participating in a cluster should have a unique ALLOCLASS assigned. NetApp recommends avoiding the use of the values 1 or 2, to avoid confusion with external fibre channel storage.

4.2.1.4 Fibre channel disk device names.

All disk devices discovered on fibre channel connections are named in accordance with the following rules:

- The allocation class is **1.**
- The **ddc** portion of the device name is PGA.
- The **u** portion of the device name is a numeric value ranging from 1 to 9999 and is the "vendor unique" (UDID) value assigned to the LUN being presented by the storage appliance.

Following are typical Fibre Channel disk names:

> 1DGA1:
> 1DGA1000:

4.2.2 OpenVMS ODS and file system structure.

This section describes the disk file systems supported by the OpenVMS operating system.

4.2.2.1 ODS (On Disk Structure)

The disk file system on OpenVMS is proprietary and uses three (3) formats. The first format On Disk Structure 1 (ODS-1) is no longer actively used and is mentioned only for completeness in this documentation. ODS levels 2 and 5 are the current formats in use with ODS-2 being the most common. ODS-5 shares most of the same attributes as ODS-2, but supports case-sensitive file names and also supports the "long" file names found on many of today's UNIX, Linux, and Windows operating systems. ODS-3 and ODS-4 formats are used on CD-ROM based storage and are outside the scope of this documentation.

4.2.2.2 File system disk layout and directory structure.

OpenVMS disk volumes are laid out in a hierarchal fashion, as a tree starting with the Master File Directory (MFD). When a disk volume is first initialized (or later re-initialized), certain required files as well as the MFD get created. The MFD directory is a file named 000000.DIR (six zeros, a period delimiter, and the extension DIR which signifies that this file is a directory). As data and other directory trees are created on the volume, the linkage from the lowest points in the tree must eventually lead back to the MFD. Broken linkage may make certain portions of the disk structure inaccessible. This almost never happens unless "misguided" human intervention has occurred. The OpenVMS operating system provides certain commands which aid in the recovery of data/directory integrity should a problem occur. Refer to the proper OpenVMS documentation for further information on this subject.

4.2.2.3 OpenVMS file name syntax

Many OpenVMS DCL (Digital Command Language) commands require file names to be supplied as parameters. Following is a description of the OpenVMS file naming format and how it is syntactically applied. As the ODS-2 file system structure is the most prevalent format in use, all discussion and examples refer to that format. Any reference to ODS-5 conventions will be explicitly stated.

ODS-2 volumes share a common set of rules:

- All names are case-insensitive and are stored in upper case ("DOG" is the same as "dog", "dOg", etc).
- The file system supports multi-version files with up to 32767 versions of the same file name in any given directory.
- Filenames must contain the following components:
    - File name – (Up to 39 characters. May contain the letters A-Z, digits 0-9, and some special characters like $ _).
    - Extension separator – (The period character).
    - File extension – (Same rules as the file name).
    - Version separator – (The semi-colon character).
    - File version – (A numeric value from 1 through 32767).

Following is an example of a typical file name:

EMPLOYEES.DAT;1

4.2.2.4 Directory specifications and syntax

Navigating to specific directories to locate and use files requires the use of directory names prefixed before the file name. Directory specifications are only required if the desired file is not in the current default directory (PWD is the UNIX equivalent).

Directory tree hierarchy is expressed by the syntax **[**<**directory_name**>**]**. The enclosing brackets are a required part of the syntax. Multiple levels of directory traverse is expressed by separating each level by a period (i.e.: **[**<**directory_level1**>.<**directory_level2**>**]**, etc). As directory tree level nesting can become quite large, OpenVMS allows the creation of "rooted directories" (defined through logical names) to enable short-cuts to very long directory paths. This feature is very similar to file system "mount points" in UNIX. Consult the OpenVMS system documentation to learn more about directory navigation and "rooted directories".

4.2.2.5 Full file specifications

As noted in the previous sections above, the specification syntax for the various elements of the file system follows a hierarchical convention. This section documents the formal (full) file syntax that can be used to find/access any shared file in the OpenVMS environment.

The full file specification has the following syntax:

<**node_name_or_alloclass**>***$***<**ddcu**>**:/**<**directory_tree**>**/**<**filename**>.<**extension**>;<**version**>

- <**node_name_or_alloclass**> may be omitted if your current process is on the same node offering the shared resource
- **1** or **2** alloclass should always be specified if the device is presented from Fibre Channel SANs
- **<ddcu:>** device specification may be omitted if you access files in your current default directory. <**ddcu:**> may be replaced by a "rooted directory" logical name
- <**directory_tree**> specification may be omitted if the file is in the current default directory, or the directory hierarchy is defined within the "rooted directory" logical name
- portions of the directory name(s) may be wild carded (See OpenVMS documentation)
- <**filename**> is mandatory but may be wild carded (See OpenVMS documentation)
- <**extension**> may be omitted or wild carded. Certain applications will have default extensions. You must specify the extension to override the application defaults.
- <**version**> is optional. Version number specifications may be wild carded if you are operating on multiple versions of the same file name. The OpenVMS file system will always choose the file with the highest version number. If a lower version is desired, then the version specification must be explicit.

Below are some examples of full file specifications:

FRICK$DKA0:[SYS0.SYSEXE]SWAPFILE.SYS;1
(The file SWAPFILE.SYS;1 located in the directory tree [SYS0.SYSEXE] on a disk DKA0: on the OpenVMS node FRICK).

1DGA31:[TEST_TREE_01]TEST_FILE_0001.DAT;1
(The file TEST_FILE_0001.DAT;1 located in the directory [TEST_TREE_01] on a SAN disk 1DGA31).

4.3 Getting Initiator information from OpenVMS

This section describes methods which for discovering information about the initiator ports on the OpenVMS system.

4.3.1 Host Bus Adapter (HBA) card types

There are several HBA card models which may be used in the Alphaserver running OpenVMS. These cards must interface with the Alphaserver's PCI-X bus. These models are:

- HP KGPSA – An HP branded Emulex card type which may be purchased directly from HP or one of its resellers. (1gbs and 2gbs versions are available).
- HP FCA2354 (alternate name for LP9002)
- HP FCA2384 (alternate name for LP9802)
- HP FCA2684 (alternate name for LP10000)
- HP FCA2684DC (alternate name for dual port LP10000).
- Emulex LP8000 – A single port 1gbs PCI model
  Note: The Emulex LP9000 is not supported by HP in Alphaserver systems.
- Emulex LP9002 – A single port 2gbs model (used in our operating system qualifications).
- Emulex LP9802 – A single port 2gbs PCI-X model
- Emulex LP10000 – A single port 2gbs PCI-X model.
- Emulex LP10000-DC – a dual port 2gps PCI-X model

Note: Qlogic HBA models will not work on Alphaserver models. However, several Qlogic models are used on the HP Integrity (Itanium 64) servers running OpenVMS 8.3. (Itanium 64 servers do not use Emulex HBAs at this time.).

4.3.2 System Inquiry Utility with Fibre Channel extension

The utility used to query the running system for information about its fibre channel connections is called the System Dump Analyzer (SDA). This utility is a standard feature of all OpenVMS installations. This utility can both be used to analyze system crash dumps, as well as to obtain information about the currently running system. An extension to the SDA enables a user to display information about the fibre channel devices on the system. The user running the SDA must be privileged (this command is usually run from the SYSTEM account). To run the analyzer, enter the following command at the DCL ($) prompt:

$ ***ANALYZE/SYSTEM*[return]**

You will now be in a conversational mode with the SYSTEM DUMP ANALYZER (SDA), as indicated by the SDA> prompt.

Warning: The SDA tool can also cause damage to a running system – caution should be used when running SDA to stay within the confines of the commands documented here

4.3.2.1 Selecting which HBA device to interrogate

From the SDA> prompt, enter the following command to select the HBA you wish to interrogate:

Note: When first entering SDA, the first HBA card, FGA0: is selected.

SDA> ***FC SET DEVICE*** <**fc_device**>**[return]**
Where fc_device is the desired HBA port (FGA0:, FGB0:, etc…).

4.3.2.2 Displaying the HBA port initiator address (WWPN)

Once the desired HBA has been selected (See Section 4.3.2.1), you may issue the following command to view that device's WWPN initiator address. The syntax is:

SDA> FC ***SHOW DEVICE*** **[return]**

You can also directly specify the HBA device on the command line as follows:

SDA> *FC SHOW DEVICE* <**fc_device**>**[return]**

Below are examples of these command sequences:

```
SDA> FC SET DEVICE FGA0:
SDA> FC SHOW DEVICE
FGA0: operational firmware revision CS3.90A7
port_name(adapter_id) = 1000-0000-C93D-BE9F, node_name(host_id) = 2000-0000-C93D-BE9F
SDA>

SDA> FC SHOW DEVICE FGB0:
FGB0: operational firmware revision CS3.93A0
port_name(adapter_id) = 1000-0000-C941-2937, node_name(host_id) = 2000-0000-C941-2937
SDA>
```

Figure 4-12 SDA FC Set Device and FC Show Device

4.3.2.3 Displaying targets visible from an HBA port

All fibre channel fabric members need a way to be identified so that protocol exchanges can be directed (routed) correctly. This is accomplished by each node having a World Wide Node Name (WWNN), each port having a World Wide Port Name (WWPN), and each connection at the fabric switches having a unique (within the fabric) Fibre Channel ID (FCID).

To display storage target port WWPN(s) (and other initiators in same fabric zone), first select the desired HBA device (see Section 4.3.2.1) and then issue the following command:

SDA> ***FC NAME_LIST*** **[return]**

Below are examples showing the output from this command:

```
SDA> FC SET DEVICE FGA0:
SDA> FC NAME_LIST
FGA0: Name List
 Index    qfl      qbl      port name        node name       state  ale_index  rpi
 ----- -------- -------- ----------------- ----------------- ------- --------- ----
   0:  00000000 00000000 0000000000000000 0000000000000000 *******     0     0000
   1:  8158D4A0 8158D4A0 2007000DEC064A40 2004000DEC064A41  VALID      1     0001
   2:  8158D4E8 8158D4E8 248D000DEC064A40 2004000DEC064A41  VALID      2     0003
   3:  8158D530 8158D530 500A098183312BD8 500A098083312BD8  VALID      4     0005
   4:  8158D578 8158D578 500A098193312BD8 500A098083312BD8  VALID      5     0006
   5:  8158D5C0 8158D5C0 500A098597192BD4 500A098087192BD4  VALID      6     0007
   6:  8158D608 8158D608 10000000C9391B27 20000000C9391B27  VALID      3     0004
   7:  8158D650 8158D650 500A098587192BD4 500A098087192BD4  VALID      7     0008
SDA>

SDA> FC SET DEVICE FGB0:
SDA> FC NAME_LIST
FGB0: Name List
 Index    qfl      qbl      port name        node name       state  ale_index  rpi
 ----- -------- -------- ----------------- ----------------- ------- --------- ----
   0:  00000000 00000000 0000000000000000 0000000000000000 *******     0     0000
   1:  817D20E0 817D20E0 2008000DEC064A40 2004000DEC064A41  VALID      1     0001
   2:  817D2128 817D2128 248D000DEC064A40 2004000DEC064A41  VALID      2     0003
   3:  817D2170 817D2170 500A098283312BD8 500A098083312BD8  VALID      4     0005
   4:  817D21B8 817D21B8 500A098293312BD8 500A098083312BD8  VALID      5     0006
   5:  817D2200 817D2200 10000000C934064A 20000000C934064A  VALID      3     0004
   6:  817D2248 817D2248 500A098697192BD4 500A098087192BD4  VALID      6     0007
   7:  817D2290 817D2290 500A098687192BD4 500A098087192BD4  VALID      7     0008
SDA>
```

Figure 4-13 SDA FC NAME_LIST

To display storage target port FCIDs (addresses) (and other initiators in same fabric zone), first select the desired HBA device (see Section 4.3.2.1) and then issue the following command:

SDA> ***FC ADDRESS_LIST*** **[return]**

Below are examples showing the output from this command:

```
SDA> FC SET DEVICE FGA0:
SDA> FC ADDRESS_LIST
FGA0: Address List
 Index Address   State    Probe State NLE  PLE   IND     SEQ#   XID  OELS FLAGS
 ----- -------- ---------- ----------- ---- ---- -------- -------- ---- ---- -----
  0:  00000000                    0    0 00000000       0 0000    0 0000
  1:  00FFFFFE  VALID               1    0 00000000       0 0000    0 0301
  2:  00FFFFFC  VALID               2    0 00000000       0 0000    0 0301
  3:  005D0000  VALID   REG LOGIN     6    0 00000000       1 0000    0 0300
  4:  005D0005  VALID   REG LOGIN     3    0 00000000       0 0000    0 0200
  5:  005D0007  VALID   REG LOGIN     4    0 00000000       0 0000    0 0200
  6:  005D04DA  VALID   REG LOGIN     5    0 00000000       0 0000    0 0200
  7:  005D05DA  VALID   REG LOGIN     7    0 00000000       0 0000    0 0200
  8:  00FFFC5D  KNOWN               0    0 00000000       3 0000    0 0001
SDA>

SDA> FC SET DEVICE FGB0:
SDA> FC ADDRESS_LIST
FGB0: Address List
 Index Address   State    Probe State NLE  PLE   IND     SEQ#   XID  OELS FLAGS
 ----- -------- ---------- ----------- ---- ---- -------- -------- ---- ---- -----
  0:  00000000                    0    0 00000000       0 0000    0 0000
  1:  00FFFFFE  VALID               1    0 00000000       0 0000    0 0301
  2:  00FFFFFC  VALID               2    0 00000000       0 0000    0 0301
  3:  005D0001  VALID   REG LOGIN     5    0 00000000       1 0000    0 0300
  4:  005D0006  VALID   REG LOGIN     3    0 00000000       0 0000    0 0200
  5:  005D0008  VALID   REG LOGIN     4    0 00000000       0 0000    0 0200
  6:  005D03DA  VALID   REG LOGIN     6    0 00000000       0 0000    0 0200
  7:  005D06DA  VALID   REG LOGIN     7    0 00000000       0 0000    0 0200
  8:  00FFFC5D  KNOWN               0    0 00000000       3 0000    0 0001
SDA>
```

Figure 4-14 SDA FC ADDRESS_LIST

4.3.2.4 Relationship between the NAME_LIST and ADDRESS_LIST

Issuing SDA commands to examine the HBA device NAME_LIST data and the ADDRESS_LIST data are just two (2) ways of referring to the same fibre channel objects. There is a direct correlation as demonstrated in the following example.

This example examines the two different addressing pointers used by the HBA device FGA0:

```
SDA> FC SET DEVICE FGA0:
SDA> FC NAME_LIST
FGA0: Name List
 Index   qfl     qbl     port name       node name      state  ale_index  rpi
 ----- -------- -------- ---------------- ---------------- ------- --------- ----
   0:  00000000 00000000 0000000000000000 0000000000000000 *******     0     0000
   1:  8158D4A0 8158D4A0 2007000DEC064A40 2004000DEC064A41  VALID     1     0001
   2:  8158D4E8 8158D4E8 248D000DEC064A40 2004000DEC064A41  VALID     2     0003
   3:  8158D530 8158D530 500A098183312BD8 500A098083312BD8  VALID     4     0005
   4:  8158D578 8158D578 500A098193312BD8 500A098083312BD8  VALID     5     0006
   5:  8158D5C0 8158D5C0 500A098597192BD4 500A098087192BD4  VALID     6     0007
   6:  8158D608 8158D608 10000000C9391B27 20000000C9391B27  VALID     3     0004
   7:  8158D650 8158D650 500A098587192BD4 500A098087192BD4  VALID     7     0008
SDA> FC ADDRESS_LIST
FGA0: Address List
 Index Address    State    Probe State  NLE PLE    IND     SEQ#   XID  OELS FLAGS
 ----- -------- ---------- ----------- ---- ---- -------- -------- ---- ---- -----
   0:  00000000                    0    0 00000000       0 0000    0 0000
   1:  00FFFFFE   VALID               1    0 00000000       0 0000    0 0301
   2:  00FFFFFC   VALID               2    0 00000000       0 0000    0 0301
   3:  005D0000   VALID    REG LOGIN     6    0 00000000       1 0000    0 0300
   4:  005D0005   VALID    REG LOGIN     3    0 00000000       0 0000    0 0200
   5:  005D0007   VALID    REG LOGIN     4    0 00000000       0 0000    0 0200
   6:  005D04DA   VALID    REG LOGIN     5    0 00000000       0 0000    0 0200
   7:  005D05DA   VALID    REG LOGIN     7    0 00000000       0 0000    0 0200
   8:  00FFFC5D   KNOWN                 0    0 00000000       3 0000    0 0001
SDA>
```

Figure 4-15 NAME_LIST - ADDRESS_LIST Relationship

In order to route fibre channel protocol messages correctly, the fabric name servers used in the fibre channel fabric switches must be able to translate the WWPN(s) provided by the HBA ports to an internal fabric shorthand FCID used by the switch ports for fast switching. The cross-referencing of these translations can be seen in the outputs shown above. For any WWPN listed in the NAME_LIST, the value in the column entitled “ale_index” points to the row intersection in the ADDRESS_LIST where the “index” value matches. There is a reverse translation (correlation) where the “nle” column value from the ADDRESS_LIST matches the “index” value in the NAME_LIST.

4.3.2.5 Other SDA FC Commands

There are other SDA FC commands, which are primarily used by HP server and support personnel to help diagnose fibre channel problems. These commands are not covered here.

This page is intentionally blank.

5 Fabric Zoning in the OpenVMS Environment

This section covers topics concerning how fabric zoning should be defined for storage access in the OpenVMS environment. The number of fabrics used can range from the simple single fabric, to more complex multiple fabrics. Dual fabrics are usually implemented for fiber channel connection redundancy and fault tolerance. The number of fabrics and the zoning within them will have a direct impact on the multipathing access to LUN storage.

5.1 Single versus dual fabric topology

Each topology has widespread use in datacenters, today. The choice of which topology to use is based upon a number of factors. Some of these are:

Single Fabric

- Simplicity
- Lower infrastructure costs
- Redundancy not required
- Hosts not running "mission-critical" applications

Dual Fabric

- High availability
- Fault tolerance
- Higher bandwidth
- "Mission-critical" applications on hosts

NetApp recommends that at least two (2) fabrics be used when connecting OpenVMS hosts to fibre channel storage.

The following sub-sections describe (via narrative and diagram) the differences between using a single fabric versus using dual fabrics. The examples provided are intended to illustrate recommended, industry standard practices for fibre channel interconnects. The actual topology used is determined by the requirements of each individual site.

5.1.1 Single fabric

Figure 5-1 below shows a FASxxxx storage system attached to a single fabric. The fabric may consist of one or multiple fibre channel switches and, in fact, the storage controller may be attached to multiple switches. The FASxxxx storage system may have anywhere from four to eight connections to the fabric. Because of the non-redundant nature of the single fabric, this configuration cannot be considered to be fully redundant. When a host has multiple paths to a single LUN configured on a storage controller, it is necessary for the host to have multipathing software installed (integral to OpenVMS). In this diagram, all hosts would have multiple paths unless the fabric is zoned to prevent this. A host with a single fibre channel connection can still have multiple logical paths to the LUN if there are multiple connections from the NetApp storage controllers to the fabric. If it is not desired to have multiple paths to the LUNs, see the section on fibre channel zoning to learn how to eliminate the redundant paths or Section 6 on OpenVMS multipathing control.

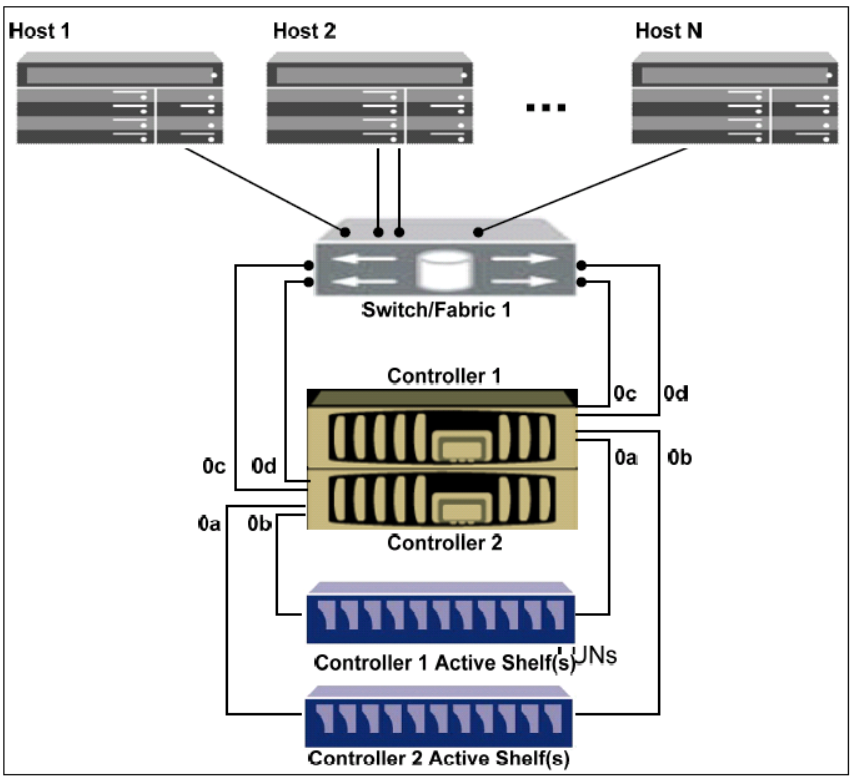

Figure 5-1 Single Fabric

5.1.2 Dual fabric

Figure 5-2 below shows a dual-fabric-attached configuration for the FASxxxx storage appliance where four fibre channel target ports are used on each storage controller. When properly configured with multipathing software (integral to OpenVMS), the hosts are fully redundant from the storage perspective since the HBA, wiring, fabrics, storage controller and disks are all redundantly configured. For multiple-host configurations, the hosts may be heterogeneous (i.e. OpenVMS, Windows, Linux and Unix) since each storage port is capable of customizing responses based on what type of operating system is accessing it.

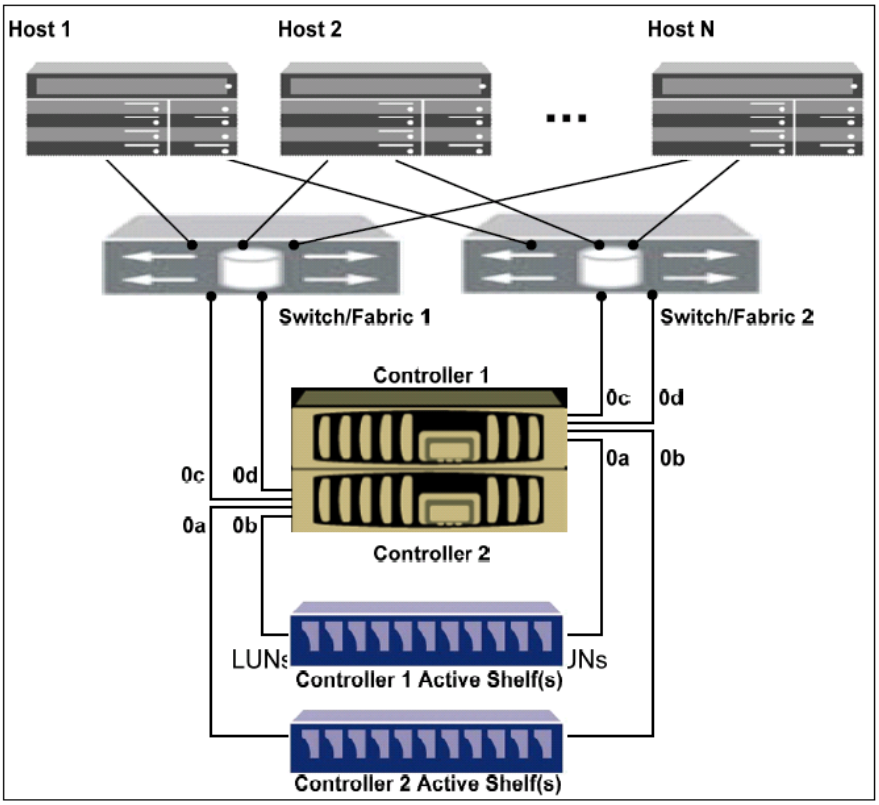

Figure 5-2 Dual Fabric

5.2 Switch zoning

How zoning will be defined depends on which families of fabric switches are used. OpenVMS SAN connections are supported on various switch models from Brocade, Cisco, and McData. These switches may be purchased from HP (under HP branding), HP resellers, or directly from the switch manufacturers. Consult the Network Appliance Matrix for supported models.

5.2.1 Selecting storage target ports for each zone

NetApp recommends distributing the target ports on each filer head to the zones in such a way that in the event of a port failure, other paths to the same filer head are still available. (See figure 5-2 as an example of connecting storage target ports to the multiple fabrics).

5.2.2 Connecting host initiator ports to the fabrics

In the recommended configuration of multiple fabrics, at least one HBA on each host should be connected to each fabric. It is a common practice on Alphaserver hosts to place the HBA cards on separate PCI busses to minimize the chance of failure. For example, the HBA for PGA0 devices would be on BUS 0 while the HBA for PGB0 devices be placed on BUS 1. NetApp recommends populating HBAs in pairs, so in a four (4) HBA configuration, place two (2) HBAs on PCI BUS 0 and the remaining two (2) HBAs on PCI BUS 1. In this case, try to configure one (1) HBA from each PCI bus to each fabric.

When setting up the zoning on the respective fabrics, NetApp recommends that all of the storage target ports in that fabric appear in each configured OpenVMS zone. Failure to follow this guideline may prevent all valid data paths from being visible and/or used by the attached hosts. Limiting the paths by making some of the targets unavailable to each initiator serves no advantage for the OpenVMS operating system. Better control of the multipathing can be configured at the individual hosts (see Section 6 for more details). As OpenVMS hosts are not affected by other initiators, you may include all of the initiators in the same zone. You may, of course, use separate zones for smaller groups of initiators (or individual initiators) as long as you remember to include the proper target ports.

Note: NetApp does not recommend reconfiguring the physical switch cabling or the zoning while the OpenVMS operating system is running.

6 OpenVMS Multipathing Control

The previous sections have described how to configure the storage appliances and hosts for connections in the OpenVMS environment. This section discusses, in detail, how to manage the multipathing, resulting from these configurations.

6.1 Preferred versus Non-preferred paths

After fibre channel SAN connections are configured, the OpenVMS operating system can discover all possible paths to any presented LUN. The discovery process occurs during system boot and may also occur on the running system when the SYSMAN utility IO AUTOCONFIGURE command is executed. The actual discovery mechanism is integral to the operating system kernel.

The paths to each device are discovered in the order by which the LUNs are presented on each discovered target port as seen by each HBA adapter port. That is, device paths seen by HBA device FGA0: are first, FGB0: next, and so on. As OpenVMS has no knowledge about efficiency of any given path when it is first discovered, the initial "current" path to any particular disk device may not always be the one that the OpenVMS system manager wishes to use for later IO.

When the requirements for the storage sub-system are first determined, much consideration should be placed on how to distribute the provisioned LUNs over the NetApp filer pairs. Some factors to consider are:

- storage size
- read/write IO loading
- proximity to other LUNs with any system/application inter-relationships
- storage usage by other Operating Systems
- fabric topology and zoning

Once the disks are laid out, the system manager can see from the OpenVMS disk device information which paths come directly from the filer head where the LUN is defined (owned), and which paths follow an indirect route through the partner filer head (proxy access through the vtic). The target port WWPN(s) are easily determined via some basic Data ONTAP commands issued to each filer head. Proxy access will render less than optimal performance and should be avoided. The methods for viewing path information and setting the desired path are covered later in this section.

6.2 Number of paths to a device

The number of paths discovered to any device is a simple multiplication of the number of HBA ports present on the host server times the number of target ports presenting the provisioned LUNs (HBA_PORT_COUNT x TARGET PORT_COUNT). Zoning within the fibre channel fabric will affect this number if some HBA initiators cannot "see" all of the possible storage target ports.

Additional paths to disk devices may be possible through MSCP (Mass Storage Control Protocol) connections in an OpenVMS cluster. There will be N-1 additional paths based on N cluster hosts serving disks via this process.

6.3 Automatic path switching

The OpenVMS operating system, by default, polls the fibre channel devices to determine the health and usability of each and every connection. This polling should not be turned off, as the ability to determine when a path has failed is dependent on this polling service. During IO operations the operating system will attempt to use the "current" path as long as it continues to be available or remains unchanged by system management. OpenVMS does not currently have any load balancing algorithms that allow multiple concurrent use of all available paths to any given disk device.

When the status of the IO path changes from available to unavailable, OpenVMS performs the following steps:

- Place the affected disk volume in a "Mount Verification" status which temporarily suspends IO operations on it.
- Search for the next available path and switch to it if one is found. This switch is permanent unless this path also fails and another switch occurs.
- Take the volume out of the "Mount Verification" status.
- Resume IO operations on the disk.

If no viable paths are found, the disk will remain in "Mount Verifications" and alerts are sent to the appropriate console and logging devices. Operator remediation will be required.

In an OpenVMS cluster, the following additional step may occur.

- If no available fibre channel paths are available, and if MSCP disk serving has been enabled on one or more other cluster member systems, then the path will be switched to the MSCP service and will remain on this path until the fibre channel path has been restored and has remained operational for a certain period of time (configurable at system generation time).

Note: MSCP (Mass Storage Control Protocol) allows an OpenVMS host to access to a mass storage device (i.e.: disk, tape, etc) without a direct attachment. OpenVMS cluster members with direct access to a mass storage device can function as a device server for other cluster members, who do not have this direct attachment. This service uses the host-cluster interconnect medium (i.e.: Ethernet, Memory Channel, FDDI, etc) as its transport. As this storage serving protocol is carried on the same channels as other cluster communications, performance is not as robust as a direct attachment. Use of MSCP as a permanent path to storage comes with a performance penalty and is discouraged.

6.4 Path switching during filer failovers

If a NetApp filer head fails over to its partner, there will be a brief time (the time span during which the failover occurs) when any active OpenVMS volume affected by the failure will be placed in "Mount Verification" status. This is normal behavior and even the booted system disk can be in this status. As soon as the failover completes, the OpenVMS host should see available paths to the disk

devices it was using and continue to operate in a normal manner. Automatic path switching will occur, as necessary, allowing OpenVMS to access the disk devices. A similar period of "Mount Verification" will occur during filer giveback/recovery. Most applications on the OpenVMS host, using the native RMS (Record Management Services) to access disk volume data files, are immune to these brief suspensions of IO.

Some other system utilities, however, will fail if path switching occurs while they are engaged in IO. Primarily, the OpenVMS backup utility will fail when engaged in block IO operations typical of IMAGE or PHYSICAL backups (file oriented backups will not be impacted). NetApp recommends reviewing all system activities when the storage system fails, and rerunning any IMAGE/PHYSICAL backup jobs if you are not 100% certain of their integrity.

6.5 Displaying and setting disk device paths

This section describes the methods used to display or change the paths to an OpenVMS disk volume.

6.5.1 About OpenVMS DCL commands

Several commands may be used to display the current or other possible paths to an OpenVMS disk volume. These commands are issued by the user from the DCL (Digital Command Language) command line prompt (usually denoted by the $). Commands to view device path information may be issued by anyone; however, commands to modify disk volume paths require that the user have sufficient privilege to do so.

6.5.1.1 OpenVMS DCL command syntax

Most OpenVMS DCL commands are clear and concise. The general syntax contains:

- A **command name**
- Zero or more required **parameters** in a space separated list
- Zero or more **command qualifiers**, each starting with a slash (/) character

The command name need not be spelled out completely. Only enough characters are required to prevent ambiguity are required (i.e.: SHOW and SHO both activate the SHOW command). This same rule applies to any optional **command qualifiers**. Most **parameters** must be spelled out in their entirety.

6.5.1.2 Positional and non-positional command qualifiers

For some OpenVMS DCL commands, the additional command qualifiers are said to be positional it they relate only to one parameter in particular. These qualifiers must immediately follow the parameter for which they apply. Other general command qualifiers may appear anywhere in the command line (as long as they do not interfere with any positional qualifiers which may be present). Refer to the DCL Command Reference, included in the OpenVMS documentation set, for the correct syntax of any OpenVMS DCL command.

6.5.2 Displaying the paths to an OpenVMS disk volume

All path information for any OpenVMS disk volume can be displayed by using the ***SHOW DEVICES*** command (with the proper qualifiers). Two (2) variations of this command are used for this purpose. The basic command syntax is:

$ ***SHOW DEVICES*** [***/MULTIPATH*** | ***/FULL***] [<**device_name**>] **[return]**

The optional **device_name** parameter displays information for that device only. Otherwise all devices will be displayed. An abbreviated device name may be given to display information about all members of a particular device class (i.e.: DG will display all **1DGAxxxx** devices on the system).

The ***/MULTIPATH*** qualifier shows the device's current path. This qualifier selects only devices capable of multipath IO.

The ***/FULL*** qualifier displays all possible paths to the device, including the status of each path.

Below are examples of the ***SHOW DEVICES*** command:

```
$ SHOW DEVICES $1$DGA21:

Device             Device        Error    Volume       Free  Trans Mnt
 Name              Status        Count    Label      Blocks Count Cnt
$1$DGA21:     (FRICK)  Mounted          2  OVMSAXP732    3365527   446   2
$
```

Figure 6-1 Basic DCL SHOW DEVICES command

This basic command shows brief information about the disk device 1DGA21:.

```
$ SHOW DEVICES/MULTIPATH $1$DGA21:
Device             Device        Error       Current
 Name              Status        Count  Paths    path
$1$DGA21:     (FRICK)  Mounted          2   5/ 5  PGA0.500A-0985-9719-2BD4
$
```

Figure 6-2 DCL SHOW DEVICES /MULTIPATH command

This command shows the same device, in this case displaying the current path. This path string is composed of the following items:

- HBA device (in this example, PGA0).
- A dot (.) separator.
- The WWPN of the target from which the path is derived (in this example **500A-0985-9719-2BD4).**

```
$ SHOW DEVICES/FULL $1$DGA21:

Disk $1$DGA21: (FRICK), device type NETAPP LUN, is online, mounted, file-
```

```
  oriented device, shareable, device has multiple I/O paths, served to cluster
  via MSCP Server, error logging is enabled.

  Error count               2    Operations completed          846806
  Owner process             ""   Owner UIC                  [SYSTEM]
  Owner process ID      00000000   Dev Prot          S:RWPL,O:RWPL,G:R,W
  Reference count            385   Default buffer size             512
  Current preferred CPU Id     0   Fastpath                          1
  WWID   01000010:60A9-8000-486E-5336-675A-2D34-5736-3069
  Total blocks           18874368   Sectors per track                32
  Total cylinders           18432   Tracks per cylinder              32
  Logical Volume Size    18874368   Expansion Size Limit         19144704
  Host name              "FRICK"    Host type, avail COMPAQ AlphaServer DS20E 666
  Alternate host name    "FRACK"    Alt. type, avail  AlphaServer ES40, yes
  Allocation class           1

  Volume label       "OVMSAXP732"   Relative volume number            0
  Cluster size              19    Transaction count               446
  Free blocks            3365527   Maximum files allowed          471859
  Extend quantity              5   Mount count                       2
  Mount status            System   Cache name       "_$1$DGA21:XQPCACHE"
  Extent cache size           64    Maximum blocks in extent cache   336552
  File ID cache size          64    Blocks in extent cache           333317
  Quota cache size             0    Maximum buffers in FCP cache       2877
  Volume owner UIC          [1,1]   Vol Prot    S:RWCD,O:RWCD,G:RWCD,W:RWCD

 Volume Status:  ODS-2, subject to mount verification, protected subsystems
    enabled, write-through caching enabled.
 Volume is also mounted on FRACK.

 I/O paths to device           5
 Path PGA0.500A-0985-9719-2BD4  (FRICK), primary path, current path.
   Error count                 1    Operations completed          801249
 Path PGA0.500A-0985-8719-2BD4  (FRICK).
   Error count                 0    Operations completed           15948
 Path PGB0.500A-0986-9719-2BD4  (FRICK).
   Error count                 0    Operations completed           14747
 Path PGB0.500A-0986-8719-2BD4  (FRICK).
   Error count                 1    Operations completed           14862
 Path MSCP  (FRACK).
   Error count                 0    Operations completed               0

$
```

Figure 6-3 DCL SHOW DEVICES /FULL command

This example shows the DCL ***SHOW DEVICES/FULL*** command for disk 1DGA21:. Of particular interest is the section showing all of the paths to the device. This example shows four (4) fibre channel paths plus an MSCP path served by another OpenVMS cluster member (FRACK). The current path is noted in the display.

6.5.3 Setting or changing the path to a device

Paths to a device may be changed by issuing the DCL ***SET DEVICE*** command. The syntax for this command is:

$ ***SET DEVICE/PATH*** =<**path_string**>***/SWITCH*** **[return]**

The value for **path_string** must be a valid path containing the following path components:

- HBA device (i.e.: PGA0, PGB0, etc…).
- A dot (.) separator.
- WWPN of the target from which the path is derived (i.e.: **500A-0985-9719-2BD4**).

The command qualifier ***/SWITCH*** requests that the ***SET DEVICE*** command to try and switch to the path, if possible. There is no guarantee that the switch will occur, only that an attempt will be made.

An example of this command is:

```
$ SET DEVICE/PATH=PGA0.500A-0985-8719-2BD4/SWITCH $1$DGA21:
$
```

Figure 6-4 DCL SET DEVICE /PATH /SWITCH command

You can verify that the switch has occurred by issuing a ***SHOW DEVICES*** command to examine the current path.

6.5.4 Enabling or disabling a path to a device

You can enable or disable a path to a device by issuing the SET DEVICE command, specifying the path, the /ENABLE or /NOENABLE qualifier, and the device name. The syntax is:

$ ***SET DEVICE/PATH***=<**path_string**> { ***/ENABLE*** | ***/NOENABLE*** }<**device_name**> **[return]**

Note: Issuing this command while trying to disable the current path will result in an error.

Below is an example of disabling a path:

```
$ SET DEVICE/PATH=PGB0.500A-0986-9719-2BD4/NOENABLE $1$DGA21:
$
$ SHOW DEVICES/FULL $1$DGA21:

Disk $1$DGA21: (FRICK), device type NETAPP LUN, is online, mounted, file-
   oriented device, shareable, device has multiple I/O paths, served to cluster
   via MSCP Server, error logging is enabled.
.
.
.
  I/O paths to device           5
  Path PGA0.500A-0985-9719-2BD4  (FRICK), primary path.
    Error count                 1    Operations completed          806326
  Path PGA0.500A-0985-8719-2BD4  (FRICK), current path.
    Error count                 0    Operations completed          17194
  Path PGB0.500A-0986-9719-2BD4  (FRICK), user disabled.
    Error count                 0    Operations completed          14862
  Path PGB0.500A-0986-8719-2BD4  (FRICK).
    Error count                 1    Operations completed          14978
  Path MSCP  (FRACK).
    Error count                 0    Operations completed              0

$
```

Figure 6-5 SET DEVICE/PATH/NOENABLE

This page is intentionally blank.

7 Troubleshooting problems

This section presents some basic troubleshooting suggestions.

Problem	Troubleshooting Steps and Possible Solutions
LUNs not visible to OpenVMS hosts.	Check for the following: • Fibre Channel cables connected properly. HBA indicators showing proper link lights. • Host HBA initiator WWPN(s) defined as members of the IGROUP(s) on the NetApp filers. (See Section 3.3). • "Dummy" LUN 0 mapped to an IGROUP on the NetApp filer pair. (See Section 3.3.1). • Correct zoning on all switches in the fabric(s). (See Section 5.2). • OpenVMS required UDID(s) applied as "dev_id" attributes on the provisioned LUNs. (See Section 3.2.3). • LUNs mapped properly in IGROUP. (See Section 3.3.2). • LUNs with duplicate UDID values not mapped to same IGROUP. (See Section 3.3.4). Take corrective action as necessary.
New LUNs added while OpenVMS is running and do not appear.	Check for the following: • OpenVMS required UDID(s) applied as "dev_id" attributes on the provisioned LUNs. (See Section 3.2.3). • LUNs mapped properly in IGROUP. (See Section 3.3.2). • Ensure that LUNs with duplicate UDID values are not mapped to the same IGROUP used by the OpenVMS cluster (or standalone host). (See Section 3.3.4). Take corrective action as necessary. Try Issuing a SYSMAN utility IO AUTOCONFIGURE command.
OpenVMS sees all of the LUNs but hangs when trying to access them.	Verify that you are running Data ONTAP version 7.2.1P1D9 or a later version with the patch correcting the bug reported in BURT #225166. (See Section 2.2).

Problem	Troubleshooting Steps and Possible Solutions
OpenVMS indicates that the disk media for any of its LUNs is offline.	The space reservation for the LUNs contained within a FlexVol flexible volume is not sufficient or has been disabled. Make sure that space reservations are enabled on the volume, and that there is sufficient allocated space to contain 100% of the LUN's space requirement. (See Section 3.1).
Some OpenVMS disks become inaccessible after a NetApp filer failover occurs.	This situation may occur for the following reasons: • The filer pair is not configured for SSI (Single Image Cluster failover mode). The original paths to the "failover" target ports were not available due to the ports being in stand-by mode when the OpenVMS host did its disk discovery. (See Section 2.3). • The fabric switches are zoned in a manner whereby the target ports now serving the failed partner's LUNs are not visible to the OpenVMS host initiators. (See Section 5). Take corrective action as is necessary.
Configuring the Alphaserver bios to learn LUN paths and 2 or more LUNs have the same UDID value.	It is possible to have more than one LUN configured with the same UDID provided that (See Section 3.3.4): • Each duplicate UDID is targeted for a separate OpenVMS cluster or standalone host. • The LUNs containing these duplicate UDID(s) are mapped to separate IGROUP(s) which do not have overlapping initiators and are intended for separate OpenVMS clusters or standalone hosts. Either correct the duplication, or ensure that the LUNs are mapped to separate IGROUP(s). (See above).