vmware PRESS

Storage Implementation in vSphere 5.0

TECHNOLOGY DEEP DIVE

Mostafa Khalil

Storage Implementation in vSphere® 5.0

VMware Press is the official publisher of VMware books and training materials, which provide guidance on the critical topics facing today's technology professionals and students. Enterprises, as well as small- and medium-sized organizations, adopt virtualization as a more agile way of scaling IT to meet business needs. VMware Press provides proven, technically accurate information that will help them meet their goals for customizing, building, and maintaining their virtual environment.

With books, certification, study guides, video training, and learning tools produced by world-class architects and IT experts, VMware Press helps IT professionals master a diverse range of topics on virtualization and cloud computing and is the official source of reference materials for preparing for the VMware Certified Professional Examination.

VMware Press is also pleased to have localization partners that can publish its products into more than 42 languages, including, but not limited to, Chinese (Simplified), Chinese (Traditional), French, German, Greek, Hindi, Japanese, Korean, Polish, Russian, and Spanish.

For more information about VMware Press, please visit
http://www.**vmware.com/go/vmwarepress**.

Storage Implementation in vSphere® 5.0

TECHNOLOGY DEEP DIVE

Mostafa Khalil, VCDX

vmware® PRESS

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

STORAGE IMPLEMENTATION IN VSPHERE® 5.0

Published by VMware, Inc.

Publishing as VMware Press

ISBN-10: 0-321-79993-3

ISBN-10: 978-0-321-79993-7

Library of Congress Cataloging-in-Publication data is on file.

Printed in the United States of America

First Printing: August 2012

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an “as is” basis. The authors, VMware Press, VMware, and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the CD or programs accompanying it.

The opinions expressed in this book belong to the author and are not necessarily those of VMware.

Corporate and Government Sales

VMware Press offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales
international@pearsoned.com

VMWARE PRESS PROGRAM MANAGER
Erik Ullanderson

ASSOCIATE PUBLISHER
David Dusthimer

EDITOR
Joan Murray

DEVELOPMENT EDITOR
Ellie Bru

MANAGING EDITOR
Sandra Schroeder

PROJECT EDITOR
Seth Kerney

COPY EDITOR
Charlotte Kughen

PROOFREADER
Megan Wade

EDITORIAL ASSISTANT
Vanessa Evans

BOOK DESIGNER
Gary Adair

COMPOSITOR
Studio Galou, LLC.

To my wife Gloria for her unconditional love and tireless efforts in helping make the time to complete this book.

Contents At A Glance

Contents

Preface

This first edition of *Storage Implementation in vSphere 5.0* is my first attempt to put all the practical experience I have acquired over the years supporting VMware products and drinking from the fountain of knowledge that is the VMware team. I share with you in-depth details of how things work so that you can identify problems if and when anything goes wrong. I originally planned to put everything in one book, but as I started writing the page count kept growing, partly due to the large number of illustrations and screenshots that I hope will make the picture clearer for you. As a result, I had to split this book into two volumes so that I don't have to sacrifice quality at the expense of page count. I hope you will find this content as useful as I intended it to be and that you'll watch for the second volume, which is coming down the pike.

The book starts with a brief introduction to the history of storage as I experienced it. It then provides details of the various storage connectivity choices and protocols supported by VMware: Fibre Channel (FC), Fibre Channel over Ethernet (FCoE), and Internet Small Computer System Interface (iSCSI). This transitions us to the foundation of vSphere storage, which is Pluggable Storage Architecture (PSA). From there I build upon this foundation with multipathing and failover (including third-party offerings) and ALUA. I then discuss storage virtual devices (SVDs) and VMDirectPath I/O architecture, implementation, and configuration. I also cover in intricate details Virtual Machine File System (VMFS) versions 3 and 5 and how this highly advanced clustered file system arbitrates concurrent access to virtual machine files as well as raw device mappings. I discuss the details of how distributed locks are handled as well as physical snapshots and virtual machines snapshots. Finally, I share with you vStorage APIs for Array Integration (VAAI) architecture and interactions with the relevant storage arrays.

Consider this volume as the first installment of more advanced content to come. I plan to update the content to vSphere 5.1, which will bear the name of *VMware Cloud Infrastructure Suite (CIS)*, and add more information geared toward design topics and performance optimization.

I would love to hear your opinions or suggestions for topics to cover. You can leave me a comment at my blog: http://vSphereStorage.com.

Thank you and God bless!

Mostafa Khalil, VCDX

Acknowledgments

I would like to acknowledge the endless support I got from my wife Gloria. I would also like to acknowledge the encouragement I got from Scot Bajtos, Senior VP of VMware Global Support Services, and Eric Wansong, VP of VMware Global Support Services (Americas).

I truly appreciate the feedback from those who took time out of their busy schedules to volunteer to review parts of the books:

Craig Risinger, Consulting Architect at VMware

Mike Panas, Senior Member of Technical Staff at VMware

Aboubacar Diar, HP Storage

Vaughn Stewart, NetApp

Jonathan Van Meter

A special thanks to Cormac Hogan, Senior Technical Marketing Architect at VMware, for permitting me to use some of his illustrations.

I also would like to acknowledge Pearson's technical reviewers, whom I knew only by their initials, and my editors Joan Murray and Ellie Bru for staying after me to get this book completed.

One last acknowledgement is to all who have taught and mentored me along the way throughout my journey. Their names are too many to count. You know who you are. Thank you all!

About the Author

Mostafa Khalil is a senior staff engineer at VMware. He is a senior member of VMware Global Support Services and has worked for VMware for more than 13 years. Prior to joining VMware, he worked at Lotus/IBM. A native of Egypt, Mostafa graduated from the Al-Azhar University's School of Medicine, and practiced medicine in Cairo. He became intrigued by the mini computer system used in his medical practice and began to educate himself about computing and networking technologies. After moving to the United States, Mostafa continued to focus on computing and acquired several professional certifications.

He is certified as VCDX (3, 4, & 5), VCAP (4 & 5)-DCD, VCAP4-DCA, VCP (2, 3, 4, & 5), MCSE, Master CNE, HP ASE, IBM CSE, and Lotus CLP.

As storage became a central element in the virtualization environment, Mostafa became an expert in this field and delivered several seminars and troubleshooting workshops at various VMware public events in the United States and around the world.

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

As an associate publisher for Pearson, I welcome your comments. You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books better.

Please note that I cannot help you with technical problems related to the topic of this book. We do have a User Services group, however, where I will forward specific technical questions related to the book.

When you write, please be sure to include this book's title and author as well as your name, email address, and phone number. I will carefully review your comments and share them with the author and editors who worked on the book.

Email: VMwarePress@vmware.com

Mail: David Dusthimer
Associate Publisher
Pearson
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Visit our website at www.informit.com/title/9780321799937 and register this book for convenient access to any updates, downloads, or errata that might be available for this book.

Chapter 5

vSphere Pluggable Storage Architecture (PSA)

vSphere 5.0 continues to utilize the Pluggable Storage Architecture (PSA) which was introduced with ESX 3.5. The move to this architecture modularizes the storage stack, which makes it easier to maintain and to open the doors for storage partners to develop their own proprietary components that plug into this architecture.

Availability is critical, so redundant paths to storage are essential. One of the key functions of the storage component in vSphere is to provide multipathing (if there are multiple paths, which path should a given I/O use) and failover (when a path goes down, I/O failovers to using another path).

VMware, by default, provides a generic Multipathing Plugin (MPP) called Native Multipathing (NMP).

Native Multipathing

To understand how the pieces of PSA fit together, Figures 5.1, 5.2, 5.4, and 5.6 build up the PSA gradually.

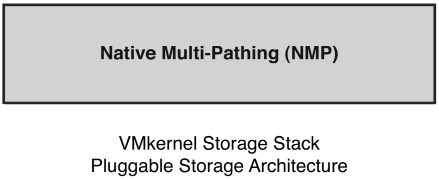


Figure 5.1 Native MPP

NMP is the component of vSphere 5 vmkernel that handles multipathing and failover. It exports two APIs: Storage Array Type Plugin (SATP) and Path Selection Plugin (PSP), which are implemented as plug-ins.

NMP performs the following functions (some done with help from SATPs and PSPs):

- Registers logical devices with the PSA framework
- Receives input/output (I/O) requests for logical devices it registered with the PSA framework
- Completes the I/Os and posts completion of the SCSI command block with the PSA framework, which includes the following operations:
  - Selects the physical path to which it sends the I/O requests
  - Handles failure conditions encountered by the I/O requests
- Handles task management operations—for example, Aborts/Resets

PSA communicates with NMP for the following operations:

- Open/close logical devices.
- Start I/O to logical devices.
- Abort an I/O to logical devices.
- Get the name of the physical paths to logical devices.
- Get the SCSI inquiry information for logical devices.

Storage Array Type Plug-in (SATP)

Figure 5.2 depicts the relationship between SATP and NMP.

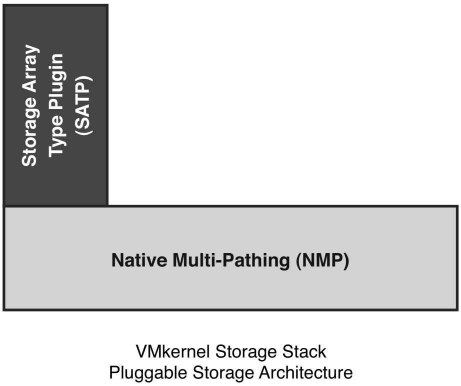


Figure 5.2 SATP

SATPs are PSA plug-ins specific to certain storage arrays or storage array families. Some are generic for certain array classes—for example, Active/Passive, Active/Active, or ALUA-capable arrays.

SATPs handle the following operations:

- Monitor the hardware state of the physical paths to the storage array
- Determine when a hardware component of a physical path has failed
- Switch physical paths to the array when a path has failed

NMP communicates with SATPs for the following operations:

- Set up a new logical device—claim a physical path
- Update the hardware states of the physical paths (for example, Active, Standby, Dead)
- Activate the standby physical paths of an active/passive array (when Active paths state is dead or unavailable)
- Notify the plug-in that an I/O is about to be issued on a given path
- Analyze the cause of an I/O failure on a given path (based on errors returned by the array)

Examples of SATPs are listed in Table 5.1:

Table 5.1 Examples of SATPs

SATP	Description
VMW_SATP_CX	Supports EMC CX that do not use the ALUA protocol
VMW_SATP_ALUA_CX	Supports EMC CX that use the ALUA protocol
VMW_SATP_SYMM	Supports EMC Symmetrix array family
VMW_SATP_INV	Supports EMC Invista array family
VMW_SATP_EVA	Supports HP EVA arrays
VMW_SATP_MSA	Supports HP MSA arrays
VMW_SATP_EQL	Supports Dell Equalogic arrays
VMW_SATP_SVC	Supports IBM SVC arrays
VMW_SATP_LSI	Supports LSI arrays and others OEMed from it (for example, DS4000 family)
VMW_SATP_ALUA	Supports non-specific arrays that support ALUA protocol
VMW_SATP_DEFAULT_AA	Supports non-specific active/active arrays
VMW_SATP_DEFAULT_AP	Supports non-specific active/passive arrays
VMW_SATP_LOCAL	Supports direct attached devices

How to List SATPs on an ESXi 5 Host

To obtain a list of SATPs on a given ESXi 5 host, you may run the following command directly on the host or remotely via an SSH session, a vMA appliance, or ESXCLI:

```
# esxcli storage nmp satp list
```

An example of the output is shown in Figure 5.3.

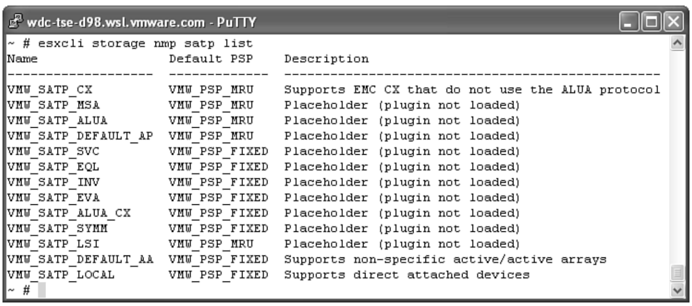


Figure 5.3 Listing SATPs

Notice that each SATP is listed in association with a specific PSP. The output shows the default configuration of a freshly installed ESXi 5 host. To modify these associations, refer to the "Modifying PSA Plug-in Configurations Using the UI" section later in this chapter.

If you installed third-party SATPs, they are listed along with the SATPs shown in Table 5.1.

> **NOTE**
> ESXi 5 only loads the SATPs matching detected storage arrays based on the corresponding claim rules. See the "Claim Rules" section later in this chapter for more about claim rules. Otherwise, you see them listed as (Plugin not loaded) similar to the output shown in Figure 5.3.

Path Selection Plugin (PSP)

Figure 5.4 depicts the relationship between SATP, PSP, and NMP.

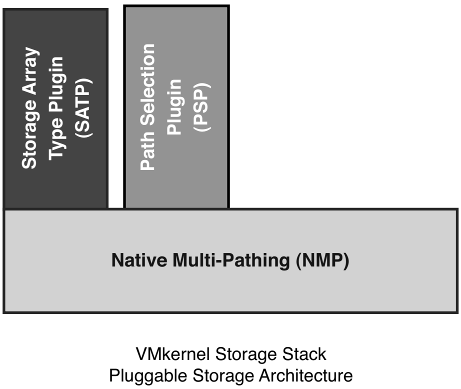


Figure 5.4 PSP

PSPs are PSA plug-ins that handle path selection policies and are replacements of failover policies used by the Legacy-MP (or Legacy Multipathing) used in releases prior to vSphere 4.x.

PSPs handle the following operations:

- Determine on which physical path to issue I/O requests being sent to a given storage device. Each PSP has access to a group of paths to the given storage device and has knowledge of the paths' states—for example, Active, Standby, Dead, as well as Asymmetric Logical Unit Access (ALUA), Asymmetric Access States (AAS) such as Active optimized Active non-optimized, and so on. This knowledge is obtained from what SATPs report to NMP. Refer to Chapter 6, "ALUA," for additional details about ALUA.
- Determine which path to activate next if the currently working physical path to storage device fails.

NOTE

PSPs do not need to know the actual storage array type (this function is provided by SATPs). However, a storage vendor developing a PSP may choose to do so (see Chapter 8, "Third-Party Multipathing I/O Plug-ins").

NMP communicates with PSPs for the following operations:

- Set up a new logical storage device and claim the physical paths to that device.
- Get the set of active physical paths currently used for path selection.
- Select a physical path on which to issue I/O requests for a given device.
- Select a physical path to activate when a path failure condition exists.

How to List PSPs on an ESXi 5 Host

To obtain a list of PSPs on a given ESXi 5 host, you may run the following command directly on the host or remotely via an SSH session, a vMA appliance, or ESXCLI:

```
# esxcli storage nmp psp list
```

An example of the output is shown in Figure 5.5.

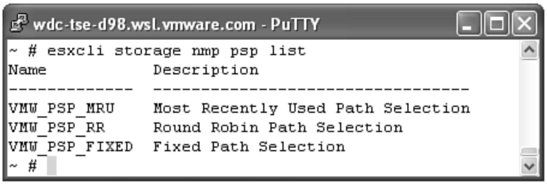


Figure 5.5 Listing PSPs

The output shows the default configuration of a freshly installed ESXi 5 host. If you installed third-party PSPs, they are also listed.

Third-Party Plug-ins

Figure 5.6 depicts the relationship between third-party plug-ins, NMP, and PSA.

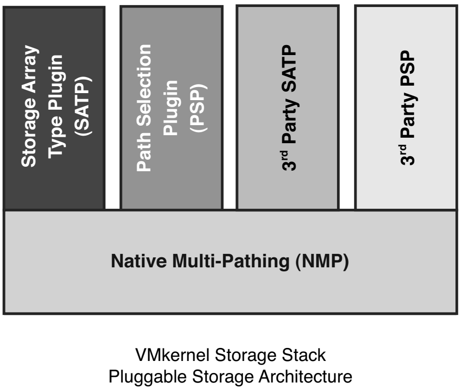


Figure 5.6 Third-party plug-ins

Because PSA is a modular architecture, VMware provided APIs to its storage partners to develop their own plug-ins. These plug-ins can be SATPs, PSPs, or MPPs.

Third-party SATPs and PSPs can run side by side with VMware-provided SATPs and PSPs.

The third-party SATPs and PSPs providers can implement their own proprietary functions relevant to each plug-in that are specific to their storage arrays. Some partners implement only multipathing and failover algorithms, whereas others implement load balancing and I/O optimization as well.

Examples of such plug-ins in vSphere 4.x that are also planned for vSphere 5 are

- **DELL_PSP_EQL_ROUTED**—Dell EqualLogic PSP that provides the following enhancements:
  - Automatic connection management
  - Automatic load balancing across multiple active paths
  - Increased bandwidth
  - Reduced network latency
- **HTI_SATP_HDLM**—Hitachi ported their HDLM MPIO (Multipathing I/O) management software to an SATP. It is currently certified for vSphere 4.1 with most of the USP family of arrays from Hitachi and HDS. A version is planned for vSphere 5 as well for the same set of arrays. Check with VMware HCL for the current list of certified arrays for vSphere 5 with this plug-in.

See Chapter 8 for further details.

Multipathing Plugins (MPPs)

Figure 5.7 depicts the relationship between MPPs, NMP, and PSA.

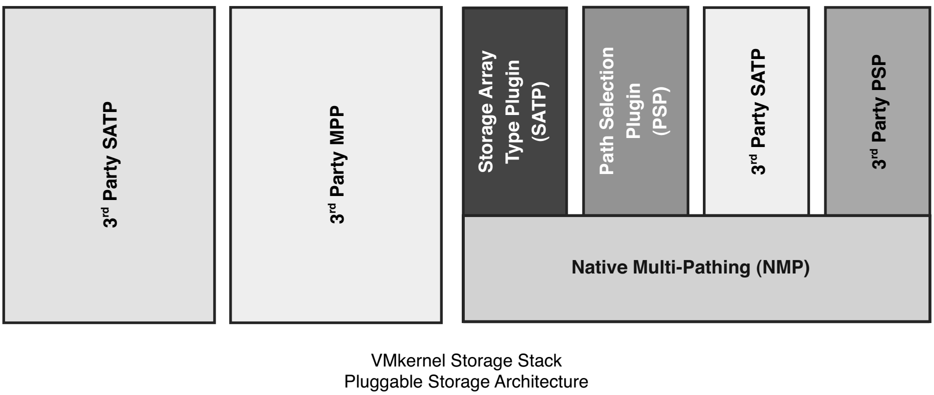


Figure 5.7 MPPs, including third-party plug-ins

MPPs that are not implemented as SATPs or PSPs can be implemented as MPPs instead. MPPs run side by side with NMP. An example of that is EMC PowerPath/VE. It is certified with vSphere 4.x and is planned for vSphere 5.

See Chapter 8 for further details.

Anatomy of PSA Components

Figure 5.8 is a block diagram showing the components of PSA framework.

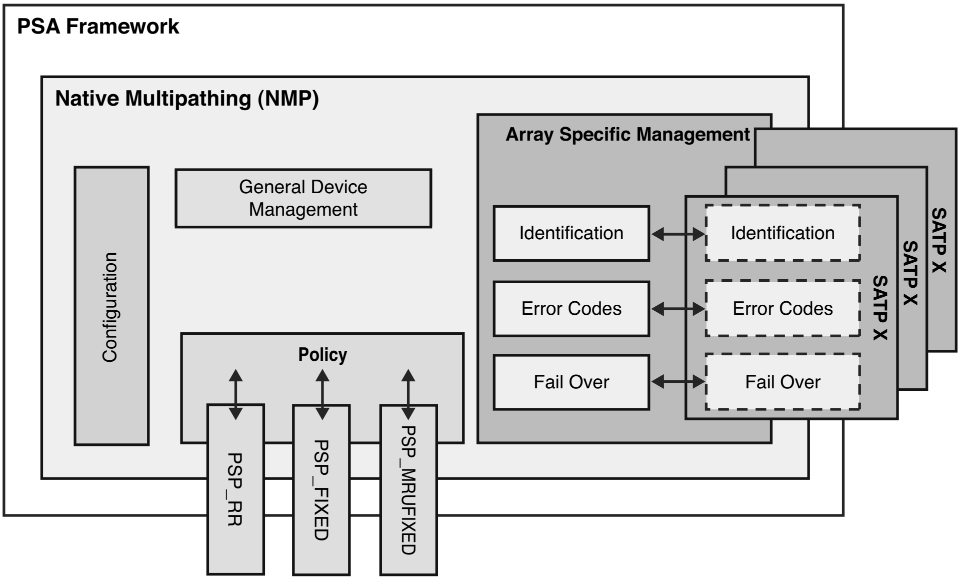


Figure 5.8 NMP components of PSA framework

Now that we covered the individual components of PSA framework, let's put its pieces together. Figure 5.8 shows the NMP component of the PSA framework. NMP provides facilities for configuration, general device management, array-specific management, and path selection policies.

The configuration of NMP-related components can be done via ESXCLI or the user interface (UI) provided by vSphere Client. Read more on this topic in the "Modifying PSA Plug-in Configurations Using the UI" section later in this chapter.

Multipathing and failover policy is set by NMP with the aid of PSPs. For details on how to configure the PSP for a given array, see the "Modifying PSA Plug-in Configurations Using the UI" section later in this chapter.

Arrray-specific functions are handled by NMP via the following functions:

- **Identification**—This is done by interpreting the response data to various inquiry commands (Standard Inquiry and Vital Product Data (VPD) received from the array/storage. This provides details of device identification which include the following:
  - Vendor
  - Model
  - LUN number
  - Device ID—for example, NAA ID, serial number
  - Supported mode pages—for example, page 80 or 83

  I cover more detail and examples of inquiry strings in Chapter 7, "Multipathing and Failover" in, the "LUN Discovery and Path Enumeration" section.
- **Error Codes**—NMP interprets error codes received from the storage arrays with help from the corresponding SATPs and acts upon these errors. For example, an SATP can identify a path as dead.
- **Failover**—After NMP interprets the error codes, it reacts in response to them. Continuing with the example, after a path is identified as dead, NMP instructs the relevant SATP to activate standby paths and then instructs the relevant PSP to issue the I/O on one of the activated paths. In this example, there are no active paths remaining, which results in activating standby paths (which is the case for Active/Passive arrays).

I/O Flow Through PSA and NMP

In order to understand how I/O sent to storage devices flows through the ESXi storage stack, you first need to understand some of the terminology relevant to this chapter.

Classification of Arrays Based on How They Handle I/O

Arrays can be one of the following types:

- **Active/Active**—This type of array would have more than one Storage Processor (SP) (also known as Storage Controller) that can process I/O concurrently on all SPs (and SP ports) with similar performance metrics. This type of array has no concept of logical unit number (LUN) ownership because I/O can be done on any LUN via any SP port from initiators given access to such LUNs.
- **Active/Passive**—This type of array would have two SPs. LUNs are distributed across both SPs in a fashion referred to as LUN ownership in which one of the SPs owns some of the LUNs and the other SP owns the remaining LUNs. The array accepts I/O to given LUN via ports on that SP that "owns" it. I/O sent to the non-owner SPs (also known as Passive SP) is rejected with a SCSI check condition and a sense code that translates to ILLEGAL REQUEST. Think of this like the No Entry sign you see at the entrance of a one-way street in the direction opposite to the traffic. For more details on sense codes, see Chapter 7 's "LUN Discovery and Path Enumeration" section.

NOTE

Some older firmware versions of certain arrays, such as HP MSA, are a variety of this type where one SP is active and the other is standby. The difference is that all LUNs are owned by the active SP and the standby SP is only used when the active SP fails. The standby SP still responds with a similar sense code to that returned from the passive SP described earlier.

- **Asymmetric Active/Active or AAA (AKA Pseudo Active/Active)**—LUNs on this type of arrays are owned by either SP similarly to the Active/Passive Arrays concept of LUN ownership. However, the array would allow concurrent I/O on a given LUN via ports on both SPs but with different I/O performance metrics as I/O is sent via proxy from the non-owner SP to the owner SP. In this case, the SP providing the lower performance metric accepts I/O to that LUN without returning a check condition. You may think of this as a hybrid between Active/Passive and Active/Active types. This can result in poor I/O performance of all paths to the owner SP that are dead, either due to poor design or LUN owner SP hardware failure.
- **Asymmetrical Logical Unit Access (ALUA)**—This type of array is an enhanced version of the Asymmetric Active/Active arrays and also the newer generation of some of the Active/Passive arrays. This technology allows initiators to identify the ports on the owner SP as one group and the ports on the non-owner SP as a

different group. This is referred to as Target Port Group Support (TPGS). The port group on the owner SP is identified as Active Optimized port group with the other group identified as Active Non-Optimized port group. NMP would send the I/O to a given LUN via a port in the ALUA optimized port group only as long as they are available. If all ports in that group are identified as dead, I/O is then sent to a port on the ALUA non-optimized port group. When sustained I/O is sent to the ALUA non-optimized port group, the array can transfer the LUN ownership to the non-owner SP and then transition the ports on that SP to ALUA optimized state. For more details on ALUA see Chapter 6.

Paths and Path States

From a storage perspective, the possible routes to a given LUN through which the I/O may travel is referred to as *paths*. A path consists of multiple points that start from the initiator port and end at the LUN.

A path can be in one of the states listed in Table 5.2.

Table 5.2 Path States

Path State	Description
Active	A path via an Active SP. I/O can be sent to any path in this state.
Standby	A path via a Passive or Standby SP. I/O is not sent via such a path.
Disabled	A path that is disabled usually by the vSphere Administrator.
Dead	A path that lost connectivity to the storage network. This can be due to an HBA (Host Bus Adapter), Fabric or Ethernet switch, or SP port connectivity loss. It can also be due to HBA or SP hardware failure.
Unknown	The state could not be determined by the relevant SATP.

Preferred Path Setting

A preferred path is a setting that NMP honors for devices claimed by VMW_PSP_FIXED PSP only. All I/O to a given device is sent over the path configured as the Preferred Path for that device. When the preferred path is unavailable, I/O is sent via one of the surviving paths. When the preferred path becomes available, I/O fails back to that path. By default, the first path discovered and claimed by the PSP is set as the preferred path. To change the preferred path setting, refer to the "Modifying PSA Plug-in Configurations Using the UI" section later in this chapter.

Figure 5.9 shows an example of a path to LUN 1 from host A (interrupted line) and Host B (interrupted line with dots and dashes). This path goes through HBA0 to target 1 on SPA.

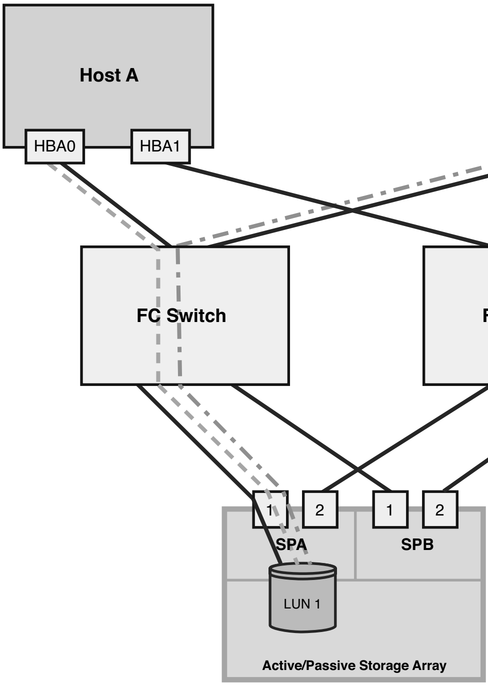


Figure 5.9 Paths to LUN1 from two hosts

Such a path is represented by the following Runtime Name naming convention. (Runtime Name is formerly known as Canonical Name.) It is in the format of HBAx:Cn:Ty:Lz—for example, vmhba0:C0:T0:L1—which reads as follows:

vmhba0, Channel 0, Target 0, LUN1

It represents the path to LUN 0 broken down as the following:

- **HBA0**—First HBA in this host. The vmhba number may vary based on the number of storage adapters installed in the host. For example, if the host has two RAID controllers installed which assume vmhba0 and vmhba1 names, the first FC HBA would be named vmhba2.
- **Channel 0**—Channel number is mostly zero for Fiber Channel (FC)- and Internet Small Computer System Interface (iSCSI)-attached devices to target 0, which is the

first target. If the HBA were a SCSI adapter with two channels (for example, internal connections and an external port for direct attached devices), the channel numbers would be 0 and 1.

- **Target 0**—The target definition was covered in Chapters 3, "FCoE Storage Connectivity," and 4, "iSCSI Storage Connectivity." The target number is based on the order in which the SP ports are discovered by PSA. In this case, SPA-Port1 was discovered before SPA-Port2 and the other ports on SPB. So, that port was given "target 0" as the part of the runtime name.

NOTE

Runtime Name, as the name indicates, does not persist between host reboots. This is due to the possibility that any of the components that make up that name may change due to hardware or connectivity changes. For example, a host might have an additional HBA added or another HBA removed, which would change the number assumed by the HBA.

Flow of I/O Through NMP

Figure 5.10 shows the flow of I/O through NMP.

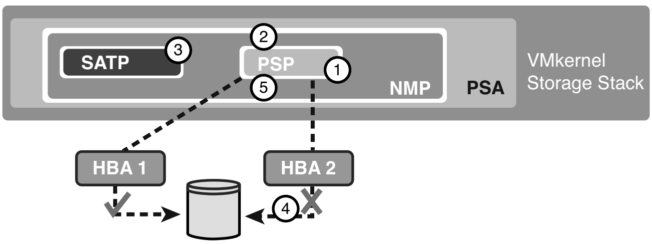


Figure 5.10 I/O flow through NMP

The numbers in the figure represent the following steps:

1. NMP calls the PSP assigned to the given logical device.
2. The PSP selects an appropriate physical path on which to send the I/O. If the PSP is VMW_PSP_RR, it load balances the I/O over paths whose states are Active or, for ALUA devices, paths via a target port group whose AAS is Active/Optimized.

3. If the array returns I/O error, NMP calls the relevant SATP.
4. The SATP interprets the error codes, activates inactive paths, and then fails over to the new active path.
5. PSP selects new active path to which it sends the I/O.

Listing Multipath Details

There are two ways by which you can display the list of paths to a given LUN, each of which are discussed in this section:

- Listing paths to a LUN using the UI
- Listing paths to a LUN using the CLI

Listing Paths to a LUN Using the UI

To list all paths to a given LUN in the vSphere 5.0 host, you may follow this procedure, which is similar to the procedure for listing all targets discussed earlier in Chapter 2, "Fibre Channel Storage Connectivity" Chapter 3 and Chapter 4:

1. Log on to the vSphere 5.0 host directly or to the vCenter server that manages the host using the VMware vSphere 5.0 Client as a user with Administrator privileges.
2. While in the Inventory—Hosts and Clusters view, locate the vSphere 5.0 host in the inventory tree and select it.
3. Navigate to the **Configuration** tab.
4. Under the Hardware section, select the **Storage** option.
5. Under the **View** field, click the **Devices** button.
6. Under the Devices pane, select one of the SAN LUNs (see Figure 5.11). In this example, the device name starts with DGC Fibre Channel Disk.

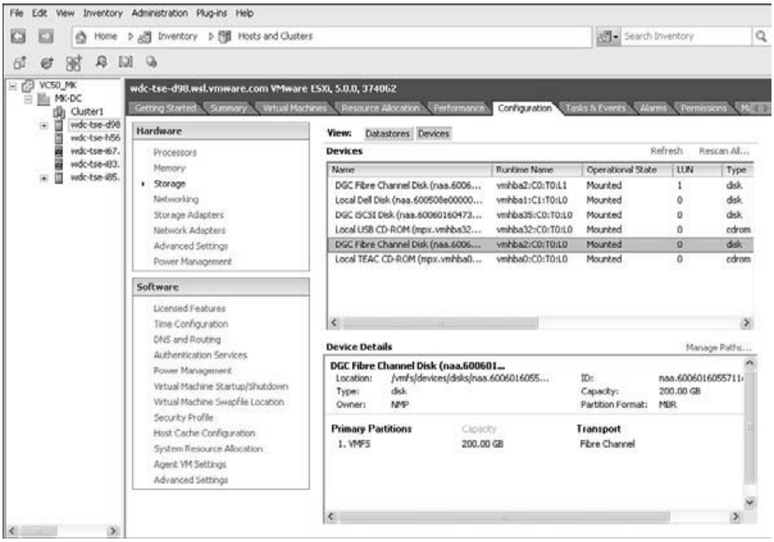


Figure 5.11 Listing storage devices

7. Select **Manage Paths** in the **Device Details** pane.
8. Figure 5.12 shows details for an FC-attached LUN. In this example, I sorted on the Runtime Name column in ascending order. The **Paths** section shows all available paths to the LUN in the format:
   - **Runtime Name**—vmhbaX:C0:Ty:Lz where X is the HBA number, y is the target number, and z is the LUN number. More on that in the "Preferred Path Setting" section later in this chapter.
   - **Target**—The WWNN followed by the WWPN of the target (separated by a space).
   - **LUN**—The LUN number that can be reached via the listed paths.
   - **Status**—This is the path state for each listed path.

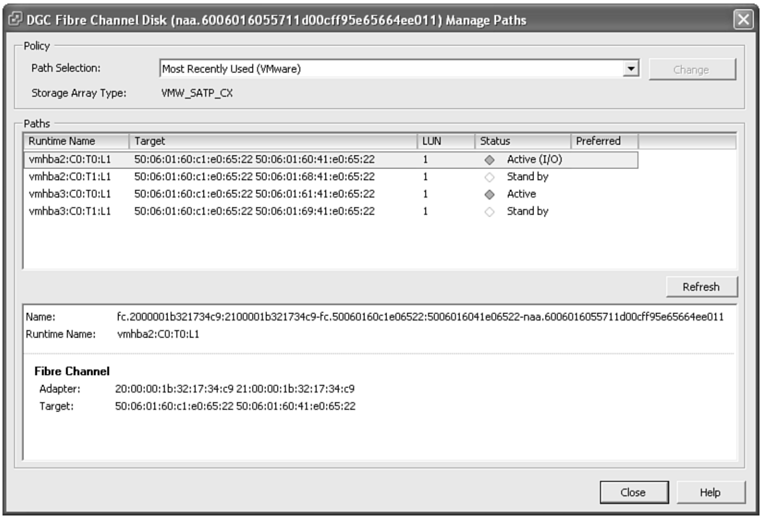


Figure 5.12 Listing paths to an FC-attached LUN

9. The Name field in the lower pane is a permanent one compared to the Runtime Name listed right below it. It is made up of three parts: HBA name, Target Name, and the LUN's device ID separated by dashes (for FC devices) or commas (for iSCSI devices). The HBA and Target names differ by the protocol used to access the LUN.

   Figure 5.12 shows the FC-based path Name, which is comprised of

   - **Initiator Name**—Made up from the letters FC followed by a period and then the HBA's WWNN and WWPN. The latter two are separated by a colon (these are discussed in Chapter 3).
   - **Target Name**—Made up from the target's WWNN and WWPN separated by a colon.
   - **LUN's Device ID**—In this example the NAA ID is naa.6006016055711d0 0cff95e65664ee011, which is based on the Network Address Authority naming convention and is a unique identifier of the logical device representing the LUN.

   Figure 5.13 shows the iSCSI-based path Name which is comprised of

   - **Initiator Name**—This is the iSCSI iqn name discussed in Chapter 4.

- **Target Name**—Made up from the target's iqn name and target number separated by colons. In this example, the target's iqn names are identical while the target numbers are different—such as t,1 and t,2. The second target info is not shown here, but you can display them by selecting one path at a time in the paths, pane to display the details in the lower pane.
- **LUN's Device ID**—In this example the NAA ID is naa.6006016047301a00 eaed23f5884ee011, which is based on the Network Address Authority naming convention and is a unique identifier of the logical device representing the LUN.

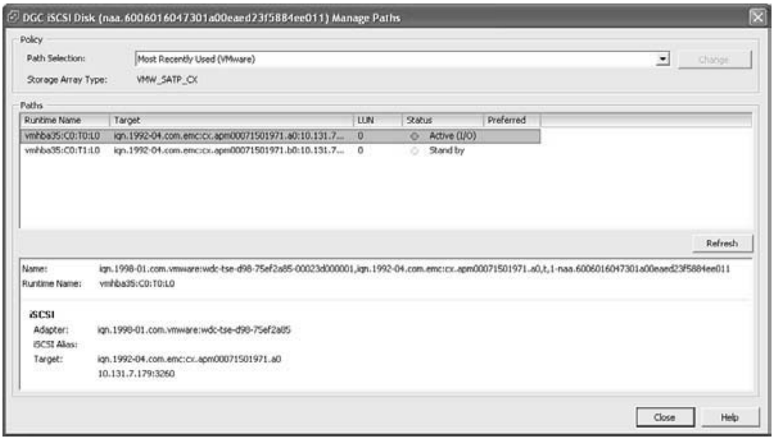


Figure 5.13 Listing paths to an iSCSI-attached LUN

Figure 5.14 shows a Fibre Channel over Ethernet (FCoE)-based path name, which is identical to the FC-based pathnames. The only difference is that fcoe is used in place of fc throughout the name.

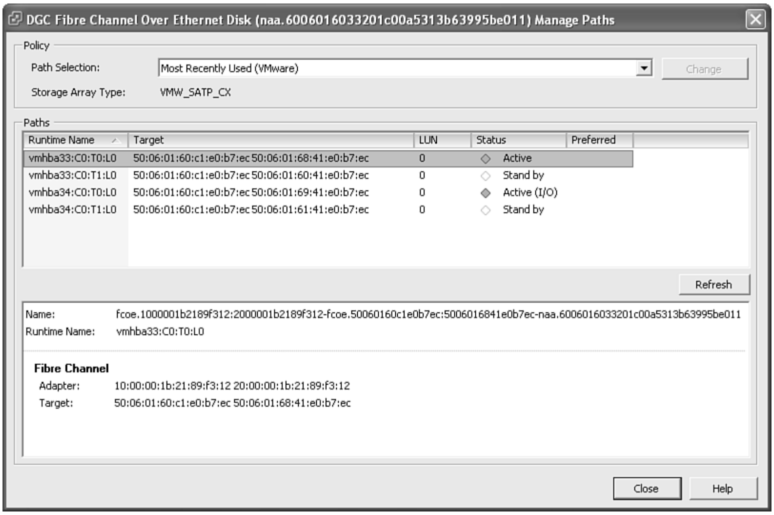


Figure 5.14 Listing paths to an FCoE-attached LUN

Listing Paths to a LUN Using the Command-Line Interface (CLI)

ESXCLI provides similar details to what is covered in the preceding section. For details about the various facilities that provide access to ESXCLI, refer to the "Locating HBA's WWPN and WWNN in vSphere 5 Hosts" section in Chapter 2.

The namespace of ESXCLI in vSphere 5.0 is fairly intuitive! Simply start with esxcli followed by the area of vSphere you want to manage—for example, esxcli network, esxcli software, esxcli storage—which enables you to manage Network, ESXi Software, and Storage, respectively. For more available options just run `esxcli -help`. Now, let's move on to the available commands:

Figure 5.15 shows the `esxcli storage nmp` namespace.

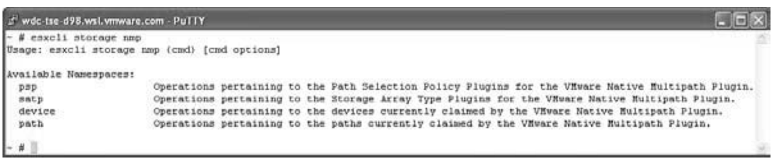


Figure 5.15 esxcli storage nmp namespace

The namespace of `esxcli storage nmp` is for all operations pertaining to native multipathing, which include psp, satp, device, and path.

I cover all these namespaces in detail later in the "Modifying PSA Plug-in Configurations Using the UI" section. The relevant operations for this section are

- `esxcli storage nmp path list`
- `esxcli storage nmp path list –d <device ID e.g. NAA ID>`

The first command provides a list of paths to *all* devices regardless of how they are attached to the host or which protocol is used.

The second command lists the paths to the device specified by the device ID (for example, NAA ID) by using the `-d` option.

The command in this example is

```
esxcli storage nmp path list -d naa.6006016055711d00cff95e65664ee011
```

You may also use the verbose command option `--device` instead of `-d`.

You can identify the NAA ID of the device you want to list by running a command like this:

```
esxcfg-mpath -b |grep -B1 "fc Adapter"| grep -v -e "--" |sed 's/
Adapter.*//'
```

You may also use the verbose command option `--list-paths` instead of `–b`.

The output of this command is shown in Figure 5.16.

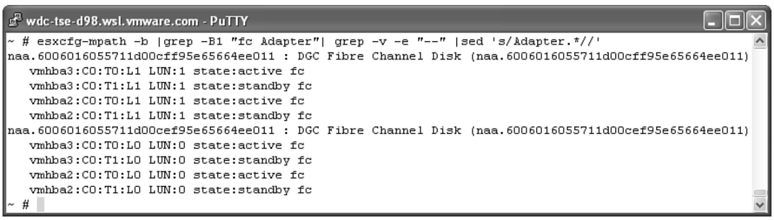


Figure 5.16 Listing paths to an FC-attached LUN via the CLI

This output shows all FC-attached devices. The Device Display Name of each device is listed followed immediately by the Runtime Name (for example, vmhba3:C0:T0:L1) of all paths to that device. This output is somewhat similar to the lagacy multipathing outputs you might have seen with ESX server release 3.5 and older.

The Device Display Name is actually listed after the device NAA ID and a colon.

From the runtime name you can identify the LUN number and the HBA through which they can be accessed. The HBA number is the first part of the Runtime Name, and the LUN number is the last part of that name.

All block devices conforming to the SCSI-3 standard have an NAA device ID assigned, which is listed at the beginning and the end of the Device Display Name line in the preceding output.

In this example, FC-attached LUN 1 has NAA ID `naa.6006016055711d00cff95e65664ee011` and that of LUN0 is `naa.6006016055711d00cef95e65664ee011`. I use the device ID for LUN 1 in the output shown in Figure 5.17.

Figure 5.17 Listing pathnames to an FC-attached device

You may use the verbose version of the command shown in Figure 5.17 by using `--device` instead of `-d`.

From the outputs of Figure 5.16 and 5.17, LUN 1 has four paths.

Using the Runtime Name, the list of paths to LUN1 is

- `vmhba3:C0:T1:L1`
- `vmhba3:C0:T0:L1`

- vmhba2:C0:T1:L1
- vmhba2:C0:T0:L1

This translates to the list shown in Figure 5.18 based on the physical pathnames. This output was collected using this command:

```
esxcli storage nmp path list -d naa.6006016055711d00cff95e65664ee011 |grep
fc
```

Or the verbose option using the following:

```
esxcli storage nmp path list --device naa.6006016055711d00cff95e65664ee011
|grep fc
```

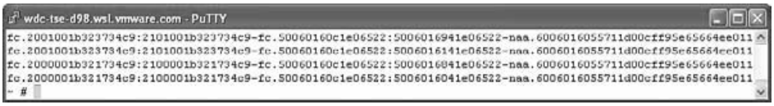


Figure 5.18 Listing physical pathnames of an FC-attached LUN

This output is similar to the aggregate of all paths that would have been identified using the corresponding UI procedure earlier in this section.

Using Table 2.1, "Identifying SP port association with each SP," in Chapter 2, we can translate the targets listed in the four paths as shown in Table 5.3:

Table 5.3 Identifying SP Port for LUN Paths

Runtime Name	Target WWPN	Sp Port Association
vmhba3:C0:T1:L1	5006016941e06522	SPB1
vmhba3:C0:T0:L1	5006016141e06522	SPA1
vmhba2:C0:T1:L1	5006016841e06522	SPB0
vmhba2:C0:T0:L1	5006016041e06522	SPA0

Identifying Path States and on Which Path the I/O Is Sent—FC

Still using the FC example (refer to Figure 5.17), two fields are relevant to the task of identifying the path states and the I/O path: Group State and Path Selection Policy Path Config. Table 5.4 shows the values of these fields and their meanings.

Table 5.4 Path State Related Fields

Runtime Name	Group State	PSP Path Config	Meaning
vmhba3:C0:T1:L1	Standby	non-current path; rank: 0	Passive SP—no I/O
vmhba3:C0:T0:L1	Active	non-current path; rank: 0	Active-SP—no I/O
vmhba2:C0:T1:L1	Standby	non-current path; rank: 0	Passive SP—no I/O
vmhba2:C0:T0:L1	Active	current path; rank: 0	Active SP—I/O

Combining the last two tables, we can extrapolate the following:

- The LUN is currently owned by SPA (therefore the state is Active).
- The I/O to the LUN is sent via the path to SPA Port 0.

NOTE

This information is provided by the PSP path configuration because its function is to "Determine on which physical path to issue I/O requests being sent to a given storage device" as stated under the PSP section.

The rank configuration listed here shows the value of 0. I discuss the ranked I/O in Chapter 7.

Example of Listing Paths to an iSCSI-Attached Device

To list paths to a specific iSCSI-attached LUN, try a different approach for locating the device ID:

```
esxcfg-mpath -m |grep iqn
```

You can also use the verbose command option:

```
esxcfg-mpath --list-map |grep iqn
```

The output for this command is shown in Figure 5.19.

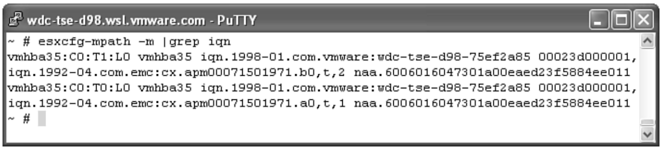


Figure 5.19 Listing paths to an iSCSI-attached LUN via the CLI

In the output, the lines wrapped. Each line actually begins with vmhba35 for readability. From this ouput, we have the information listed in Table 5.5.

Table 5.5 Matching Runtime Names with Their NAA IDs

Runtime Name	NAA ID
vmhba35:C0:T1:L0	naa.6006016047301a00eaed23f5884ee011
vmhba35:C0:T0:L0	naa.6006016047301a00eaed23f5884ee011

This means that these two paths are to the same LUN 0 and the NAA ID is `naa.6006016047301a00eaed23f5884ee011`.

Now, get the pathnames for this LUN. The command is the same as what you used for listing the FC device:

```
esxcli storage nmp path list -d naa.6006016047301a00eaed23f5884ee011
```

You may also use the verbose version of this command:

```
esxcli storage nmp path list --device naa.6006016047301a00eaed23f5884ee011
```

The output is shown in Figure 5.20.

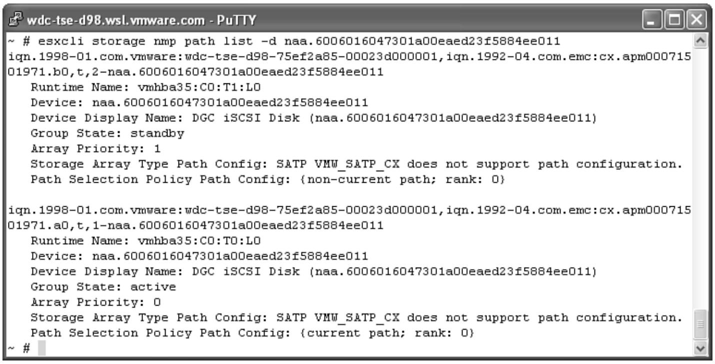


Figure 5.20 Listing paths to an iSCSI-attached LUN via CLI

Note that the path name was wrapped for readability.

Similar to what you observed with the FC-attached devices, the output is identical except for the actual path name. Here, it starts with iqn instead of fc.

The Group State and Path Selection Policy Path Config shows similar content as well. Based on that, I built Table 5.6.

Table 5.6 Matching Runtime Names with Their Target IDs and SP Ports

Runtime Name	Target IQN	Sp Port Association
vmhba35:C0:T1:L0	iqn.1992-04.com.emc:cx.apm00071501971.b0	SPB0
vmhba35:C0:T0:L0	iqn.1992-04.com.emc:cx.apm00071501971.a0	SPA0

To list only the pathnames in the output shown in Figure 5.20, you may append `|grep iqn` to the command.

The output of the command is listed in Figure 5.21 and was wrapped for readability. Each path name starts with `iqn`:

```
esxcli storage nmp path list --device naa.6006016047301a00eaed23f5884ee011
|grep iqn
```

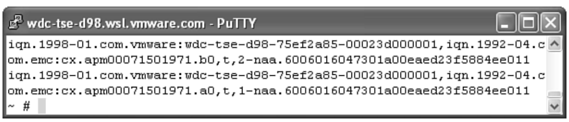


Figure 5.21 Listing pathnames of iSCSI-attached LUNs

Identifying Path States and on Which Path the I/O Is Sent—iSCSI

The process of identifying path states and I/O path for iSCSI protocol is identical to that of the FC protocol listed in the preceding section.

Example of Listing Paths to an FCoE-Attached Device

The process of listing paths to FCoE-attached devices is identical to the process for FC except that the string you use is `fcoe Adapter` instead of `fc Adapter`.

A sample output from an FCoE configuration is shown in Figure 5.22.

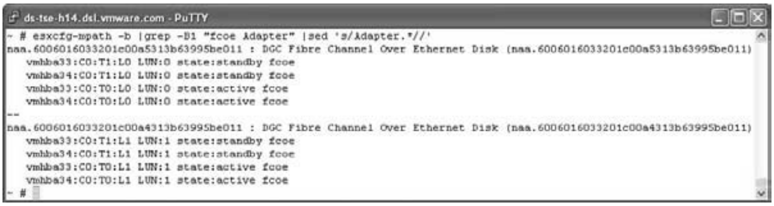


Figure 5.22 List of runtime paths of FCoE-attached LUNs via CLI

The command used is the following:

```
esxcfg-mpath -b |grep -B1 "fcoe Adapter" |sed 's/Adapter.*//'
```

You may also use the verbose command:

```
esxcfg-mpath --list-paths |grep -B1 "fcoe Adapter" |sed 's/Adapter.*//'
```

Using the NAA ID for LUN 1, the list of pathnames is shown in Figure 5.23.

Figure 5.23 List of pathnames of an FCoE-attached LUN

You may also use the verbose version of the command shown in Figure 5.23 by using `--device` instead of `-d`.

This translates to the physical pathnames shown in Figure 5.24.

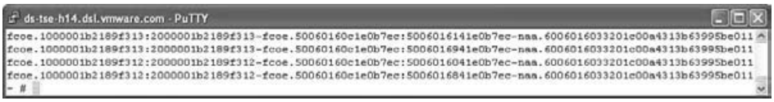


Figure 5.24 List of paths names of an FCoE LUN

The command used to collect the ouput shown in Figure 5.24 is

```
esxcli storage nmp path list -d 6006016033201c00a4313b63995be011 |grep fcoe
```

Using Table 2.1, "Identifying SP Port Association with Each SP," in Chapter 2, you can translate the targets listed in the returned four paths as shown in Table 5.7.

Table 5.7 Translation of FCoE Targets

Runtime Name	Target WWPN	SP Port Association
vmhba34:C0:T1:L1	5006016141e0b7ec	SPA1
vmhba34:C0:T0:L1	5006016941e0b7ec	SPB1
vmhba33:C0:T1:L1	5006016041e0b7ec	SPA0
vmhba33:C0:T0:L1	5006016841e0b7ec	SPB0

Identifying Path States and on Which Path the I/O Is Sent—FC

Still following the process as you did with the FC example (refer to Figure 5.17), two fields are relevant to the task of identifying the path states and the I/O path: Group State and Path Selection Policy Path Config. Table 5.8 shows the values of these fields and their meaning.

Table 5.8 Interpreting Path States—FCoE

Runtime Name	Group State	PSP Path Config	Meaning
vmhba34:C0:T1:L1	Standby	non-current path; rank: 0	Passive SP — no I/O
vmhba34:C0:T0:L1	Active	current path; rank: 0	Active-SP — I/O
vmhba33:C0:T1:L1	Standby	non-current path; rank: 0	Passive SP — no I/O
vmhba33:C0:T0:L1	Active	non-current path; rank: 0	Active SP — no I/O

Combining the last two tables, we can extrapolate the following:

- The LUN is currently “owned” by SPB (hence the state is Active).
- The I/O to the LUN is sent via the path to SPB Port 1.

Claim Rules

Each storage device is managed by one of the PSA plug-ins at any given time. In other words, a device cannot be managed by more than one PSA plug-in.

For example, a host that has a third-party MPP installed alongside with NMP, devices managed by the third-party MPP cannot be managed by NMP unless the configuration is changed to assign these devices to NMP. The process of associating certain devices with certain PSA plug-ins is referred to as *claiming* and is defined by Claim Rules. These rules define the correlation between a device and NMP or MPP. NMP has additional association between the claimed device and a specific SATP and PSP.

This section shows you how to list the various claim rules. The next section discusses how to change these rules.

Claim rules can be defined based on one or a combination of the following:

- **Vendor String**—In response to the standard inquiry command, the arrays return the standard inquiry response, which includes the Vendor string. This can be used in the definition of a claim rule based on the exact match. A partial match or a string with padded spaces does not work.
- **Model String**—Similar to the Vendor string, the Model string is returned as part of the standard inquiry response. Similar to the Vendor string, a claim rule can be defined using the exact match of the Model string and padded spaces are not supported here.
- **Transport**—Defining a claim rule based on the transport type, Transport facilitates claiming of all devices that use that transport. Valid transport types are block, fc, iscsi, iscsivendor, ide, sas, sata, usb, parallel, and unknown.
- **Driver**—Specifying a driver name as one of the criteria for a claim rule definition allows all devices accessible via such a driver to be claimed. An example of that is a claim rule to mask all paths to devices attached to an HBA that uses mptscsi driver.

MP Claim Rules

The first set of claim rules defines which MPP claims which devices. Figure 5.25 shows the default MP claim rules.

```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.25 Listing MP Claim Rules

The command to list these rules is

```
esxcli storage core claimrule list
```

The namespace here is for the Core Storage because the MPP definition is done on the PSA level. The output shows that this rule class is MP, which indicates that these rules define the devices' association to a specific multipathing plug-in.

There are two plugins specified here: NMP and MASK_PATH. I have already discussed NMP in the previous sections. The MASK_PATH plug-in is used for masking paths to specific devices and is a replacement for the deprecated Legacy Multipathing LUN Masking vmkernel parameter. I provide some examples in the "Modifying PSA Plug-in Configurations Using the UI" section.

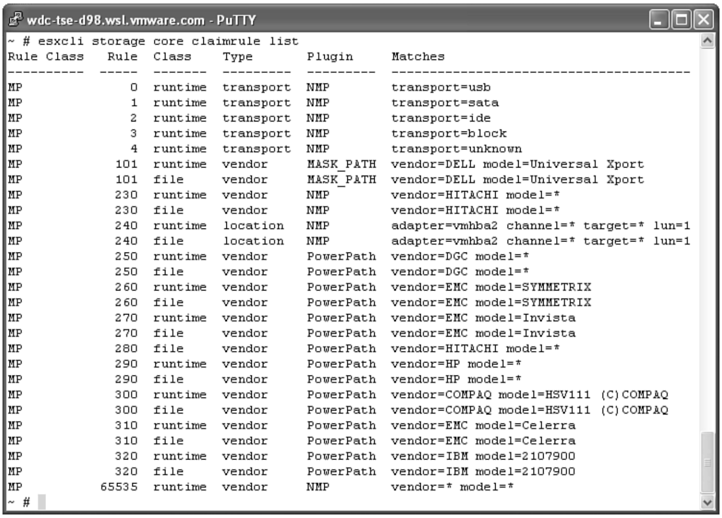
Table 5.9 lists each column name in the ouput along with an explanation of each column.

Table 5.9 Explanation of Claim Rules Fields

Column Name	Explanation
Rule Class	The plugin class for which this claim rule set is defined. This can be MP, Filter, or VAAI.
Rule	The rule number. This defines the order the rules are loaded. Similar to firewall rules, the first match is used and supersedes rules with larger numbers.
Class	The value can be `runtime` or `file`. A value of `file` means that the rule definitions were stored to the configuration files (more on this later in this section). A value of `Runtime` means that the rule was read from the configuration files and loaded into memory. In other words, it means that the rule is active. If a rule is listed as `file` only and no `runtime`, the rule was just created but has not been loaded yet. Find out more about loading rules in the next section.
Type	The type can be `vendor`, `model`, `transport`, or `driver`. See the explanation in the "Claim Rules" section.
Plugin	The name of the plug-in for which this rule was defined.
Matches	This is the most important field in the rule definition. This column shows the "Type" specified for the rule and its value. When the specified type is `vendor`, an additional parameter, `model`, must be used. The `model` string must be an exact string match or include an `*` as a wild card. You may use a ^ as "begins with" and then the string followed by an `*`—for example, `^OPEN-*`.

The highest rule number in any claim rules set is 65535. It is assigned here to a Catch-All rule that claims devices from “any” vendor with “any” model string. It is placed as the last rule in the set to allow for lower numbered rules to claim their specified devices. If the attached devices have no specific rules defined, they get claimed by NMP.

Figure 5.26 is an example of third-party MP plug-in claim rules.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            230  runtime  vendor     NMP        vendor=HITACHI model=*
MP            230  file     vendor     NMP        vendor=HITACHI model=*
MP            240  runtime  location   NMP        adapter=vmhba2 channel=* target=* lun=1
MP            240  file     location   NMP        adapter=vmhba2 channel=* target=* lun=1
MP            250  runtime  vendor     PowerPath  vendor=DGC model=*
MP            250  file     vendor     PowerPath  vendor=DGC model=*
MP            260  runtime  vendor     PowerPath  vendor=EMC model=SYMMETRIX
MP            260  file     vendor     PowerPath  vendor=EMC model=SYMMETRIX
MP            270  runtime  vendor     PowerPath  vendor=EMC model=Invista
MP            270  file     vendor     PowerPath  vendor=EMC model=Invista
MP            280  file     vendor     PowerPath  vendor=HITACHI model=*
MP            290  runtime  vendor     PowerPath  vendor=HP model=*
MP            290  file     vendor     PowerPath  vendor=HP model=*
MP            300  runtime  vendor     PowerPath  vendor=COMPAQ model=HSV111 (C)COMPAQ
MP            300  file     vendor     PowerPath  vendor=COMPAQ model=HSV111 (C)COMPAQ
MP            310  runtime  vendor     PowerPath  vendor=EMC model=Celerra
MP            310  file     vendor     PowerPath  vendor=EMC model=Celerra
MP            320  runtime  vendor     PowerPath  vendor=IBM model=2107900
MP            320  file     vendor     PowerPath  vendor=IBM model=2107900
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.26 Listing EMC PowerPath/VE claim rules.

Here you see that rules number 250 through 320 were added by PowerPath/VE, which allows PowerPath plug-in to claim all the devices listed in Table 5.10.

Table 5.10 Arrays Claimed by PowerPath

Storage Array	Vendor	Model
EMC CLARiiON Family	DGC	Any (* is a wild card)
EMC Symmetrix Family	EMC	SYMMETRIX
EMC Invista	EMC	Invista
HITACHI	HITACHI	Any
HP	HP	Any
HP EVA HSV111 family (Compaq Branded)	HP	HSV111 (C) COMPAQ
EMC Celerra	EMC	Celerra
IBM DS8000 family	IBM	2107900

NOTE

There is currently a known limitation with claim rules that use a partial match on the model string. So, older versions of PowerPath/VE that used to have rules stating model=OPEN may not claim the devices whose model string is something such as OPEN-V, OPEN-10, and so on. As evident from Figure 5.26, version 5.7 no longer uses partial matches. Instead, partial matches have been replaced with an `*`.

Plug-in Registration

New to vSphere 5 is the concept of *plug-in registration*. Actually this existed in 4.x but was not exposed to the end user. When a PSA plug-in is installed, it gets registered with the PSA framework along with their dependencies, if any, similar to the output in Figure 5.27.

```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core plugin registration list
Module Name          Plugin Name          Plugin Class  Dependencies                    Full Path
-------------------  -------------------  ------------  ------------------------------  ---------
mask_path_plugin     MASK_PATH            MP
nmp                  NMP                  MP
vmw_satp_symm        VMW_SATP_SYMM        SATP
vmw_satp_svc         VMW_SATP_SVC         SATP
vmw_satp_msa         VMW_SATP_MSA         SATP
vmw_satp_lsi         VMW_SATP_LSI         SATP
vmw_satp_inv         VMW_SATP_INV         SATP          vmw_satp_lib_cx
vmw_satp_eva         VMW_SATP_EVA         SATP
vmw_satp_eql         VMW_SATP_EQL         SATP
vmw_satp_cx          VMW_SATP_CX          SATP          vmw_satp_lib_cx
vmw_satp_alua_cx     VMW_SATP_ALUA_CX     SATP          vmw_satp_alua,vmw_satp_lib_cx
vmw_satp_lib_cx      None                 SATP
vmw_satp_alua        VMW_SATP_ALUA        SATP
vmw_satp_default_ap  VMW_SATP_DEFAULT_AP  SATP
vmw_satp_default_aa  VMW_SATP_DEFAULT_AA  SATP
vmw_satp_local       VMW_SATP_LOCAL       SATP
vmw_psp_lib          None                 PSP
vmw_psp_mru          VMW_PSP_MRU          PSP           vmw_psp_lib
vmw_psp_rr           VMW_PSP_RR           PSP           vmw_psp_lib
vmw_psp_fixed        VMW_PSP_FIXED        PSP           vmw_psp_lib
vmw_vaaip_emc        None                 VAAI
vmw_vaaip_mask       VMW_VAAIP_MASK       VAAI
vmw_vaaip_symm       VMW_VAAIP_SYMM       VAAI          vmw_vaaip_emc
vmw_vaaip_netapp     VMW_VAAIP_NETAPP     VAAI
vmw_vaaip_lhn        VMW_VAAIP_LHN        VAAI
vmw_vaaip_hds        VMW_VAAIP_HDS        VAAI
vmw_vaaip_eql        VMW_VAAIP_EQL        VAAI
vmw_vaaip_cx         VMW_VAAIP_CX         VAAI          vmw_vaaip_emc,vmw_satp_lib_cx
vaai_filter          VAAI_FILTER          Filter
~ #
```

Figure 5.27 Listing PSA plug-in registration

This output shows the following:

- **Module Name**—The name of the plug-in kernel module; this is the actual plug-in software binary as well as required libraries, if any, that get plugged into vmkernel.

- **Plugin Name**—This is the name by which the plug-in is identified. This is the exact name to use when creating or modifying claim rules.
- **Plugin class**—This is the name of the class to which the plug-in belongs. For example, the previous section covered the MP class of plug-ins. The next sections discuss SATP and PSP plug-ins and later chapters cover VAAI and VAAI_Filter classes.
- **Dependencies**—These are the libraries and other plug-ins which the registered plug-ins require to operate.
- **Full Path**—This is the full path to the files, libraries, or binaries that are specific to the registered plug-in. This is mostly blank in the default registration.

SATP Claim Rules

Now that you understand how NMP plugs into PSA, it's time to examine how SATP plugs into NMP.

Each SATP is associated with a default PSP. The defaults can be overridden using SATP claim rules. Before I show you how to list these rules, first review the default settings.

The command used to list the default PSP assignment to each SATP is

```
esxcli storage nmp satp list
```

The output of this command is shown in Figure 5.28.

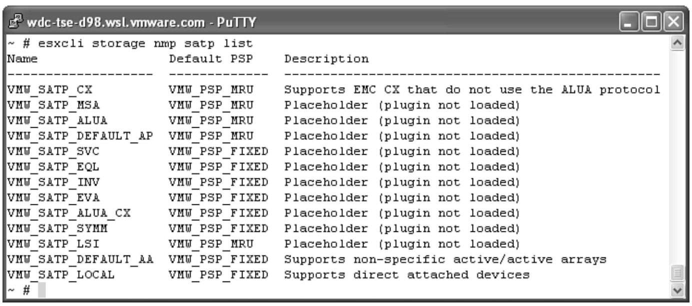


Figure 5.28 Listing SATPs and their default PSPs

The name space is Storage, NMP, and finally SATP.

> **NOTE**
>
> VMW_SATP_ALUA_CX plug-in is associated with VMW_PSP_FIXED. Starting with vSphere 5.0, the functionality of VMW_PSP_FIXED_AP has been rolled into VMW_PSP_FIXED. This facilitates the use of the Preferred Path option with ALUA arrays while still handling failover triggering events in a similar fashion to Active/Passive arrays. Read more on this in Chapter 6.

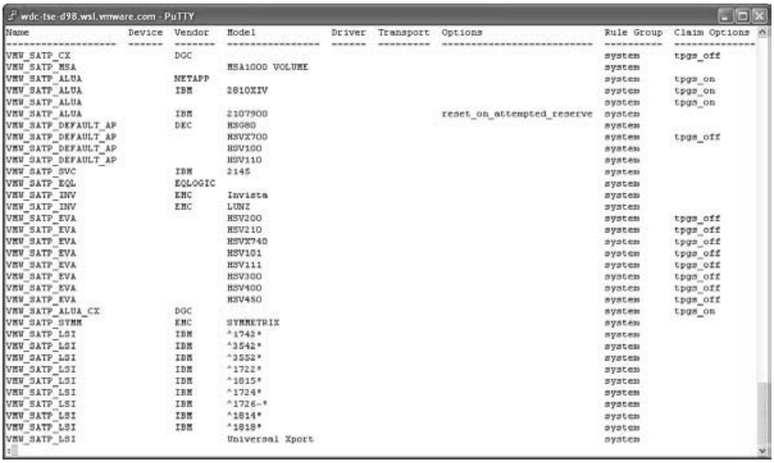
Knowing which PSP is the default policy for which SATP is half the story. NMP needs to know which SATP it will use with which storage device. This is done via SATP claim rules that associate a given SATP with a storage device based on matches to Vendor, Model, Driver, and/or Transport.

To list the SATP rule, run the following:

```
esxcli storage nmp satp rule list
```

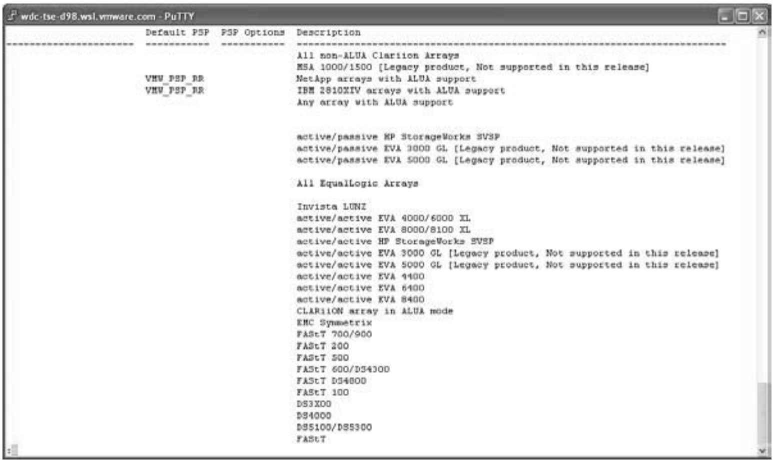
The output of the command is too long and too wide to capture in one screenshot. I have divided the output to a set of images in which I list a partial output then list the text of the full output in a subsequent table. Figures 5.29, 5.30, 5.31, and 5.32 show the four quadrants of the output.

> **TIP**
>
> To format the output of the preceding command so that the text is arranged better for readability, you can pipe the output to `less -S`. This truncates the long lines and aligns the text under their corresponding columns.
>
> So, the command would look like this:
>
> ```
> esxcli storage nmp satp list | less -S
> ```



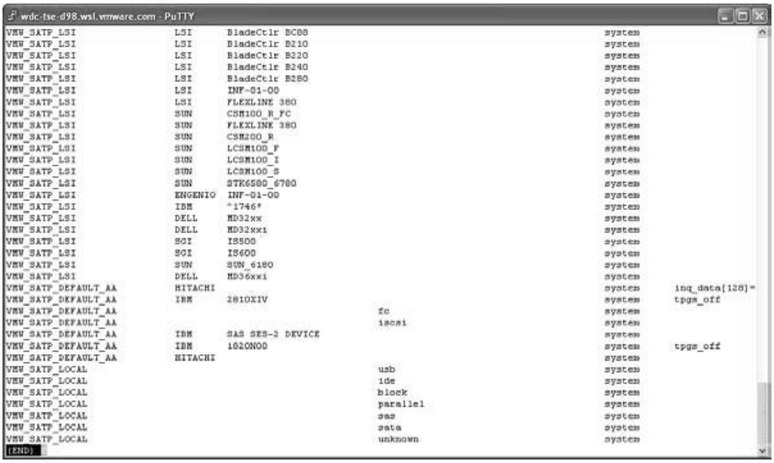
Name	Device	Vendor	Model	Driver	Transport	Options	Rule Group	Claim Options
VMW_SATP_CX		DGC					system	tpgs_off
VMW_SATP_MSA			MSA1000 VOLUME				system	
VMW_SATP_ALUA		NETAPP					system	tpgs_on
VMW_SATP_ALUA		IBM	2810XIV				system	tpgs_on
VMW_SATP_ALUA							system	tpgs_on
VMW_SATP_ALUA		IBM	2107900			reset_on_attempted_reserve	system	
VMW_SATP_DEFAULT_AP		DEC	HSG80				system	
VMW_SATP_DEFAULT_AP			HSVX700				system	tpgs_off
VMW_SATP_DEFAULT_AP			HSV100				system	
VMW_SATP_DEFAULT_AP			HSV110				system	
VMW_SATP_SVC		IBM	2145				system	
VMW_SATP_EQL		EQLOGIC					system	
VMW_SATP_INV		EMC	Invista				system	
VMW_SATP_INV		EMC	LUNZ				system	
VMW_SATP_EVA			HSV200				system	tpgs_off
VMW_SATP_EVA			HSV210				system	tpgs_off
VMW_SATP_EVA			HSVX740				system	tpgs_off
VMW_SATP_EVA			HSV101				system	tpgs_off
VMW_SATP_EVA			HSV111				system	tpgs_off
VMW_SATP_EVA			HSV300				system	tpgs_off
VMW_SATP_EVA			HSV400				system	tpgs_off
VMW_SATP_EVA			HSV450				system	tpgs_off
VMW_SATP_ALUA_CX		DGC					system	tpgs_on
VMW_SATP_SYMM		EMC	SYMMETRIX				system	
VMW_SATP_LSI		IBM	^1742*				system	
VMW_SATP_LSI		IBM	^3542*				system	
VMW_SATP_LSI		IBM	^3552*				system	
VMW_SATP_LSI		IBM	^1722*				system	
VMW_SATP_LSI		IBM	^1815*				system	
VMW_SATP_LSI		IBM	^1724*				system	
VMW_SATP_LSI		IBM	^1726-*				system	
VMW_SATP_LSI		IBM	^1814*				system	
VMW_SATP_LSI		IBM	^1818*				system	
VMW_SATP_LSI			Universal Xport				system	

Figure 5.29 Listing SATP claim rules—top-left quadrant of output.



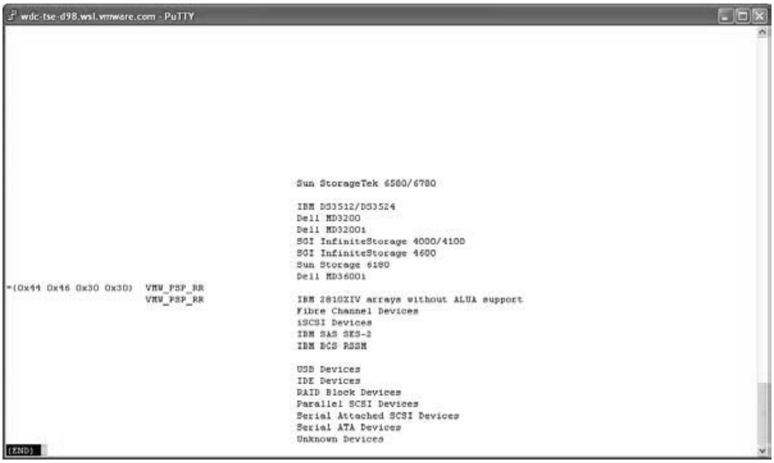
	Default PSP	PSP Options	Description
			All non-ALUA Clariion Arrays
			MSA 1000/1500 [Legacy product, Not supported in this release]
	VMW_PSP_RR		NetApp arrays with ALUA support
	VMW_PSP_RR		IBM 2810XIV arrays with ALUA support
			Any array with ALUA support
			active/passive HP StorageWorks SVSP
			active/passive EVA 3000 GL [Legacy product, Not supported in this release]
			active/passive EVA 5000 GL [Legacy product, Not supported in this release]
			All EqualLogic Arrays
			Invista LUNZ
			active/active EVA 4000/6000 XL
			active/active EVA 8000/8100 XL
			active/active HP StorageWorks SVSP
			active/active EVA 3000 GL [Legacy product, Not supported in this release]
			active/active EVA 5000 GL [Legacy product, Not supported in this release]
			active/active EVA 4400
			active/active EVA 6400
			active/active EVA 8400
			CLARiiON array in ALUA mode
			EMC Symmetrix
			FAStT 700/900
			FAStT 200
			FAStT 500
			FAStT 600/DS4300
			FAStT DS4800
			FAStT 100
			DS3X00
			DS4000
			DS5100/DS5300
			FAStT

Figure 5.30 Listing SATP claim rules—top-right quadrant of output.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
VMW_SATP_LSI          LSI      BladeCtlr BC88                         system
VMW_SATP_LSI          LSI      BladeCtlr B210                         system
VMW_SATP_LSI          LSI      BladeCtlr B220                         system
VMW_SATP_LSI          LSI      BladeCtlr B240                         system
VMW_SATP_LSI          LSI      BladeCtlr B280                         system
VMW_SATP_LSI          LSI      INF-01-00                              system
VMW_SATP_LSI          LSI      FLEXLINE 380                           system
VMW_SATP_LSI          SUN      CSM100_R_FC                            system
VMW_SATP_LSI          SUN      FLEXLINE 380                           system
VMW_SATP_LSI          SUN      CSM200_R                               system
VMW_SATP_LSI          SUN      LCSM100_F                              system
VMW_SATP_LSI          SUN      LCSM100_I                              system
VMW_SATP_LSI          SUN      LCSM100_S                              system
VMW_SATP_LSI          SUN      STK6580_6780                           system
VMW_SATP_LSI          ENGENIO  INF-01-00                              system
VMW_SATP_LSI          IBM      ^1746*                                 system
VMW_SATP_LSI          DELL     MD32xx                                 system
VMW_SATP_LSI          DELL     MD32xxi                                system
VMW_SATP_LSI          SGI      IS500                                  system
VMW_SATP_LSI          SGI      IS600                                  system
VMW_SATP_LSI          SUN      SUN_6180                               system
VMW_SATP_LSI          DELL     MD36xxi                                system
VMW_SATP_DEFAULT_AA   HITACHI                                         system   inq_data[128]=
VMW_SATP_DEFAULT_AA   IBM      2810XIV                                system   tpgs_off
VMW_SATP_DEFAULT_AA                                fc                 system
VMW_SATP_DEFAULT_AA                                iscsi              system
VMW_SATP_DEFAULT_AA   IBM      SAS SES-2 DEVICE                       system
VMW_SATP_DEFAULT_AA   IBM      1820N00                                system   tpgs_off
VMW_SATP_DEFAULT_AA   HITACHI                                         system
VMW_SATP_LOCAL                                     usb                system
VMW_SATP_LOCAL                                     ide                system
VMW_SATP_LOCAL                                     block              system
VMW_SATP_LOCAL                                     parallel           system
VMW_SATP_LOCAL                                     sas                system
VMW_SATP_LOCAL                                     sata               system
VMW_SATP_LOCAL                                     unknown            system
(END)
```

Figure 5.31 Listing SATP claim rules—bottom-left quadrant of output



```
wdc-tse-d98.wsl.vmware.com - PuTTY

                                         Sun StorageTek 6580/6780

                                         IBM DS3512/DS3524
                                         Dell MD3200
                                         Dell MD3200i
                                         SGI InfiniteStorage 4000/4100
                                         SGI InfiniteStorage 4600
                                         Sun Storage 6180
                                         Dell MD3600i
=(0x44 0x46 0x30 0x30)  VMW_PSP_RR
                        VMW_PSP_RR       IBM 2810XIV arrays without ALUA support
                                         Fibre Channel Devices
                                         iSCSI Devices
                                         IBM SAS SES-2
                                         IBM BCS RSSM

                                         USB Devices
                                         IDE Devices
                                         RAID Block Devices
                                         Parallel SCSI Devices
                                         Serial Attached SCSI Devices
                                         Serial ATA Devices
                                         Unknown Devices
(END)
```

Figure 5.32 Listing SATP claim rules—bottom-right quadrant of output

To make things a bit clearer, let's take a couple of lines from the output and explain what they mean.

Figure 5.33 shows the relevant rules for CLARiiON arrays both non-ALUA and ALUA capable. I removed three blank columns (Driver, Transport, and Options) to fit the content on the lines.

```
wdc-tse-d98.wsl.vmware.com - PuTTY
Name              Device  Vendor   Model    Rule Group  Claim Options  Description
VMW_SATP_CX               DGC               system      tpgs_off       All non-ALUA Clariion Arrays
VMW_SATP_ALUA_CX          DGC               system      tpgs_on        CLARiiON array in ALUA mode

:
```

Figure 5.33 CLARiiON Non-ALUA and ALUA Rules

The two lines show the claim rules for EMC CLARiiON CX family. Using this rule, NMP identifies the array as CLARiiON CX when the Vendor string is DGC. If NMP stopped at this, it would have used VMW_SATP_CX as the SATP for this array. However, this family of arrays can support more than one configuration. That is the reason the value `Claim Options` column comes in handy! So, if that option is `tpgs_off`, NMP uses the VMW_SATP_CX plug-in, and if the option is `tpgs_on`, NMP uses VMW_SATP_ALUA_CX. I explain what these options mean in Chapter 6.

Figure 5.34 shows another example that utilizes additional options. I removed the Device column to fit the content to the display.

```
wdc-tse-d98.wsl.vmware.com - PuTTY
Name                 Vendor   Model  Rule Group  Claim Options                        Default PSP
VMW_SATP_DEFAULT_AA  HITACHI         system      inq_data[128]={0x44 0x46 0x30 0x30}  VMW_PSP_RR
VMW_SATP_DEFAULT_AA  HITACHI         system
:
```

Figure 5.34 Claim rule that uses Claim Options

In this example, NMP uses VMW_SATP_DEFAULT_AA SATP with all arrays returning `HITACHI` as a model string. However, the default PSP is selected based on the values listed in the Claim Options column:

- If the column is blank, the default PSP (which is VMW_PSP_FIXED and is based on the list shown earlier in this section in Figure 5.28) is used. In that list, you see that VMW_SATP_DEFAULT_AA is assigned the default PSP named VMW_PSP_FIXED.
- If the column shows `inq_data[128]={0x44 0x46 0x30 0x30}`, which is part of the data reported from the array via the Inquiry String, NMP overrides the default PSP configuration and uses VMW_PSP_RR instead.

Modifying PSA Plug-in Configurations Using the UI

You can modify PSA plug-ins' configuration using the CLI and, to a limited extent, the UI. Because the UI provides far fewer options for modification, let me address that first to get it out of the way!

Which PSA Configurations Can Be Modified Using the UI?

You can change the PSP for a given device. However, this is done on a LUN level rather than the array.

Are you wondering why you would want to do that?

Think of the following scenario:

You have Microsoft Clustering Service (MSCS) cluster nodes in Virtual Machines (VMs) in your environment. The cluster's shared storage is Physical Mode Raw Device Mappings (RDMs), which are also referred to as (Passthrough RDMs). Your storage vendor recommends using Round-Robin Path Selection Policy (VMW_PSP_RR). However, VMware does not support using that policy with the MSCS clusters in shared RDMs.

The best approach is to follow your storage vendor's recommendations for most of the LUNs, but follow the procedure listed here to change just the RDM LUNs' PSP to their default PSPs.

Procedure to Change PSP via UI

1. Use the vSphere client to navigate to the MSCS node VM and right-click the VM in the inventory pane. Select **Edit Settings** (see Figure 5.35).

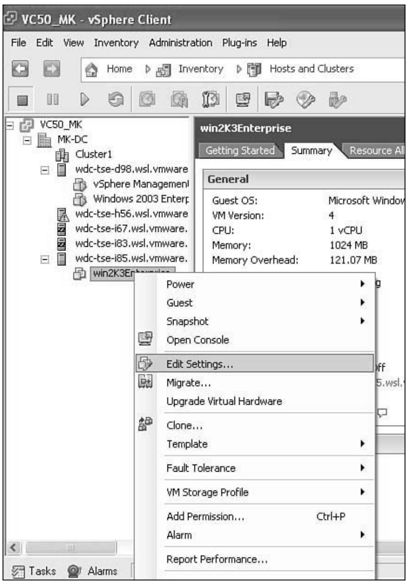


Figure 5.35 Editing VM's settings via the UI

The resulting dialog is shown in Figure 5.36.

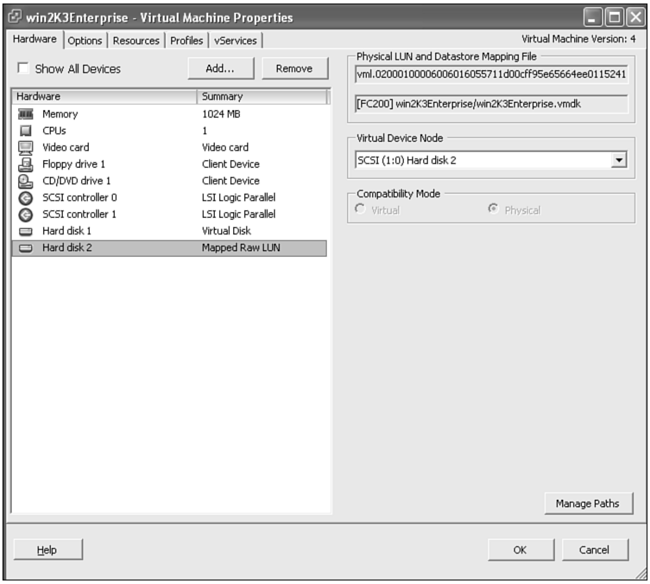


Figure 5.36 Virtual Machine Properties dialog

2. Locate the RDM listed in the Hardware tab. You can identify this by the summary column showing Mapped Raw LUN. On the top right-hand side you can locate the Logical Device Name, which is prefixed with vml in the field labeled Physical LUN and Datastore Mapping File.
3. Double-click the text in that field. Right-click the selected text and click **Copy** (see Figure 5.37).

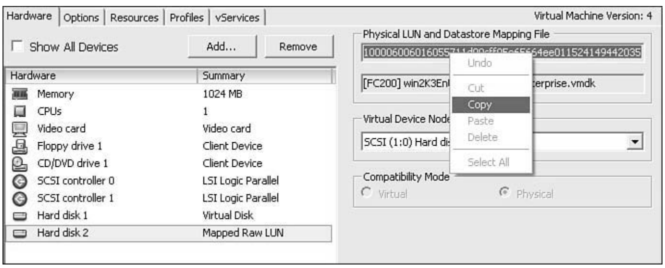


Figure 5.37 Copying RDM's VML ID (Logical Device Name) via the UI

4. I use the copied text to follow Steps 4 and 5 of doing the same task via the CLI in the next section. However, for this section, click the **Manage Paths** button in the dialog shown in Figure 5.37.

   The resulting Manage Paths dialog is shown in Figure 5.38.

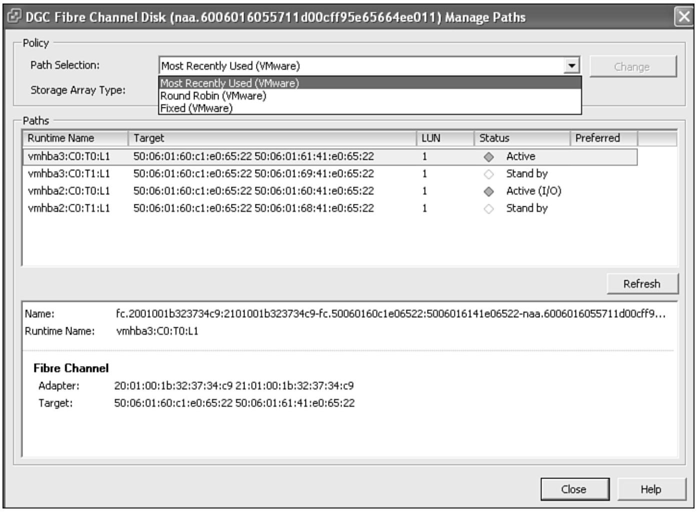


Figure 5.38 Modifying PSP selection via the UI

5. Click the pull-down menu next to the Path Selection field and change it from Round Robin (VMware) to the default PSP for your array. Click the **Change** button. To locate which PSP is the default, check VMware HCL. If the PSP listed there is Round Robin, follow the examples listed in the previous section, "SATP Claim Rules," to identify which PSP to select.

6. Click **Close**.

Modifying PSA Plug-ins Using the CLI

The CLI provides a range of options to configure, customize, and modify PSA plug-in settings. I provide the various configurable options and their use cases as we go.

Available CLI Tools and Their Options

New to vSphere 5.0 is the expansion of using esxcli as the main CLI utility for managing ESXi 5.0. The same binary is used whether you log on to the host locally or remotely via

SSH. It is also used by vMA or vCLI. This simplifies administrative tasks and improves portability of scripts written to use esxcli.

> **TIP**
>
> The only difference between the tools used locally or via SSH compared to those used in vMA and Remote CLI is that the latter two require providing the server name and the user's credentials on the command line. Refer to Chapter 3 in which I covered using the FastPass (fp) facility of vMA and how to add the users' credentials to the CREDSTORE environment variable on vCLI.
>
> Assuming that the server name and user credentials are set in the environment, the command-line syntax in all the examples in this book is identical regardless of where you use them.

ESXCLI Namespace

Figure 5.39 shows the command-line help for esxcli.

```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli
Usage: esxcli [options] {namespace}+ {cmd} [cmd options]

Options:
  --formatter=FORMATTER
                        Override the formatter to use for a given command. Available
                        formatter: xml, csv, keyvalue
  --format-param=FORMATPARAMS
                        Set a formatter parameter to give the formatter information
                        about how to format the command
  --debug               Enable debug or unsupported options
  --batch=BATCH         Batch mode (debug only)
  --batch-param=BATCHPARAM
                        Batch mode parameters (debug only)
  --version             Display version information for the script
  -?, --help            Display usage information for the script

Available Namespaces:
  esxcli                Commands that operate on the esxcli system itself allowing
                        users to get additional information.
  fcoe                  VMware FCOE commands.
  hardware              VMKernel hardware properties and commands for configuring
                        hardware.
  iscsi                 VMware iSCSI commands.
  license               Operations pertaining to the licensing of vmware and third
                        party modules on the ESX host. These operations currently only
                        include updating third party module licenses.
  network               Operations that pertain to the maintenance of networking on an
                        ESX host. This includes a wide variety of commands to
                        manipulate virtual networking components (vswitch, portgroup,
                        etc) as well as local host IP, DNS and general hsot networking
                        settings.
  software              Manage the ESXi software image and packages
  storage               VMware storage commands.
  system                VMKernel system properties and commands for configuring
                        properties of the kernel core system.
  vm                    A small number of operations that allow a user to Control
                        Virtual Machine operations.

~ #
```

Figure 5.39 Listing esxcli namespace

The relevant namespace for this chapter is `storage`. This is what most of the examples use. Figure 5.40 shows the command-line help for the `storage` namespace:

```
esxcli storage
```

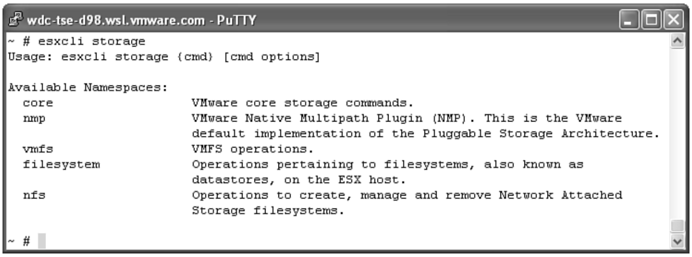


Figure 5.40 Listing esxcli `storage` namespace

Table 5.11 lists ESXCLI namespaces and their usage.

Table 5.11 Available Namespaces in the `storage` Namespace

Name Space	Usage
core	Use this for anything on the PSA level like other MPPs, PSA claim rules, and so on.
nmp	Use this for NMP and its “children,” such as SATP and PSP.
vmfs	Use this for handling VMFS volumes on snapshot LUNs, managing extents, and upgrading VMFS manually.
filesystem	Use this for listing, mounting, and unmounting supported datastores.
nfs	Use this to mount, unmount, and list NFS datastores.

Adding a PSA Claim Rule

PSA claim rules can be for MP, Filter, and VAAI classes. I cover the latter two in Chapter 6.

Following are a few examples of claim rules for the MP class.

Adding a Rule to Change Certain LUNs to Be Claimed by a Different MPP

In general, most arrays function properly using the default PSA claim rules. In certain configurations, you might need to specify a different PSA MPP.

A good example is the following scenario:

You installed PowerPath/VE on your ESXi 5.0 host but then later realized that you have some MSCS cluster nodes running on that host and these nodes use Passthrough RDMs (Physical compatibility mode RDM). Because VMware does not support third-party MPPs with MSCS, you must exclude the LUNs from being managed by PowerPath/VE.

You need to identify the device ID (NAA ID) of each of the RDM LUNs and then identify the paths to each LUN. You use these paths to create the claim rule.

Here is the full procedure:

1. Power off one of the MSCS cluster nodes and locate its home directory. If you cannot power off the VM, skip to Step 6.

   Assuming that the cluster node is located on Clusters_Datastore in a directory named node1, the command and its output would look like Listing 5.1.

Listing 5.1 Locating the RDM Filename

```
#cd /vmfs/volumes/Clusters_datastore/node1

#fgrep scsi1 *.vmx |grep fileName

scsi1:0.fileName = "/vmfs/volumes/4d8008a2-9940968c-04df-001e4f1fbf2a/
node1/quorum.vmdk"

scsi1:1.fileName = "/vmfs/volumes/4d8008a2-9940968c-04df-001e4f1fbf2a/
node1/data.vmdk"
```

   The last two lines are the output of the command. They show the RDM filenames for the node's shared storage, which are attached to the virtual SCSI adapter named `scsi1`.

2. Using the RDM filenames, including the path to the datastore, you can identify the logical device name to which each RDM maps as shown in Listing 5.2.

Listing 5.2 Identifying RDM's Logical Device Name Using the RDM Filename

```
#vmkfstools --queryrdm /vmfs/volumes/4d8008a2-9940968c-04df-001e4f1fbf2a/
node1/quorum.vmdk

Disk /vmfs/volumes/4d8008a2-9940968c-04df-001e4f1fbf2a/node1/quorum.vmdk is
a Passthrough Raw Device Mapping
Maps to: vml.02000100006006016055711d00cff95e65664ee011524149442035
```

You may also use the shorthand version using `-q` instead of `--queryrdm`.

This example is for the `quorum.vmdk`. Repeat the same process for the remaining RDMs. The device name is prefixed with vml and is highlighted.

3. Identify the NAA ID using the vml ID as shown in Listing 5.3.

Listing 5.3 Identifying NAA ID Using the Device vml ID

```
#esxcfg-scsidevs --list --device vml.02000100006006016055711d00cff95e65664
ee011524149442035 |grep Display


Display Name: DGC Fibre Channel Disk (naa.6006016055711d00cff95e65664ee011)
```

You may also use the shorthand version:

```
#esxcfg-scsidevs -l -d vml.02000100006006016055711d00cff95e65664
ee011524149442035 |grep Display
```

4. Now, use the NAA ID (highlighted in Listing 5.3) to identify the paths to the RDM LUN.

   Figure 5.41 shows the output of command:

```
esxcfg-mpath -m |grep naa.6006016055711d00cff95e65664ee011 | sed 's/
fc.*//'
```

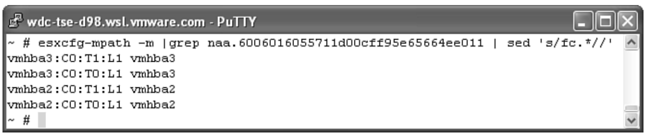


Figure 5.41 Listing runtime pathnames to an RDM LUN

You may also use the verbose version of the command:

```
esxcfg-mpath --list-map |grep naa.6006016055711d00cff95e65664ee011 |
sed 's/fc.*//'
```

This truncates the output beginning with "`fc`" to the end of the line on each line. If the protocol in use is not FC, replace that with "`iqn`" for iSCSI or "`fcoe`" for FCoE.

The output shows that the LUN with the identified NAA ID is LUN 1 and has four paths shown in Listing 5.4.

Listing 5.4 RDM LUN's Paths

```
vmhba3:C0:T1:L1
vmhba3:C0:T0:L1
vmhba2:C0:T1:L1
vmhba2:C0:T0:L1
```

If you cannot power off the VMs to run Steps 1–5, you may use the UI instead.

5. Use the vSphere client to navigate to the MSCS node VM. Right-click the VM in the inventory pane and then select **Edit Settings** (see Figure 5.42).

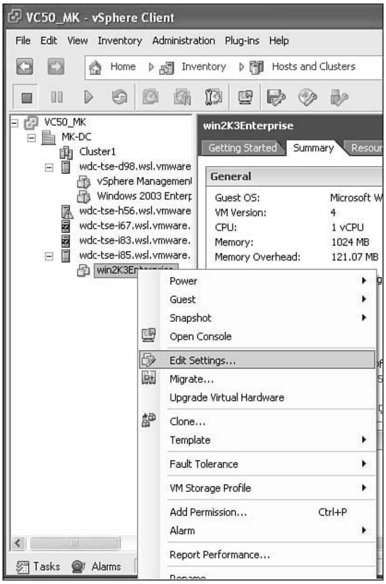


Figure 5.42 Editing VM's settings via the UI

6. In the resulting dialog (see Figure 5.43), locate the RDM listed in the Hardware tab. You can identify this by the summary column showing Mapped Raw LUN. On the top right-hand side you can locate the Logical Device Name, which is prefixed with `vml` in the field labeled Physical LUN and Datastore Mapping File.

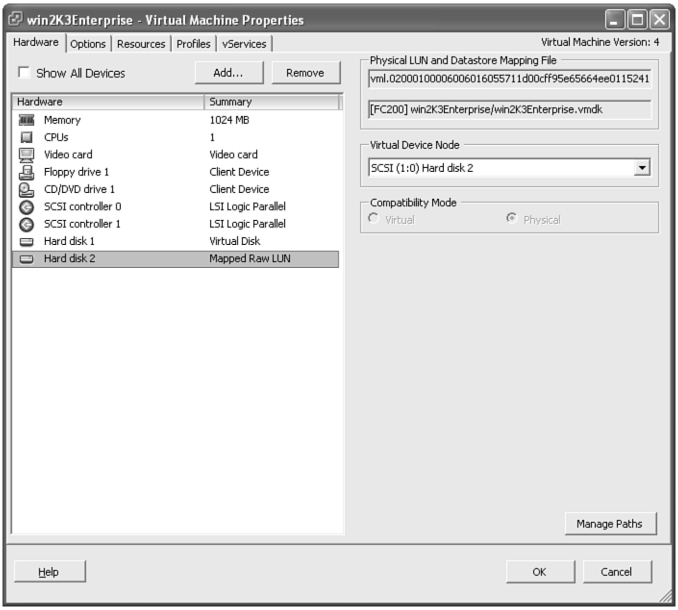


Figure 5.43 Virtual machine properties dialog

7. Double-click the text in that field. Right-click the selected text and click **Copy** as shown in Figure 5.44.

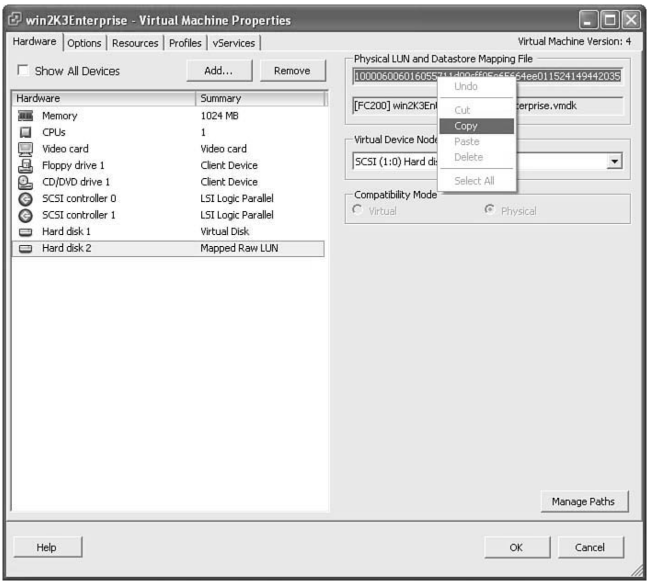


Figure 5.44 Copying RDM's VML ID (Logical Device Name) via the UI

8. You may use the copied text to follow Steps 4 and 5. Otherwise, you may instead get the list of paths to the LUN using the **Manage Paths** button in the dialog shown in Figure 5.44.

9. In the Manage Paths dialog (see Figure 5.45), click the Runtime Name column to sort it. Write down the list of paths shown there.

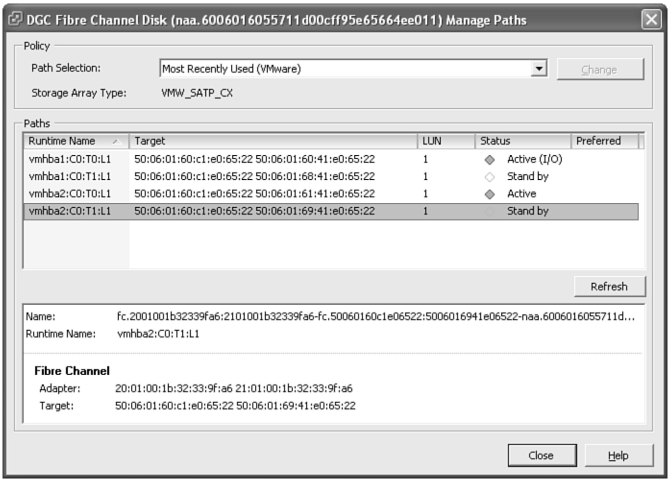


Figure 5.45 Listing the runtime pathnames via the UI

10. The list of paths shown in Figure 5.45 are

```
vmhba1:C0:T0:L1
vmhba1:C0:T1:L1
vmhba2:C0:T0:L1
vmhba2:C0:T1:L1
```

NOTE

Notice that the list of paths in the UI is different from that obtained from the command line. The reason can be easily explained; I used two different hosts for obtaining the list of paths. If your servers were configured identically, the path list should be identical as well.

However, this is not critical because the LUN's NAA ID is the same regardless of paths used to access it. This is what makes NAA ID the most unique element of any LUN, and that is the reason ESXi utilizes it for uniquely identifying the LUNs. I cover more on that topic later in Chapter 7.

11. Create the claim rule.

I use the list of paths obtained in Step 5 for creating the rule from the ESXi host from which it was obtained.

The Ground Rules for Creating the Rule

- The rule number must be lower than any of the rules created by PowerPath/VE installation. By default, they are assigned rules 250–320 (refer to Figure 5.26 for the list of PowerPath claim rules).
- The rule number must be higher than 101 because this is used by the Dell Mask Path rule. This prevents claiming devices masked by that rule.
- If you created other claim rules in the past on this host, use a rule number that is different from what you created in a fashion that the new rules you are creating now do not conflict with the earlier rules.
- If you must place the new rules in an order earlier than an existing rule but there are no rule numbers available, you may have to move one of the lower-numbered rules higher by the number of rules you plan on creating.

  For example, you have previously created rules numbered 102–110 and that rule 109 cannot be listed prior to the new rules you are creating. If the new rules count is four, you need to assign them rule numbers 109–112. To do that, you need to move rules 109 and 110 to numbers 113 and 114. To avoid having to do this in the future, consider leaving gaps in the rule numbers among sections.

  An example of moving a rule is

  ```
  esxcli storage core claimrule move --rule 109 --new-rule 113
  esxcli storage core claimrule move --rule 110 --new-rule 114
  ```

  You may also use the shorthand version:

  ```
  esxcli storage core claimrule move -r 109 -n 113
  esxcli storage core claimrule move -r 110 -n 114
  ```

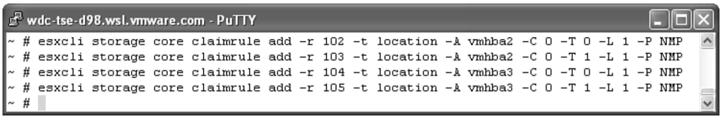
Now, let's proceed with adding the new claim rules:

1. The set of four commands shown in Figure 5.46 create rules numbered 102–105. The rules criteria are
   - The claim rule type is "location" (`-t location`).
   - The location is specified using each path to the same LUN in the format:
     - `–A` or `--adapter vmhba(x)` where `X` is the vmhba number associated with the path.

   - `–C` or `--channel (Y)` where `Y` is the channel number associated with the path.
   - `–T` or `--target (Z)` where `Z` is the target number associated with the path.
   - `–L` or `--lun (n)` where `n` is the LUN number.

- The plug-in name is NMP, which means that this claim rule is for NMP to claim the paths listed in each rule created.

> **NOTE**
>
> It would have been easier to create a single rule using the LUN's NAA ID by using the `--type device` option and then using `--device <NAA ID>`. However, the use of device as a rule type is not supported with MP class plug-ins.

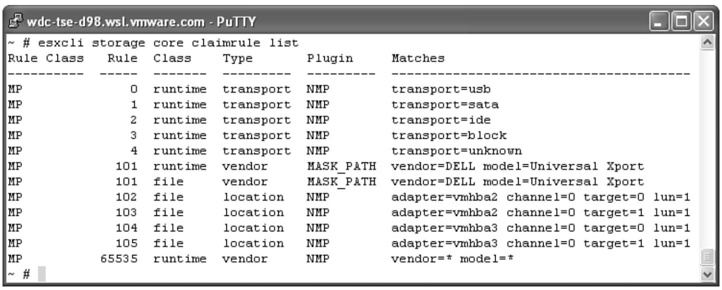


```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule add -r 102 -t location -A vmhba2 -C 0 -T 0 -L 1 -P NMP
~ # esxcli storage core claimrule add -r 103 -t location -A vmhba2 -C 0 -T 1 -L 1 -P NMP
~ # esxcli storage core claimrule add -r 104 -t location -A vmhba3 -C 0 -T 0 -L 1 -P NMP
~ # esxcli storage core claimrule add -r 105 -t location -A vmhba3 -C 0 -T 1 -L 1 -P NMP
~ #
```

Figure 5.46 Adding new MP claim rules

2. Repeat Step 1 for each LUN you want to reconfigure.
3. Verify that the rules were added successfully. To list the current set of claim rules, run the command shown in Figure 5.47:

   `esxcli storage core claimrule list.`



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            102  file     location   NMP        adapter=vmhba2 channel=0 target=0 lun=1
MP            103  file     location   NMP        adapter=vmhba2 channel=0 target=1 lun=1
MP            104  file     location   NMP        adapter=vmhba3 channel=0 target=0 lun=1
MP            105  file     location   NMP        adapter=vmhba3 channel=0 target=1 lun=1
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.47 Listing added claim rules

Notice that the four new rules are now listed, but the `Class` column shows them as file. This means that the configuration files were updated successfully but the rules were not loaded into memory yet.

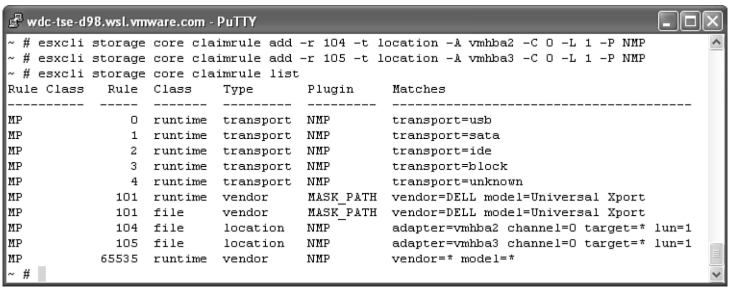
NOTE

I truncated the PowerPath rules in Figure 5.47 for readability. Also note that using the Location type utilizes the current runtime names of the devices, and they may change in the future. If your configuration changes—for example, adding new HBAs or removing existing ones—the runtime names change, too. This results in these claim rules claiming the wrong devices. However, in a static environment, this should not be an issue.

TIP

To reduce the number of commands used and the number of rules created, you may omit the `-T` or `--target` option, which assumes a wildcard. You may also use the `–u` or `--autoassign` option to auto-assign the rule number. However, the latter assigns rule numbers starting with 5001, which may be higher than the existing claim rules for the device hosting the LUN you are planning to claim.

Figure 5.48 shows a sample command line that implements a wildcard for the target. Notice that this results in creating two rules instead of four and the "target" match is `*`.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule add -r 104 -t location -A vmhba2 -C 0 -L 1 -P NMP
~ # esxcli storage core claimrule add -r 105 -t location -A vmhba3 -C 0 -L 1 -P NMP
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            104  file     location   NMP        adapter=vmhba2 channel=0 target=* lun=1
MP            105  file     location   NMP        adapter=vmhba3 channel=0 target=* lun=1
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.48 Adding MP claim rules using a wildcard

4. Before loading the new rules, you must first unclaim the paths to the LUN specified in that rule set. You use the NAA ID as the device ID:

```
esxcli storage core claiming unclaim --type device –-device naa.600601
6055711d00cff95e65664ee011
```

You may also use the shorthand version:

```
esxcli storage core claiming unclaim -t device -d naa.6006016055711d00
cff95e65664ee011
```

5. Load the new claim rules so that the paths to the LUN get claimed by NMP:

```
esxcli storage core claimrule load
```

6. Use the following command to list the claim rules to verify that they were successfully loaded:

```
esxcli storage core claimrule list
```

Now you see that each of the new rules is listed twice—once with file class and once with runtime class—as shown in Figure 5.49.

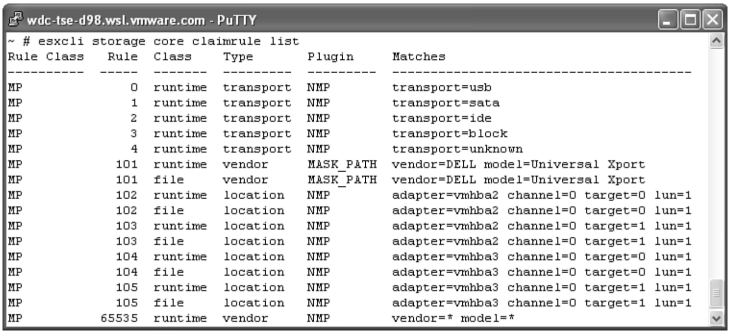


Figure 5.49 Listing MP claim rules

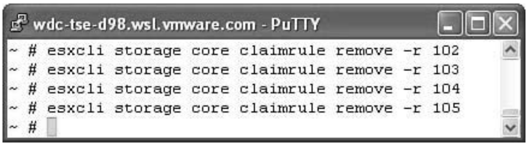
How to Delete a Claim Rule

Deleting a claim rule must be done with extreme caution. Make sure that you are deleting the rule you intend to delete. Prior to doing so, make sure to collect a "vm-support" dump by running `vm-support` from a command line at the host or via SSH. Alternatively, you can select the menu option Collect Diagnostics Data via the vSphere client.

To delete a claim rule, follow this procedure via the CLI (locally, via SSH, vCLI, or vMA):

1. List the current claim rules set and identify the claim rule or rules you want to delete. The command to list the claim rules is similar to what you ran in Step 6 and is shown in Figure 5.49.

2. For this procedure, I am going to use the previous example and delete the four claim rules I added earlier which are rules 102–105. The command for doing that is in Figure 5.50.



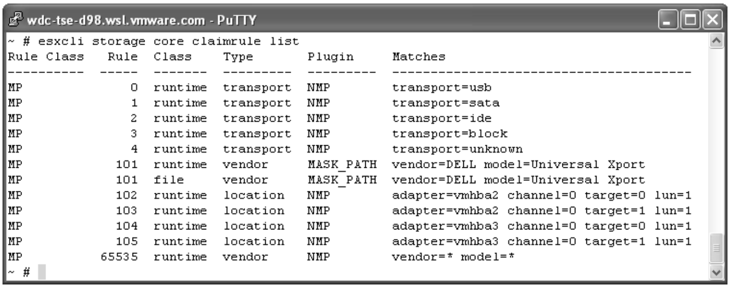
```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule remove -r 102
~ # esxcli storage core claimrule remove -r 103
~ # esxcli storage core claimrule remove -r 104
~ # esxcli storage core claimrule remove -r 105
~ #
```

Figure 5.50 Removing claim rules via the CLI

You may also run the verbose command:

```
esxcli storage core claimrule remove --rule <rule-number>
```

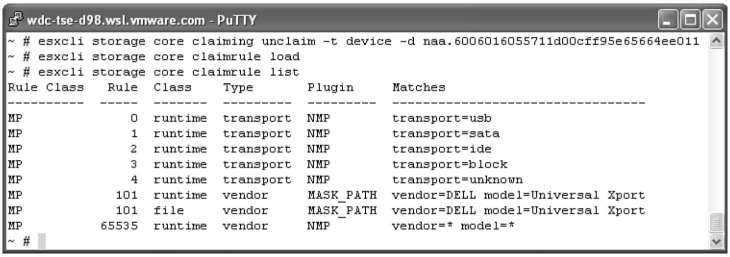
3. Running the `claimrule` list command now results in an output similar to Figure 5.51. Observe that even though I just deleted the claim rules, they still show up on the list. The reason for that is the fact that I have not loaded the modified claim rules. That is why the deleted rules show runtime in their `Class` column.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            102  runtime  location   NMP        adapter=vmhba2 channel=0 target=0 lun=1
MP            103  runtime  location   NMP        adapter=vmhba2 channel=0 target=1 lun=1
MP            104  runtime  location   NMP        adapter=vmhba3 channel=0 target=0 lun=1
MP            105  runtime  location   NMP        adapter=vmhba3 channel=0 target=1 lun=1
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.51 Listing MP claim rules

5. Because I know from the previous procedure the device ID (NAA ID) of the LUN whose claim rules I deleted, I ran the `unclaim` command using the `-t` device or `--type` option and then specified the `-d` or `--device` option with the NAA ID. I then loaded the claim rules using the load option. Notice that the deleted claim rules are no longer listed see Figure 5.52.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claiming unclaim -t device -d naa.6006016055711d00cff95e65664ee011
~ # esxcli storage core claimrule load
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.52 Unclaiming a device using its NAA ID and then loading the claim rules

You may also use the verbose command options:

```
esxcli storage core claiming unclaim --type device --device <Device-ID>
```

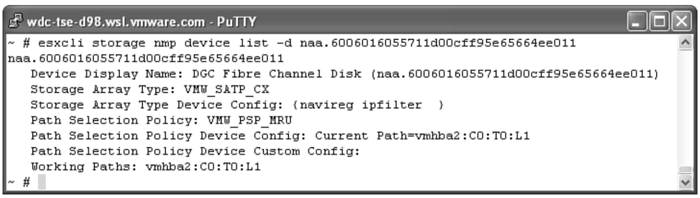
You may need to claim the device after loading the claim rule by repeating the claiming command using the "claim" instead of the "unclaim" option:

```
esxcli storage core claiming claim -t device -d <device-ID>
```

How to Mask Paths to a Certain LUN

Masking a LUN is a similar process to that of adding claim rules to claim certain paths to a LUN. The main difference is that the plug-in name is MASK_PATH instead of NMP as used in the previous example. The end result is that the masked LUNs are no longer visible to the host.

1. Assume that you want to mask LUN 1 used in the previous example and it still has the same NAA ID. I first run a command to list the LUN visible by the ESXi host as an example to show the before state (see Figure 5.53).



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage nmp device list -d naa.6006016055711d00cff95e65664ee011
naa.6006016055711d00cff95e65664ee011
   Device Display Name: DGC Fibre Channel Disk (naa.6006016055711d00cff95e65664ee011)
   Storage Array Type: VMW_SATP_CX
   Storage Array Type Device Config: {navireg ipfilter  }
   Path Selection Policy: VMW_PSP_MRU
   Path Selection Policy Device Config: Current Path=vmhba2:C0:T0:L1
   Path Selection Policy Device Custom Config:
   Working Paths: vmhba2:C0:T0:L1
~ #
```

Figure 5.53 Listing LUN properties using its NAA ID via the CLI

You may also use the verbose command option `--device` instead of `-d`.

2. Add the MASK_LUN claim rule, as shown in Figure 5.54.

```
~ # esxcli storage core claimrule add -r 110 -t location -A vmhba2 -C 0 -L 1 -P MASK_PATH
~ # esxcli storage core claimrule add -r 111 -t location -A vmhba3 -C 0 -L 1 -P MASK_PATH
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            110  file     location   MASK_PATH  adapter=vmhba2 channel=0 target=* lun=1
MP            111  file     location   MASK_PATH  adapter=vmhba3 channel=0 target=* lun=1
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.54 Adding Mask Path claim rules

As you see in Figure 5.54, I added rule numbers 110 and 111 to have MASK_PATH plug-in claim all targets to LUN1 via vmhba2 and vmhba3. The claim rules are not yet loaded, hence the file class listing and no runtime class listings.

3. Load and then list the claim rules (see Figure 5.55).

```
~ # esxcli storage core claimrule load
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            110  runtime  location   MASK_PATH  adapter=vmhba2 channel=0 target=* lun=1
MP            110  file     location   MASK_PATH  adapter=vmhba2 channel=0 target=* lun=1
MP            111  runtime  location   MASK_PATH  adapter=vmhba3 channel=0 target=* lun=1
MP            111  file     location   MASK_PATH  adapter=vmhba3 channel=0 target=* lun=1
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.55 Loading and listing claim rules after adding Mask Path rules

Now you see the claim rules listed with both file and runtime classes.

4. Use the reclaim option to unclaim and then claim the LUN using its NAA ID. Check if it is still visible (see Figure 5.56).

```
~ # esxcli storage core claiming reclaim -d naa.6006016055711d00cff95e65664ee011
~ # esxcli storage nmp device list -d naa.6006016055711d00cff95e65664ee011
Unknown device naa.6006016055711d00cff95e65664ee011
~ #
```

Figure 5.56 Reclaiming the paths after loading the Mask Path rules

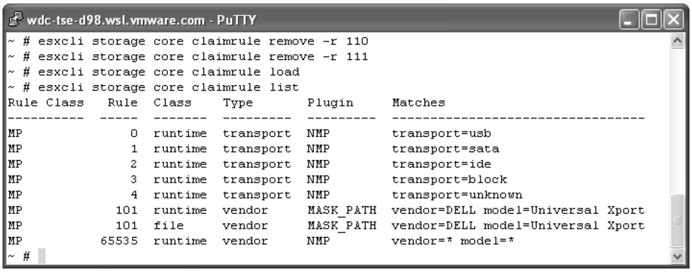
You may also use the verbose command option `--device` instead of `-d`.

Notice that after reclaiming the LUN, it is now an Unknown device.

How to Unmask a LUN

To unmask this LUN, reverse the preceding steps and then reclaim the LUN as follows:

1. Remove the MASK_PATH claim rules (numbers 110 and 111) as shown in Figure 5.57.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claimrule remove -r 110
~ # esxcli storage core claimrule remove -r 111
~ # esxcli storage core claimrule load
~ # esxcli storage core claimrule list
Rule Class   Rule  Class    Type       Plugin     Matches
----------  -----  -------  ---------  ---------  ---------------------------------
MP              0  runtime  transport  NMP        transport=usb
MP              1  runtime  transport  NMP        transport=sata
MP              2  runtime  transport  NMP        transport=ide
MP              3  runtime  transport  NMP        transport=block
MP              4  runtime  transport  NMP        transport=unknown
MP            101  runtime  vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP            101  file     vendor     MASK_PATH  vendor=DELL model=Universal Xport
MP          65535  runtime  vendor     NMP        vendor=* model=*
~ #
```

Figure 5.57 Removing the Mask Path claim rules

You may also use the verbose command options:

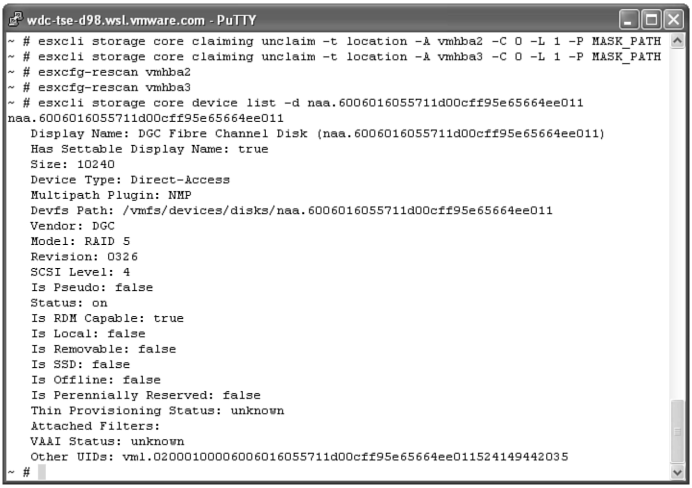
```
esxcli storage core claimrule remove --rule <rule-number>
```

2. Unclaim the paths to the LUN in the same fashion you used while adding the MASK_PATH claim rules—that is, using the `-t` location and omitting the `-T` option so that the target is a wildcard.

3. Rescan using both HBA names.

4. Verify that the LUN is now visible by running the list command.

   Figure 5.58 shows the outputs of Steps 2–4.



```
wdc-tse-d98.wsl.vmware.com - PuTTY
~ # esxcli storage core claiming unclaim -t location -A vmhba2 -C 0 -L 1 -P MASK_PATH
~ # esxcli storage core claiming unclaim -t location -A vmhba3 -C 0 -L 1 -P MASK_PATH
~ # esxcfg-rescan vmhba2
~ # esxcfg-rescan vmhba3
~ # esxcli storage core device list -d naa.6006016055711d00cff95e65664ee011
naa.6006016055711d00cff95e65664ee011
   Display Name: DGC Fibre Channel Disk (naa.6006016055711d00cff95e65664ee011)
   Has Settable Display Name: true
   Size: 10240
   Device Type: Direct-Access
   Multipath Plugin: NMP
   Devfs Path: /vmfs/devices/disks/naa.6006016055711d00cff95e65664ee011
   Vendor: DGC
   Model: RAID 5
   Revision: 0326
   SCSI Level: 4
   Is Pseudo: false
   Status: on
   Is RDM Capable: true
   Is Local: false
   Is Removable: false
   Is SSD: false
   Is Offline: false
   Is Perennially Reserved: false
   Thin Provisioning Status: unknown
   Attached Filters:
   VAAI Status: unknown
   Other UIDs: vml.02000100006006016055711d00cff95e65664ee011524149442035
~ #
```

Figure 5.58 Unclaiming the Masked Paths

You may also use the verbose command options:

```
esxcli storage core claiming unclaim --type location --adapter vmhba2
--channel 0 --lun 1 --plugin MASK_PATH
```

Changing PSP Assignment via the CLI

The CLI enables you to modify the PSP assignment per device. It also enables you to change the default PSP for a specific storage array or family of arrays. I cover the former use case first because it is similar to what you did via the UI in the previous section. I follow with the latter use case.

Changing PSP Assignment for a Device

To change the PSP assignment for a given device, you may follow this procedure:

1. Log on to the ESXi 5 host locally or via SSH as root or using vMA 5.0 as vi-admin.
2. Identify the device ID for each LUN you want to reconfigure:

   ```
   esxcfg-mpath -b |grep -B1 "fc Adapter"| grep -v -e "--" |sed 's/
   Adapter.*//'
   ```

   You may also use the verbose version of this command:

   ```
   esxcfg-mpath --list-paths grep -B1 "fc Adapter"| grep -v -e "--" | sed
   's/Adapter.*//'
   ```

   Listing 5.5 shows the output of this command.

Listing 5.5 Listing Device ID and Its Paths

```
naa.60060e8005275100000027510000011a : HITACHI Fibre Channel Disk (naa.6006
0e8005275100000027510000011a)
    vmhba2:C0:T0:L1 LUN:1 state:active fc
    vmhba2:C0:T1:L1 LUN:1 state:active fc
    vmhba3:C0:T0:L1 LUN:1 state:active fc
    vmhba3:C0:T1:L1 LUN:1 state:active fc
```

From there, you can identify the device ID (in this case, it is the NAA ID). Note that this output was collected using a Universal Storage Platform®V (USP V), USP VM, or Virtual Storage Platform (VSP).

This output means that LUN1 has device ID `naa.60060e8005275100000027510000011a`.

3. Using the device ID you identified, run this command:

```
esxcli storage nmp device set -d <device-id> --psp=<psp-name>
```

You may also use the verbose version of this command:

```
esxcli storage nmp device set --device <device-id> --psp=<psp-name>
```

For example:

```
esxcli storage nmp device set -d naa.60060e8005275100000027510000011a
--psp=VMW_PSP_FIXED
```

This command sets the device with ID `naa.60060e8005275100000027510000011a` to be claimed by the PSP named VMW_PSP_FIXED.

Changing the Default PSP for a Storage Array

There is no simple way to change the default PSP for a specific storage array unless that array is claimed by an SATP that is specific for it. In other words, if it is claimed by an SATP that also claims other brands of storage arrays, changing the default PSP affects *all* storage arrays claimed by the SATP. However, you may add an SATP claim rule that uses a specific PSP based on your storage array's Vendor and Model strings:

1. Identify the array's Vendor and Model strings. You can identify these strings by running

```
esxcli storage core device list -d <device ID> |grep 'Vendor\|Model'
```

Listing 5.6 shows an example for a device on an HP P6400 Storage Array.

Listing 5.6 Listing Device's Vendor and Model Strings

```
esxcli storage core device list -d naa.600508b4000f02cb0001000001660000
|grep 'Model\|Vendor'
   Vendor: HP
   Model: HSV340
```

In this example, the Vendor String is HP and the Model is HSV340.

2. Use the identified values in the following command:

```
esxcli storage nmp satp rule add --satp <current-SATP-USED> --vendor
<Vendor string> --model <Model string> --psp <PSP-name> --description
<Description>
```

> **TIP**
>
> It is always a good practice to document changes manually made to the ESXi host configuration. That is why I used the `--description` option to add a description of the rules I add. This way other admins would know what I did if they forget to read the change control record that I added using the company's change control software.

In this example, the command would be like this:

```
esxcli storage nmp satp rule add --satp VMW_SATP_EVA --vendor HP
--model HSV340 --psp VMW_PSP_FIXED --description "Manually added to
use FIXED"
```

It runs silently and returns an error if it fails.

Example of an error:

```
"Error adding SATP user rule: Duplicate user rule found for SATP VMW_
SATP_EVA matching vendor HP model HSV340 claim Options PSP VMW_PSP_
FIXED and PSP Options"
```

This error means that a rule already exists with these options. I simulated this rule by first adding it and then rerunning the same command. To view the existing SATP claim rules list for all HP storage arrays, you may run the following command:

```
esxcli storage nmp satp rule list |less -S |grep 'Name\|---\|HP'|less
-S
```

Figure 5.59 shows the output of this command (I cropped some blank columns, including Device, for readability):

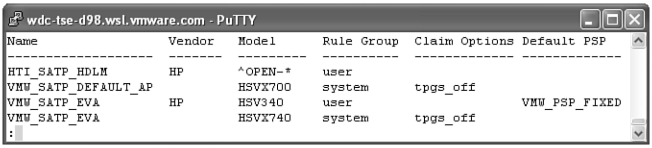


Figure 5.59 Listing SATP rule list for HP devices

You can easily identify non-system rules where the `Rule Group` column value is `user`. Such rules were added by a third-party MPIO installer or manually added by an ESXi 5 administrator. The rule in this example shows that I had already added VMW_PSP_FIXED as the default PSP for VMW_SATP_EVA when the matching vendor is HP and Model is HSV340.

I don't mean to state by this example that HP EVA arrays with HSV340 firmware should be claimed by this specific PSP. I am only using it for demonstration purposes. You *must* verify which PSP is supported by and certified for your specific storage array from the array vendor.

As a matter of fact, this HP EVA model happens to be an ALUA array and the SATP must be VMW_SATP_ALUA see Chapter 6. How did I know that? Let me explain!

- Look at the output in Figures 5.29–5.32. There you should notice that there are no listings of HP EVA arrays with Claim Options value of `tpgs_on`. This means that they were not claimed by any specific SATP explicitly.
- To filter out some clutter from the output, run the following command to list all claim rules with a match on Claim Options value of `tpgs_on`.

```
esxcli storage nmp satp rule list |grep 'Name\|---\|tpgs_on' |less -S
```

Listing 5.7 shows the output of that command:

Listing 5.7 Listing SATP Claim Rules List

```
Name                 Device  Vendor   Model     Rule Group Claim Options
-------------------  ------  -------  --------  ---------- ------------
VMW_SATP_ALUA                NETAPP             system      tpgs_on
VMW_SATP_ALUA                IBM      2810XIV   system      tpgs_on
VMW_SATP_ALUA                                   system      tpgs_on
VMW_SATP_ALUA_CX             DGC                system      tpgs_on
```

I cropped some blank columns for readability.

Here you see that there is a claim rule with a blank vendor and the Claim Options is `tpgs_on`. This claim rule claims *any* device with *any* vendor string as long as its Claim Options is `tpgs_on`.

Based on this rule, VMW_SATP_ALUA claims *all* ALUA-capable arrays including HP storage arrays based on a match on the Claim Options value of `tpgs_on`.

What does this mean anyway?

It means that the claim rule that I added for the HSV340 is wrong because it will force it to be claimed by an SATP that does not handle ALUA. I must remove the rule that I added then create another rule that does not violate the default SATP assignment:

1. To remove the SATP claim rule, use the same command used to add, substituting the add option with remove:

   ```
   esxcli storage nmp satp rule remove --satp VMW_SATP_EVA --vendor HP
   --model HSV340 --psp VMW_PSP_FIXED
   ```

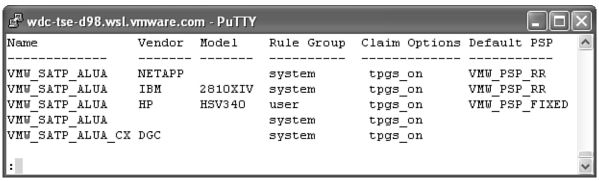
2. Add a new claim rule to have VMW_SATP_ALUA claim the HP EVA HSV340 when it reports Claim Options value as `tpgs_on`:

   ```
   esxcli storage nmp satp rule add --satp VMW_SATP_ALUA --vendor
   HP --model HSV340 --psp VMW_PSP_FIXED --claim-option tpgs_on
   --description "Re-added manually for HP HSV340"
   ```

3. Verify that the rule was created correctly. Run the same command used in Step 2 in the last procedure:

   ```
   esxcli storage nmp satp rule list |grep 'Name\|---\|tpgs_on' |less -S
   ```

   Figure 5.60 shows the output.



wdc-tse-d98.wsl.vmware.com - PuTTY

```
Name              Vendor  Model    Rule Group  Claim Options Default PSP
----------------- ------- -------- ----------  ------------- -----------
VMW_SATP_ALUA     NETAPP           system      tpgs_on       VMW_PSP_RR
VMW_SATP_ALUA     IBM     2810XIV  system      tpgs_on       VMW_PSP_RR
VMW_SATP_ALUA     HP      HSV340   user        tpgs_on       VMW_PSP_FIXED
VMW_SATP_ALUA                      system      tpgs_on
VMW_SATP_ALUA_CX  DGC              system      tpgs_on
:
```

Figure 5.60 SATP rule list after adding rule

Notice that the claim rule has been added in a position prior to the catch-all rule described earlier. This means that this HP EVA HSV340 model will be claimed by VMW_SATP_ALUA when the Claim Options value is `tpgs_on`.

NOTE

If you had manually set certain LUNs to a specific PSP previously, the preceding command will not affect that setting.

To reset such a LUN to use the current default PSP, use the following command:

```
esxcli storage nmp device set --device <device-ID> --default
```

For example:

```
esxcli storage nmp device set --device naa.6006016055711d00cef95e65
664ee011 --default
```

NOTE

All EVAs today have the `tpgs_on` option enabled by default, and it CANNOT be changed by the user. So adding an EVA claim rule would only be useful in the context of trying to use a different PSP by default for all EVA LUNs or assigning PSP defaults to EVA different from other ALUA-capable arrays using the default SATP_ALUA.

Summary

This chapter covered PSA (VMware Pluggable Storage Architecture) components. I showed you how to list PSA plug-ins and how they interact with vSphere ESXi 5. I also showed you how to list, modify, and customize PSA claim rules and how to work around some common issues.

It also covered how ALUA-capable devices interact with SATP claim rules for the purpose of using a specific PSP.

This page intentionally left blank

Index

Symbols

A

B

C

D

E

F

G

H

I

K-L

M

N

O

P

Q-R

S

T

U

V

W-Z