

EMC POWERPATH LOAD BALANCING AND FAILOVER

Comparison with native MPIO operating system solutions

Abstract

EMC® PowerPath® and PowerPath/VE provide intelligent load balancing and failover. This white paper explains how PowerPath's optimization policies enhance I/O throughput and ensure fast and non-disruptive path failover and failback. The native multipathing I/O solutions from operating systems vendors and their common load-balancing policies are contrasted with PowerPath.

February 2011

Copyright © 2011 EMC Corporation. All Rights Reserved.

EMC believes the information in this publication is accurate of its publication date. The information is subject to change without notice.

The information in this publication is provided “as is”. EMC Corporation makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on EMC.com.

VMware, ESX, ESXi, vMotion, and vSphere are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions. All other trademarks used herein are the property of their respective owners. All other trademarks used herein are the property of their respective owners.

Part Number h8180

Table of Contents

Executive summary	4
Audience.....	4
Multipathing solution considerations	4
Architecture	5
Using pseudo devices	7
Path set.....	8
PowerPath comparison to native solutions	9
Load balancing.....	9
PowerPath load balancing.....	9
Round Robin drawbacks.....	12
Windows load balancing	13
RHEL load balancing.....	14
VMware vSphere load balancing	15
HP-UX 11i v3 load balancing	16
AIX load balancing.....	17
Oracle Solaris load balancing.....	18
Load-balancing performance comparison.....	18
Path failover and recovery	20
Active-active arrays	21
ALUA and active-passive arrays	22
Proactive path testing on failures	23
Path testing	24
Virtualized and physical environments	27
Conclusion	28

Executive summary

EMC® PowerPath® Multipathing automatically tunes your storage area network (SAN) and selects alternate paths for your data if necessary. Residing on the server, PowerPath Multipathing enhances SAN performance and application availability. It also integrates multiple-path I/O capabilities, automatic load balancing, and path failover functions for complete path management. This white paper examines the functionality and benefits provided by EMC PowerPath and PowerPath/VE compared to the native multipath input/output (MPIO) solutions available from Microsoft Windows, Linux, VMware, AIX, Solaris, and HP-UX.

This paper does not investigate third-party multipathing solutions but rather focuses on native operating system MPIO solutions.

Audience

This white paper is intended for storage and server administrators as well as EMC personnel who want a deeper understanding of PowerPath multipathing and failover and how it contrasts to native MPIO solutions.

Multipathing solution considerations

In today's open systems data centers, there are a variety of applications, server platforms, storage arrays, operating systems, and protocols. Teams of application, server, and storage personnel must install, configure, manage, and monitor this environment. With this complexity combined with dynamic changes, choosing a multipathing product that has broad platform support and provides stable and predictable I/O performance has enormous benefits for both server and storage administrators.

When considering a SAN with multiple operating systems, there are many choices for I/O load balancing and path failover. However, there are many differences between a comprehensive path management solution and simplistic load balancing. Choosing an appropriate solution means understanding the key components of path management.

- **Load balancing** — Has the same basic premise in both storage and networking. Workloads are distributed across available hardware components. The goal of load balancing is to optimize resources, maximize throughput, and reduce I/O completion time. It is important to understand *how* I/O paths are being used. Administrators can improve application, storage, and network performance through load balancing and multipathing.
- **Path failover and recovery** — Path failover utilizes redundant I/O channels to redirect application reads and writes when one or more paths are no longer available. When the path returns to an operational state, the application I/O will

once again use the restored path. A path management tool supporting failover must be able to fail over and recover automatically and non-disruptively.

- **Virtualized and physical environments** — Mixing physical and virtual in the data center is common. There is varying usage depending on customer and application requirements. A path management solution must seamlessly support multiple operating system platforms across a wide variety of servers and storage.
- **Path testing** — Path tests must be run to ascertain the status of the paths. Administrators are more familiar with dead path testing. However, path testing is important on live and dead paths. Path testing can be performed on a fixed interval. Intelligent path tests are load-based, which means that they are run as needed.
- **Consistency across operating systems** — Data centers have multiple applications running on different operating systems. The systems will likely share the same Fibre Channel network and the attached storage array. There are complexities when managing I/O traffic from the host through the network to the storage array. Using a multipathing product that behaves similarly across SAN-attached host platforms will ensure that traffic patterns are stable and predictable under all operating conditions.
- **Management** — Monitoring, alerting, diagnostics, integration with operating system tools, and ease of configuration allow administrators to spend less time managing multipathing and more quickly and effectively respond to faults and potential trouble spots. PowerPath management will be discussed separately in an upcoming white paper entitled *EMC PowerPath Management*.

Architecture

PowerPath integrates into multiple operating systems providing multipathing and advanced services support to managed devices. PowerPath sits above the physical HBAs and below the file systems, volume managers, and applications, as shown in Figure 1. On some platforms, PowerPath sits above the disk driver but below the partition implementation. On others, it sits above both the disk driver and the partition implementation. I/O requests from the file systems, volume managers, and applications are intercepted and processed by the PowerPath driver.

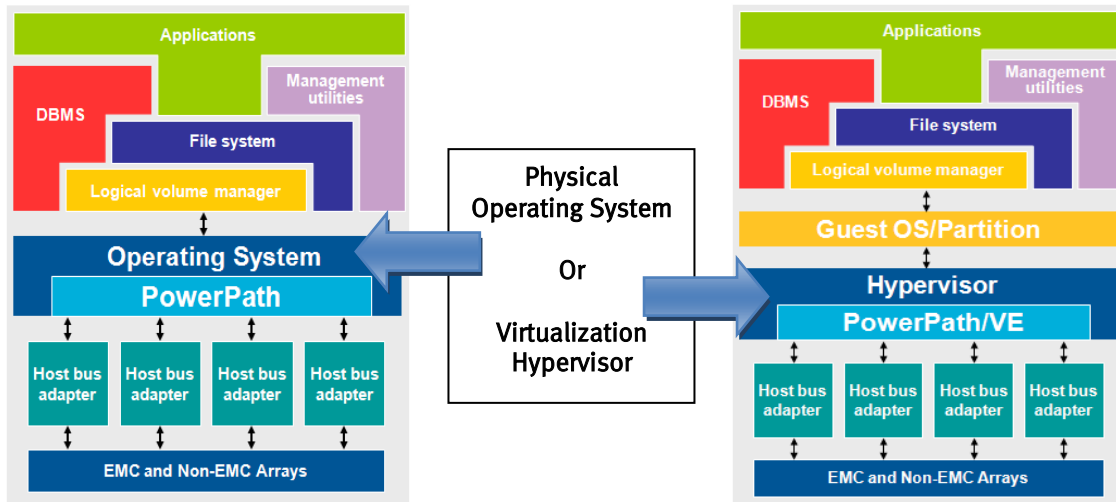


Figure 1. A high-level representation of the architecture of PowerPath and PowerPath/VE and their relationships to native operating systems

The PowerPath base driver is the core of the PowerPath kernel architecture. This is often referred to as the “C-clamp.” This name is derived from the conceptual structure of PowerPath (Figure 2), where the base driver appears to clamp the extensions together with the “jaws” of the driver forming a shape similar to the letter “C.” The base driver hooks into platform-specific features and layers as a driver. It includes the interface to the operating system and provides unified framework for Extensions. The Extensions are integrated in the form of a stack and are essentially kernel modules that manage multipath storage devices by passing server-to-storage I/O.

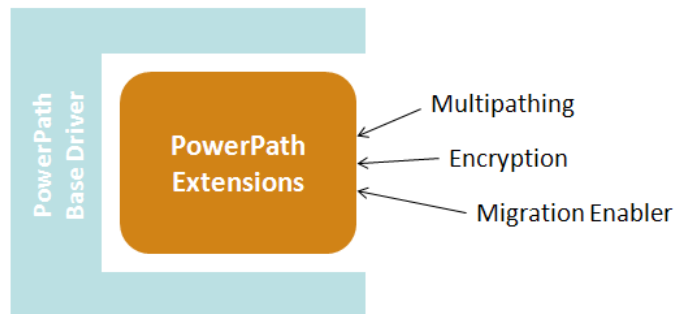


Figure 2. A PowerPath “C-clamp” is a logical representation of the base driver and Extensions relationship

There are multiple PowerPath Extensions providing various levels of functionality. The most commonly known Extensions are Multipathing, Encryption, and Migration Enabler. These are available on all platforms (except HP-UX, which does not support Encryption at this time). More information can be found on [Powerlink®](#).

Using pseudo devices

A pseudo device is a special kind of device (which could be defined as an operating system object that is used to access devices) created by PowerPath. For each logical device that PowerPath discovers, it creates one PowerPath device, called a pseudo device. One or more native paths to the same storage LUN are presented as a single pseudo device. A pseudo device provides load balancing and failover and represents all paths (native devices) that are available to the related logical device.

Each operating system creates native devices to represent and provide access to logical devices. A native device is path-specific, and represents a single path to a logical device. The device is native in that it is provided by the operating system. No reconfiguration is required for applications to use native devices.

The most important characteristic of a pseudo device is that it is location-independent. A native device is identified by its controller/ target/ disk (CTD) number combination. This identification makes the native device location-dependent. However, a PowerPath device is not identified in this manner since it is related to a logical device.

PowerPath presents its devices differently, depending on the platform. Much of this difference is due to the design of the host operating system. PowerPath supports pseudo devices on all platforms except HP-UX. Examples of pseudo devices are *harddisk1* on Windows, *emcpower10* on VMware vSphere™, or *emcpowerc* on Linux.

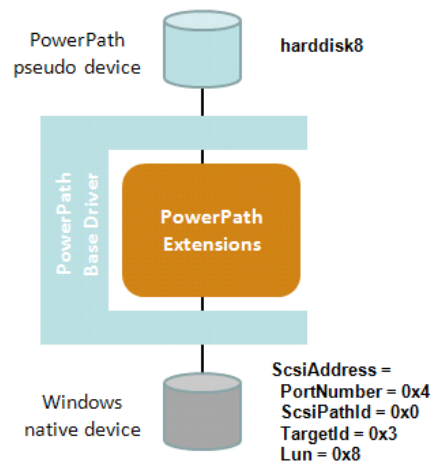


Figure 3. Example of pseudo and native devices on Microsoft Windows

On some operating systems, PowerPath can manage paths for native devices as well. When using native devices, PowerPath is transparent to the application. PowerPath maintains the correspondence between an individual native device and the [path set](#) to which it belongs. As I/O requests are processed, PowerPath redirects requests to the native device in the path set that will provide the best throughput. Pseudo devices are preferable because they allow administrators to use advanced PowerPath features like data-at-rest encryption with PowerPath Encryption with RSA® and non-disruptive migrations with PowerPath Migration Enabler.

Note: In SLES11 and RHEL6 (when supported), native devices are not recommended.

Path set

PowerPath groups all paths to the same logical device into a *path set*. PowerPath creates a path set for each logical device, and populates the path set with all usable paths to that logical device. Path sets are used for both load balancing and path failover, as shown in Figure 4. From an application standpoint, a path set appears as a single, highly available path to storage. PowerPath hides the complexity of the path in the set between the host and storage system. With the logical concept of a path set, PowerPath hides multiple HBAs, cables, ports, and switches.

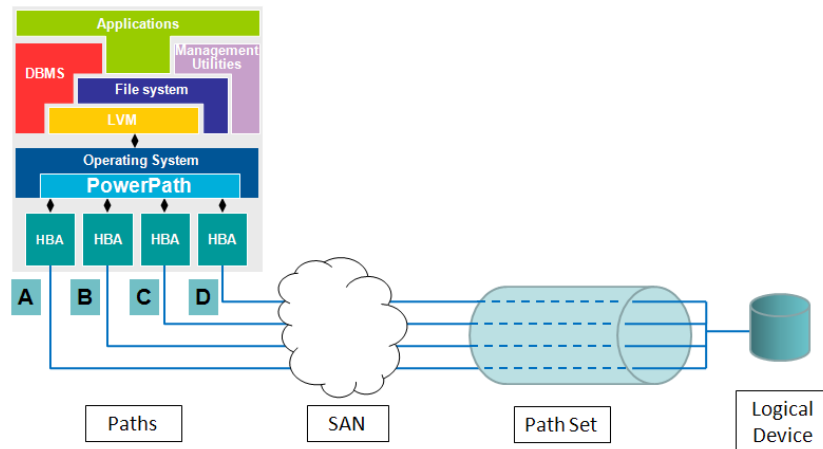


Figure 4. Multiple paths combined in a path set to the logical device

When attached to an active/active storage system (EMC and non-EMC), PowerPath creates a path set that can use any path in the set to service an I/O request. If a path fails, I/O is redirected to another viable path within the set. This redirection is transparent to the application, which is not aware of the error on the initial path. This avoids sending I/O errors to the application.

In an ALUA storage system, a path set will include the entire logical path connected to the active storage processor (active or optimized paths for ALUA), as well as to all the logical paths connected to the inactive storage processor (passive or non-optimized paths for ALUA). Once created, PowerPath can use any path in the set among the active paths to service an I/O request. If a path fails, PowerPath can redirect an I/O request from that path to any other active path in the set. Just as with active/active storage systems, this redirection is transparent to the application, which is not aware of the error on the initial path and therefore does not receive an error. If all the active paths in the set fail, PowerPath initiates a trespass and the passive paths become active (or non-optimized paths become optimized with ALUA). This trespass is also transparent to the application.

PowerPath comparison to native solutions

The following sections cover the key components of a multipathing solution as defined in the [Multipathing solution considerations](#) section.

Load balancing

PowerPath Multipathing License

PowerPath supports up to 32 paths from multiple HBAs (iSCSI TOEs or FCoE CNAs) to multiple storage ports when the multipathing license is applied. Without the multipathing license, PowerPath will utilize only a single port of one adapter. This is known as “PowerPath SE” or “unlicensed PowerPath.” In this mode, the single active port can be zoned to a maximum of two storage ports. This configuration provides storage port failover only and not host-based load balancing or host-based failover. This configuration is supported, but it is not recommended if the customer wants true I/O load balancing at the host and also HBA failover. PowerPath’s intelligent I/O routing and HBA failover can be achieved only when the software is licensed.

Note: PowerPath/VE for vSphere is not supported in an unlicensed mode.

PowerPath load balancing

PowerPath balances the I/O load on a host-by-host basis. It maintains statistics on all I/O for all paths. For each I/O request, PowerPath intelligently chooses, based on statistics and heuristics and the load-balancing and failover policy in effect, the most underutilized available path.

PowerPath’s optimized algorithms are not designed to perfectly balance the load on all paths in a path set. Because of varying I/O load and downstream SAN activity, it is unlikely that paths will have equal load. PowerPath will continue to use the same path with lightly sustained loads, so it may appear that there are I/O imbalances across the paths. Simply, equalizing the load across all available paths isn’t always beneficial to the host because all paths are not necessarily equal in day-to-day SAN activities. PowerPath aims to maximize performance and availability across all channels. This means that using the same path is an optimization that is preferable to equalization. Using a Round Robin policy can provide a less desirable result to overall host throughput because this policy seeks to equalize and not optimize, resulting in thrashing by the operating system. PowerPath considers all the I/O processing and bus capacity of all paths. A path never needs to be overloaded and slow while other paths are idle. Also, a busy path is never treated as equal to an idle path.

PowerPath has an automated load-balancing policy configuration mechanism. When the operating system boots and PowerPath is initialized, device-specific information on the managed LUNs is read by PowerPath. The optimized load-balancing policy is automatically set on a per-LUN basis. PowerPath supports EMC and certain qualified non-EMC arrays. If multiple arrays are accessible on the SAN and zoned to the

PowerPath host, each LUN that is masked to the server will be assigned the policy that provides the best load-balancing algorithm.

EMC has developed multiple load-balancing policies intended to support various customer requirements for production and test environments. PowerPath is recommended for simple and complex storage configurations. All customer types can benefit from the optimized policies. The Symmetrix[®] Optimized and CLARiiON[®] Optimized policies are the default policies for Symmetrix, and VNX[™] and CLARiiON devices, respectively. Use of other policies should be used only under the direction of EMC Customer Support. These policies are used by nearly all EMC customers for their Symmetrix, VNX, and CLARiiON arrays.

PowerPath has multiple algorithms. The comparisons in this paper with native multipathing focus on Symmetrix and CLARiiON Optimized modes.

- **Symmetrix Optimized** — EMC proprietary algorithm that is designed for past and present Symmetrix models. This policy is recommended for all Symmetrix implementations.
- **CLARiiON Optimized** — EMC proprietary algorithm that is designed for VNX and CLARiiON models. This policy is recommended for all VNX and CLARiiON implementations. Asynchronous Logical Unit Access (ALUA)¹ and non-ALUA modes are supported. This policy recognizes the difference between optimized and non-optimized paths as specified in the ALUA design.

For more information on other PowerPath policies, consult the *EMC PowerPath Product Guide*. The other policies are infrequently used. They are:

Adaptive, Least Block, Least I/O (also known as Least Queued I/O), Request, Round Robin, Streamlo, and Basic Failover

Because of the proprietary design and patents of PowerPath, the exact algorithm for these policies cannot be detailed here. However, this paper will explain the high-level functions of the policies and their interactions that contribute to its advanced load-balancing capability.

PowerPath's default policy for various storage types is also the optimized policy. This means that administrators do not have to change or tweak configuration parameters. This allows the administrator to spend time on other activities rather than on configuring multipathing options.

PowerPath selects a path for each I/O according to the load-balancing and failover policy for that logical device. PowerPath chooses the best path according to the chosen algorithm. The Multipathing Extension considers all possible paths for the I/O and selects the best one.

For every I/O, a weight is associated with all available and valid (not dead) paths for the I/O. Dead paths for the device are not considered. When all paths have been considered, PowerPath chooses the path with the lowest weight. The lower the

¹ ALUA simulates an active/active array. However, the optimized storage processor owns the LUN and I/O should be directed to it when possible. The non-optimized path should be used when the optimized paths are no longer available.

weight, the better the chance for the path to be chosen. Let's look more closely at PowerPath's optimized policies.

Symmetrix Optimized and CLARiiON Optimized use a number of factors in their calculations when selecting the best path for an I/O under normal, degraded, and faulted conditions in the SAN. The weight for path selection is based on the following:

- **Pending I/Os on the path** — PowerPath considers the amount of outstanding read or write requests waiting to access the storage array, which is directly impacted by queue depth.
- **Size of I/Os** — Physical and especially virtualized hosts could have multiple applications running with differing I/O characteristics. Depending on the application, the I/O size can vary from small to large. PowerPath's optimized algorithms naturally avoid small I/Os from being stuck behind large I/Os in the queue because it will select the least loaded path
- **Types of I/Os** — Pending reads and writes are considered. PowerPath weighs reads and writes differently because it takes into account array-specific capabilities with regards to the I/O profile.
- **Paths most recently used** — PowerPath will attempt to use the same path again if there isn't a less weighted path available. This avoids the need to continuously switch the path for every I/O.

On a VNX or CLARiiON operating in ALUA or active/passive mode, PowerPath will load balance the I/O among the *owning* storage processor ports. In an ALUA configuration, a LUN will be owned by either SPA or SPB. Each SP has multiple ports. (The exact number of ports is dependent on the VNX or CLARiiON model.) The owning SP ports are considered to be the optimized ports because those paths have the lowest latency. If the owning SP has four ports, then PowerPath will load balance across those four optimized paths. PowerPath will likely not use the non-owning (or non-optimized) paths unless there is extreme degradation in all optimized paths. In practice, the non-optimized paths are not used unless there is a trespass.

PowerPath hosts do not communicate with each other. Therefore, path decisions for a single host are not impacted by the path decisions on other hosts. However, activities of other hosts are indirectly taken into consideration. For example, if a path is degraded because of the other host activity or because it has a slower capability, I/Os will remain longer in the queue. PowerPath's optimized algorithms take that queue length into account and select the next best path.

Storage and server administrators have been impacted by a multitude of events in the SAN that have impacted application performance. Administrators have experienced a range of conditions in normal and faulted environments that impact I/O throughput. Examples include:

- Oversubscribed storage array ports

- Imbalanced I/O load on the storage array due to a variety of hosts with differing application I/O profiles zoned to the same storage ports
- Buffer-to-buffer credit starvation forcing a slowdown in SAN response time, which increases queue length
- Misbehaving HBA driver code causing repetitive fabric logins resulting in reduced application throughput on some ports
- ISL loss between Fibre Channel switches on Fabric A causes a drop in bandwidth between switches resulting in imbalance of I/O throughput compared with Fabric B
- Degrading fiber-optic infrastructure (for example, damaged fiber-optic cables or decaying laser optics) causing path instability
- And many more...

PowerPath's advanced load-balancing algorithms are designed to best utilize the highly available and redundant infrastructure built by storage and server administrators. SANs are naturally dynamic, which inevitably results in a change to path utilization. Application changes, adding or removing hosts, and faults can cause unbalanced I/O paths, leading to performance degradation. With the use of its optimized load-balancing algorithms, PowerPath can compensate and adjust for the dynamic changes in physical and virtual environments. Unlike other load-balancing solutions, the administrator may have to reconfigure paths, adjust path management parameters, and continue to reconfigure them as I/O traffic between the host and storage shift in response to usage changes.

Round Robin drawbacks

Round Robin is the default and most commonly used load-balancing policy for most native MPIO operating systems solutions. It is a very simple load-balancing algorithm. This policy has advantages over the basic failover policy because all paths are now capable of supporting I/O to the device simultaneously. For administrators, the ability to use all paths for load balancing and failover has been a huge benefit. However, as discussed in the previous section, not all paths are equal due to issues in the host, network, and storage array. To optimize available paths, administrators need to be concerned with how those paths are utilized.

Round Robin uses all paths in a static configuration. However, SANs are dynamic by design. Servers, applications, networking components, and storage arrays are added, removed, and modified intentionally and sometimes accidentally. A load-balancing policy must be able to act and react in this type of environment. Round Robin does:

- *Not* dynamically reroute I/O traffic
- *Not* identify any changes in network activity
- *Not* consider any changes in the storage array

- *Not* consider unique array properties
- *Not* track HBA queue depth
- *Not* consider I/O size

Round Robin has no intelligence behind its path routing. As long as a path is considered available, it has equal weight to all other paths unless the MPIO version has user-configurable settings for weighting paths. However, as noted in the previous sections, paths are not always equal. There is no such thing as a perfect or completely static SAN. Every data center changes and every data center is different. Some will experience more disruptions than others. Some will be designed with best practices in mind while some will not. Under normal operating conditions without any faults where the I/O is evenly distributed across the network and the array, Round Robin will work well enough to balance I/O. No matter how well the SAN is designed, problems will arise at some point.

Storage and server administrators want a load-balancing solution that works best and under all conditions and not just when the environment is optimal. Using all paths equally is not the same as optimizing those paths. PowerPath leverages the highly available, redundant SAN infrastructure to maximize I/O throughput by intelligently and dynamically routing application traffic without the need for user intervention. SAN architects design environments that are resilient, reliable, and predictable. The worst-case scenarios are always considered in planning. No one wants them to occur, but they must be planned for. With that in mind, architects don't design environments that are simply "good enough."

Whether the SAN is simple or complex, administrators will have to grapple with the expected and unexpected changes to greater or lesser degrees depending on the environment. With all the variables in the SAN, using PowerPath provides a level of path optimization and I/O availability that is not possible with Round Robin solutions. Let's focus the comparison on this policy.

Windows load balancing

In Windows 2003, Microsoft developed the MPIO framework, allowing third-party storage manufacturers to write software code to use this framework. This was not an independent load-balancing solution. Device Specific Modules (DSMs) are plug-ins to this framework. EMC developed a DSM that is MPIO-based. PowerPath is one solution that enables customers to have PowerPath features in the framework provided by Microsoft.

In Windows 2008, Microsoft developed a native multipathing product that uses the same framework with basic load-balancing and failover policies. In Windows 2008 R2, enhancements to multipathing include more policies. Microsoft allows the administrator to choose from one of the following policies²:

² Microsoft Windows 2008 R2 MPIO policies information can be found at <http://technet.microsoft.com/en-us/library/dd851699.aspx>.

- **Fail Over Only** – Policy that does not perform load balancing. This policy uses a single active path, and the rest of the paths are standby paths. The active path is used for sending all I/O. If the active path fails, then one of the standby paths is used.
- **Round Robin** – Load-balancing policy that uses all available paths in a balanced way. This is the default policy that is chosen when the storage controller follows the active/active. This would be the default for Symmetrix.
- **Round Robin with Subset** – Policy is similar to Round Robin. Load balancing that specifies a set of paths to be used in a Round Robin fashion, and with a set of standby paths. The DSM uses a standby path only when all primary paths fail. This policy is intended for arrays supporting ALUA. This would be the default for VNX and CLARiiON.
- **Least Queue Depth** – Load-balancing policy that sends I/O down the path with the fewest currently outstanding I/O requests. This policy is similar to PowerPath’s Least I/O.
- **Weighted Paths** – Load-balancing policy that assigns a weight to each path. The weight indicates the relative priority of a given path. The larger the number, the lower ranked the priority. The DSM chooses the least-weighted path from among the available paths.
- **Least Blocks** – Load-balancing policy that sends I/O down the path with the least number of data blocks currently being processed. This policy is similar to PowerPath’s Least Block.

RHEL load balancing

Device Mapper Multipathing (DM-MPIO) is the default multipathing solution for Red Hat. It includes support for the most common storage arrays that support DM-Multipath. This includes EMC arrays. The supported devices can be found in the `multipath.conf.defaults` file. The defaults in this file will automatically configure for specific array types. For example, a VNX will be set to ALUA mode. Round Robin is the default load balancing for Symmetrix VMAX™ and VNX. If your storage array supports DM-Multipath and is not configured by default in this file, you may need to add the array to the DM-Multipath configuration file `multipath.conf`. Keep in mind that the defaults aren’t necessarily the optimal settings for every customer environment.

For RHEL5 and RHEL6, the default algorithm is Round Robin, which has a parameter that specifies the number of I/O requests to route to a path before switching to the next path in the current path group. The default is 1,000, but this can be modified. The I/O profile of the host (for example, large or small block, random or sequential, read or write heavy) determines how this value should be set. The administrator determines the value.

RHEL6 provides two new path selector algorithms that determine which path to use for the next I/O operation:

- **Queue-length** – Algorithm looks at the amount of outstanding I/O to the paths to determine which path to use next.
- **Service-time** – Algorithm looks at the amount of outstanding I/O and the relative throughput of the paths to determine which path to use next.

Even though the new algorithms in RHEL6 provide more choices for administrators, Round Robin is still the default. Without an automated and optimized load-balancing policy or without documented best practices from the operating system vendor, configuration is more difficult.

Path testing on RHEL is performed by the *multipathd* daemon. Administrators can choose from multiple test modes. “Direct IO” is the default. However, there are several others to choose from depending on the array type. Path testing is every five seconds by default. This is also configurable by the user.

DM-MPIO does recognize the difference between the optimal and non-optimal paths with ALUA arrays. However, the default settings for path weights result in the non-optimal paths being used. Using the non-optimal paths increases I/O latency and completion times. This setting is also user-configurable.

VMware vSphere load balancing

By default, ESX® and ESXi™ provide an extensible multipathing module called the Native Multipathing Plugin (NMP). Generally, the VMware® NMP supports all storage arrays listed on the VMware storage HCL and provides a default path selection algorithm based on the array type. The NMP associates a set of physical paths with a specific storage device, or LUN. The specific details of handling path failover for a given storage array are delegated to a Storage Array Type Plugin. The specific details for determining which physical path is used to issue an I/O request to a storage device are handled by a Path Selection Plugin. SATPs and PSPs are sub-plug-ins within the NMP module.

VMware SATPs

Storage Array Type Plugins (SATPs) run in conjunction with the VMware NMP and are responsible for array-specific operations. ESX and ESXi offer an SATP for every type of array that VMware supports. These SATPs include an active/active SATP and active/passive SATP for non-specified storage arrays, and the local SATP for direct-attached storage. Each SATP accommodates special characteristics of a certain class of storage arrays and can perform the array-specific operations required to detect path state and to activate an inactive path. As a result, the NMP module can work with multiple storage arrays without having to be aware of the storage device specifics.

After the NMP determines which SATP to call for a specific storage device and associates the SATP with the physical paths for that storage device, the SATP implements the tasks that include the following:

- Monitors health of each physical path

- Reports changes in the state of each physical path
- Performs array-specific actions necessary for storage failover. For example, for active/passive devices, it can activate passive paths

VMware PSPs

Path Selection Plugins (PSPs) run in conjunction with the VMware NMP and are responsible for choosing a physical path for I/O requests. The VMware NMP assigns a default PSP for every logical device based on the SATP associated with the physical paths for that device. You can override the default PSP.

VMware NMP supports the following PSPs:

- **Most Recently Used (MRU)** - MRU selects the path the ESX/ESXi host used most recently to access the given device. If this path becomes unavailable, the host switches to an alternative path and continues to use the new path while it is available.
- **Fixed** - Fixed uses the designated preferred path, if it has been configured. Otherwise, it uses the first working path discovered at system boot time. If the host cannot use the preferred path, it selects a random alternative available path. The host automatically reverts back to the preferred path as soon as that path becomes available.
- **Round Robin** - Round Robin uses a path selection algorithm that rotates through all available paths, enabling load balancing across the paths. Round Robin was introduced in vSphere 4.0 after having experimental usage in ESX 3.5. Customers using vSphere will likely use this load-balancing policy.³

I/O operational limit

NMP Round Robin distributes I/O for a device down all active paths and uses a single path for a given number of I/O operations. The number of I/Os is specified with the “iooperationslimit” (iops=*X*) value. This is similar to RHEL. The default is 1,000, but this requires manual configuration. It is not automated. The I/O profile of the host (for example, large or small block, random or sequential, read or write heavy) determines how this value should be set. I/O load in a virtualized environment has varied load types in a single host. Also, the dynamic nature of virtual machines in a cluster does not guarantee that throughput will be constant. There is no optimization with this parameter. It is a static value that makes it difficult to keep pace with changes in the SAN.

HP-UX 11i v3 load balancing

In HP-UX 11i v2 and earlier, HP StorageWorks Secure Path provided load balancing and failover. It is no longer available in 11i v3 (11.31). Native multipathing is now

³ *VMware Fibre Channel SAN Configuration Guide*

included with the operating system. The default load-balancing policy for disk devices is Round Robin.

HP-UX 11i v3 supports the following load-balancing policies for disk devices:

- **Round Robin** – This default policy distributes I/O load equally across all active lunpaths irrespective of the current load on each lunpath.
- **Least command load** – This policy selects the lunpath with the least number of pending I/O requests for the next I/O operation.
- **Cell aware Round Robin** – This policy is applicable to servers supporting hard partitions that have high latencies for non-local memory access operations.
- **Closest path** – This policy selects the lunpath based on its affinity with the CPU processing the I/O operation so as to minimize memory access latency. This policy is more appropriate for cell-based platforms.
- **Preferred path, Preferred target port**– These two policies apply to certain types of targets that present an optimized/un-optimized controller model (different from active-passive).
- **Weighted Round-Robin** – This policy distributes the I/O load across all active lunpaths in a Round-Robin manner and according to the weight assigned to each lunpath.⁴

The administrator determines which policy is best for their environment. Some of these policies require manual tuning.

AIX load balancing

The path-control module (PCM) provides path management functions. The PCM Kernel Extension can provide more than one routing algorithm that can be selected by the user. The following are the algorithms for the device attributes for the AIX default PCMs:

- **Failover** - Sends all I/O down a single path. If the path is determined to be faulty, an alternate path is selected for sending all I/O. If the path being used to send I/O becomes marked failed or disabled, the next enabled path in the list is selected.
- **Round Robin** - Distributes the I/O across all enabled paths. The priority of the remaining paths is then recalculated to determine the percentage of I/O that should be sent down each path. If all paths have the same value, then the I/O is then equally distributed across all enabled paths.⁵

For both policies, path priority modifies the behavior of both algorithms. This attribute is configured by the administrator. For the Failover policy, the sequence of paths used is based on this attribute. For Round Robin, this attribute determines the percentage of I/O that is sent down each path. Therefore, a path is selected until it meets its required percentage.

⁴ *HP-UX 11i v3 Native Multi-Pathing for Mass Storage* white paper

⁵ [Multiple Path I/O page](#) on the IBM website

Path check is a user-configurable attribute. It can be enabled or disabled. When enabled, checks are performed on a fixed scheduled from 1 to 3,600 seconds.

No specific guidelines are available for these settings. The administrator decides what works best and makes appropriate changes as the environment changes.

Oracle Solaris load balancing

Native load balancing in Oracle Solaris is commonly referred to as MPxIO. In Solaris 10, enable MPxIO in the /kernel/drv/fp.conf file. In Solaris 8 and 9, enable MPxIO in the /kernel/drv/scsi_vhci.conf file. There are two load-balancing policies, failover and Round Robin, and they operate similarly to the other operating systems' policies. There are very few configuration parameters available. Customers are directed to documentation from storage array vendors for recommended settings.

Load-balancing performance comparison

The performance data outlined in this section reflects a specific set of tests. As with all performance testing, results may vary. PowerPath's optimized policy is compared to [Windows' MPIO Round Robin with Subset](#). The test is designed to demonstrate PowerPath's ability to reroute I/O to less busy paths when congestion is detected.

The host environment consisted of the following:

PowerPath host

- Operating system: Windows 2008 R2 host
- PowerPath version: 5.5 for Windows
- Load-balancing policy: CLARiiON Optimized (CLAROpt)
- Paths: 8 paths (4 paths to SPA, 4 paths to SPB)
- I/O generator: lometer

MPIO host

- Operating system: Windows 2008 R2 host
- Load-balancing policy: Windows Round Robin with Subset
- Paths: 8 paths (4 paths to SPA, 4 paths to SPB)
- I/O generator: lometer

Jamming hosts

- Operating system: 5 Windows 2008 R2 hosts
- PowerPath version: 5.5 for Windows
- Load-balancing policy: CLARiiON Optimized (CLAROpt)
- Paths: 1 (access to SPB0 only)
- I/O generator: lometer

Network

- Protocol: Fibre Channel
- Manufacturer: Brocade

Storage

- Array: CLARiiON CX4-240 running FLARE® 30
- Mode: ALUA (failover mode 4)

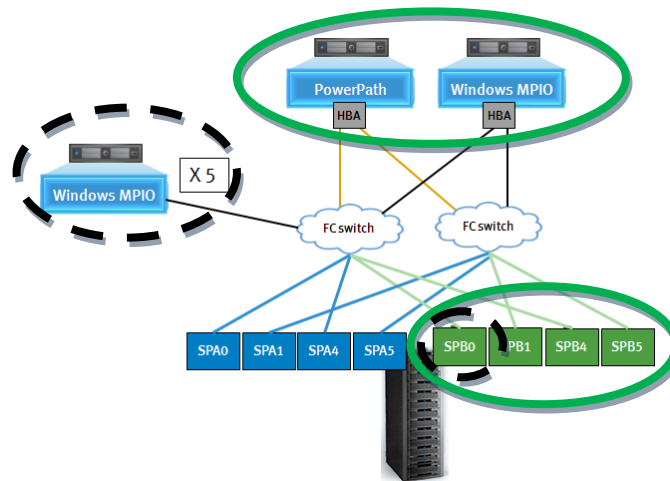


Figure 5. Physical layout of the test configuration

In this test, all LUNs are owned by Storage Processor B (SPB). Since SPB is the owning SP, ports B0, B1, B4, and B5 are the optimized paths. Therefore, all hosts were accessing their respective devices through SPB only. SPA is not being used.

The jamming hosts have access to SPB0 only (area in black in Figure 5) based on the number of physical connections and the zoning configuration. The idea here is to put a heavy load on one port of the owning SP. This simulates a common issue where one storage port is busier than others. This can occur on active/active, active/passive, and ALUA-type arrays for multiple reasons such as application I/O bursts, array imbalance due to misconfiguration of LUNs, and more. The following is a screenshot of the Navisphere® Analyzer Performance Summary showing the heavy load placed on SPB0.

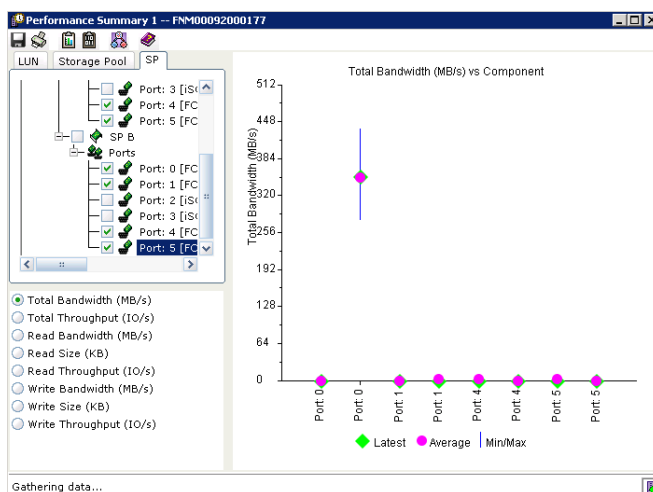


Figure 6. Congestion on Storage Port B0

The PowerPath 5.5 host has access to all SPB ports. This includes SPB0, B1, B4, and B5 (area in green in Figure 5). Based on PowerPath’s intelligent load-balancing algorithm, it will detect the congestion on SPB0. Even though the path from the HBA to SPB0 is a live path, it is not necessarily a desirable one. PowerPath routes more I/O to B1, B4, and B5. Windows MPIO Round Robin sees all paths as equal as shown in Figure 7.

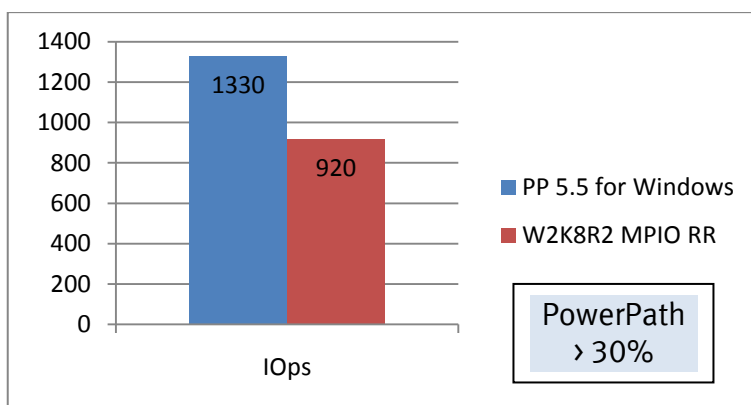


Figure 7. PowerPath’s advantage over MPIO Round Robin when there is port congestion

As a result, overall throughput on the MPIO host drops. Figure 7 shows the difference between PowerPath and MPIO for total IOPS and response time. PowerPath has a greater than 30 percent increase in IOPS when one of the storage ports is busy.

Path failover and recovery

Path failover is a feature of PowerPath that automatically redirects I/O from the failed path to an alternate path. This eliminates application downtime. Failovers are transparent and non-disruptive to applications. The following sections describe how failover occurs in different array environments.

There are several faults that can occur in the infrastructure that will result in a path failover. Figure 8 depicts the common fault points in the SAN that result in path failovers.

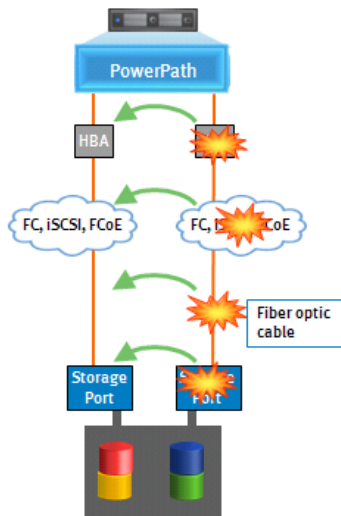


Figure 8. Points of failure in the I/O path

PowerPath enhances application availability by eliminating the I/O path as a point of failure. In more I/O-intensive configurations, there may be more HBAs and storage ports to accommodate the increased traffic. As a result, the number of points of failure increases.

- **HBA fault:** In an HBA failure, PowerPath fails the entire path over to an alternate HBA. In an ALUA or active/passive array, a trespass occurs if there is no path from an alternate HBA to the owning storage processor. When HBA comes back online, PowerPath recognizes this and resumes sending I/O through this HBA.
- **Network hardware or cabling fault:** When a hardware or firmware failure occurs in a Fibre Channel switch or other networking protocol device, PowerPath automatically uses the alternate paths to the logical device. Once the fault is correct, PowerPath automatically recognizes this and resumes sending I/O down that path.
- **Storage array port fault:** When the array port fails, PowerPath sends I/O to the remaining ports. In an ALUA array, PowerPath stops all I/Os to the failed optimized SP and trespasses the device over to the other SP. When the failed SP is coming back online, PowerPath recognizes it and resumes sending I/O to this SP. That is, in an ALUA (or active/passive model) array, PowerPath will trespass LUNs back to the default owning SP (also known as original optimized, or owning). This rebalances load across the SPs per the administrator's original configuration.

Active-active arrays

When a path component fails, PowerPath transparently redirects the I/O down the most suitable alternate path. PowerPath looks at the set of paths to the device, considers current workload and load balancing, and chooses the best path to send

the I/O down. It takes the failed path offline and redirects I/O through another path. Application will be unaware of the failure.

ALUA and active-passive arrays

Active-passive arrays

In an active-passive array like VNX and CLARiiON (when not set to ALUA mode), path failover is controlled by PowerPath in combination with array technology. For example, suppose that the paths to a LUN through SP B are active paths. The application I/O is balanced across all ports on SP B. The paths to the device through SP A are considered passive and hence do not support I/O.

ALUA arrays

In ALUA arrays like VNX and CLARiiON, path failover is controlled by PowerPath in combination with array technology. If the paths to a LUN through SP B are optimized paths, then the application I/O is balanced across all ports on SP B. The paths to the device through SP A are considered non-optimized and hence do not support I/O.

When configuring LUNs on ALUA or active/passive arrays, administrators specify which SP is the default owner of a LUN. By setting the default owner of some LUNs to SP A and some to SP B, they manually divide the I/O load between the two SPs. When a failed path is restored, PowerPath will automatically change the LUN ownership back using an explicit trespass to the default owner(s) to restore the administrator's desired LUN distribution.

Performance can be negatively impacted if the multipathing software does not support ALUA. Using the non-optimized path can increase latency, increase response time, and decrease application throughput. Since PowerPath supports ALUA, it will not send I/O to the non-optimized path unless the optimized paths become severely degraded. Many native MPIO products understand the difference between an optimized and a non-optimized path. However, knowledge of ALUA has no impact on optimized path performance. So, even though native MPIO solutions support ALUA, they still do not have the optimized load-balancing policies supported by PowerPath.

When PowerPath detects a pending Non-Disruptive Upgrade (NDU) on a VNX or CLARiiON, it proactively trespasses all LUNs active on that upgrading SP to the alternate SP. This saves host and application processing time by proactively failing over rather than waiting for I/Os to fail.

In the case of a hung or failed SP, PowerPath will trespass all LUNs to the alternate SP but this is not done proactively. That is, once PowerPath identifies that the SP is no longer reachable, it will trespass all LUNs to the remaining SP.

PowerPath has no control of the I/O until it gets the I/O back from the native SCSI driver. This is important to understand since the native driver may have its native recovery mechanisms as well as the HBA driver that sits below it. An I/O comes back to PowerPath with an error status if it has failed the HBA recovery mechanisms and the native driver recovery mechanisms. In other words, it means that the I/O was

probably retried a couple of times without success by these layers that always use the same path.

If a path fails, PowerPath redistributes I/O traffic from that path to functioning paths while stopping I/O to the failed one. PowerPath then begins to test the failed path periodically to identify if the path has recovered and can then be restored to the path set and active duty. These path failover and failover recovery processes are transparent to applications. (Occasionally, however, there might be a delay. For example, there may be HBA driver retries.) The [Path testing](#) section covers in more detail how path states are identified.

Proactive path testing on failures

PowerPath has a more intelligent response capability when a path fault occurs. This is a key feature to understanding how PowerPath responds to faults as opposed to other MPIO solutions. When a single fault occurs in the SAN, multiple components may be impacted as a result of the shared nature of environment. PowerPath has a concept of common path elements. When a failure occurs, PowerPath avoids a shared failed element in the path to ensure faster failover time compared with native MPIO solutions. Let's take the following example to demonstrate PowerPath's failover capability.

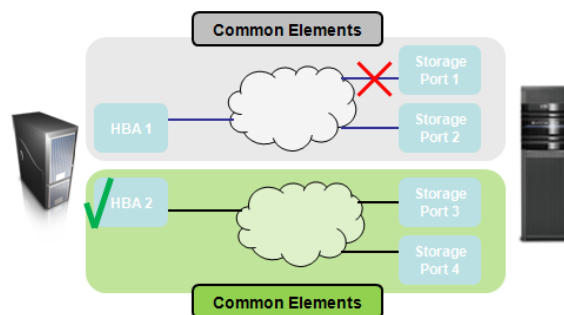


Figure 9. A redundant configuration highlighting the common elements concept

In dual fabric configurations, HBAs and storage ports are physically connected to only one fabric. This builds fault tolerance into the SAN, ensuring high availability. As a result, there are common elements and uncommon elements in the design. The common elements are physical and logically (through zoning) connected within a fabric. There is no association among the elements in two fabrics. That makes them uncommon elements because they cannot cross fabric lines.

Figure 9 shows two distinct fabrics. Each fabric has a set of common elements. The top fabric has HBA1 and Storage Ports 1 and 2. The bottom fabric has HBA2 and Storage Ports 3 and 4. In normal operating conditions, PowerPath will load balance across all available paths. When a failure occurs on a path to Storage Port 1 while I/Os are in flight, an error is returned to the host. It goes up the stack to PowerPath. PowerPath performs the following actions:

1. PowerPath immediately marks the path for testing.

2. PowerPath avoids issuing I/O on any path marked for a path test. Paths marked for testing are tested when the path test process runs in the next cycle.
3. Application I/Os are no longer sent to this path until the result of the path test is known.
4. PowerPath identifies all common elements associated to the failure and marks those paths for testing as well. These associated paths are marked as deferred paths and have the highest priority for testing. It is important to quickly identify paths that are still alive, so overall I/O bandwidth is not unnecessarily reduced for a longer duration than necessary. (The deferred state is not visible to the user.)
5. Within milliseconds following the failure yet before the path test is complete, PowerPath has not yet identified exactly what failed. Was it the HBA? The storage port? PowerPath will not hold I/Os when other viable paths are available.
6. Until PowerPath has identified whether it is an HBA or storage port failure, it makes the intelligent decision to send the next I/Os to a set of uncommon path elements. In this case, the next I/Os will use HBA 2.
7. Even though Storage Port 2 is available, PowerPath takes the proactive step to test all paths that have commonality to the original failure.

As described above, when a path fails due to an I/O error, PowerPath marks for testing all paths related to the failed one. Until these related paths are tested, PowerPath avoids selecting them for I/O. This is an important capability because native MPIO solutions may continue to send more I/Os to potentially failed components. The inability of other MPIO products to optimize this process can potentially cause sending I/O to a failed path, which in turn causes timeout-and-retry delays throughout the entire I/O subsystem (application, operation system, fabric, and array).

Path testing

PowerPath Multipathing Extension protects applications from path failures. It is designed to provide the highest level of data access through optimized load balancing and intelligent path failover and recovery. In order to provide these services, PowerPath must know the state of all paths at all times. Therefore, PowerPath maintains a daemon that periodically tests the paths that it manages.

To determine whether a path is operational, PowerPath uses a path test. A path test is a sequence of I/O requests, specifically issued by PowerPath, to determine the viability of a path. The “multipath periodic path testing” daemon probes the paths to devices so that if there is an issue with a path, it will be detected before an application sends an I/O. Also, if a dead path changes its status to alive, this path is available for sending I/O. PowerPath performs bus and path tests. A bus test ascertains the availability of an I/O path from the HBA to the storage port. A path test ascertains the availability of the full I/O path from the HBA to the LUN as shown in Figure 10. Bus tests are important because they help find HBA and storage port

issues faster without increasing the overhead on the host. Individual path tests are needed to identify the path state to a specific LUN. That could be costly to the OS for a host with a hundred LUNs and multiple redundant paths to them. However, PowerPath's path intelligent testing is load-based, which reduces unnecessary testing. This is described in more detail next.

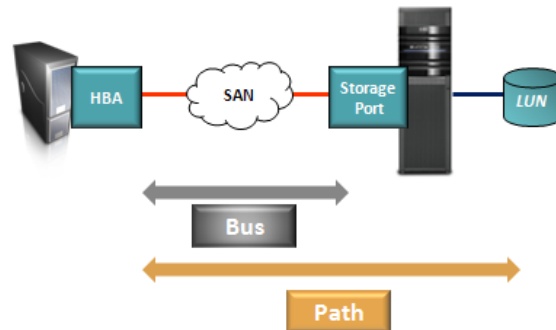


Figure 10. Difference between a bus and a path

PowerPath is more intelligent than native MPIO solutions when it comes to path testing. PowerPath performs load-based path tests. It is designed to be dynamic and meet the needs of simple hosts with a few paths as well as I/O-intensive hosts that virtualize multiple applications with hundreds of devices with a thousand paths. PowerPath does not waste host resources by testing on a static schedule. If a host is busy and application I/O is successfully completing, there is no need to repeat a test on that same path to the device.

Idle buses and paths

PowerPath tests live and idle paths periodically to identify failed paths. Storage administrators may assign additional LUNs to a host for future usage. Also, LUNs may be inactive due to a low access rate by the application. In either case, the path to PowerPath-managed LUNs is still tested. The goal of testing live yet idle paths is to prevent application I/O from being issued on dead paths that PowerPath would not have detected otherwise. This testing helps reduce timeout and retry delays. As mentioned above, in an active system, with few idle paths, live paths are rarely tested.

Periodic testing of live paths is a low-priority task. It is not designed to test all paths within a specific time, but rather to test all paths within a reasonable time, without taxing system resources and interfering with application I/O.

Dead buses and paths

If a path test fails, PowerPath disables the path and stops sending I/O to it. After the failure, PowerPath continues testing it periodically to determine if it has been repaired during the test interval. If the path passes a test, PowerPath restores it to service and resumes sending I/O to it. The array, host, and application remain available while the path is being restored. This is PowerPath's autorestore capability. It is not designed to restore a path immediately after it is repaired, but rather to

restore the path within a reasonable time after its repair. Like periodic testing of live paths, periodic autorestore is a low priority.

Note: The fastest way to restore paths is for the administrator to run the *powermt restore* command.

When an I/O error is returned by the HBA driver, PowerPath marks a path to be tested by the periodic test. As described earlier, PowerPath marks both the path with the error and related paths (for example, those paths that share an HBA and FA port with the failed path) for testing. Meanwhile, PowerPath reissues the failed I/O on another path.

When are path tests performed

PowerPath tests a path in the following scenarios:

- PowerPath periodically runs the path test process. This process sequentially visits each path and tests it if required:
 - Live paths are tested periodically.
 - Live buses are tested periodically.
 - Dead paths are tested periodically.
 - Dead buses are tested periodically.
- When a new path is added, it must be tested before being brought in service.
- When PowerPath is servicing an I/O request and there are no more live paths to try. PowerPath always tries to issue application I/O, even if all paths to the target logical device are dead when the I/O request is presented to PowerPath. Before PowerPath fails the I/O with an error condition, it tests every path to the target logical device. Only after all these path tests fail does PowerPath return an I/O error
- When the administrator runs **powermt load**, **powermt restore**, or **powermt config** (UNIX and Linux only). These commands issue many path tests, so the state of many paths may change as a result of running the commands.

How often are paths tested

The time it takes to do a path test varies. However, tests are spaced out, so at least one path on every HBA and every port is tested often. Due to the dynamic nature of PowerPath path testing and the number of paths that may need testing, it is difficult to predict exactly when a path will be tested. Path testing is completely non-disruptive to application I/O.

Testing non-optimized paths

As discussed in the [Load balancing](#) section, PowerPath supports ALUA arrays (for example, VNX). That is, PowerPath recognizes the difference between the optimized and non-optimized paths. Under normal operating conditions, only the optimized paths are used for I/O. The optimized paths can be identified easily. In the case of

VNX (or supported non-EMC arrays), the ports of the storage processor that owns the LUN are considered the optimized paths. If the owning SP fails, the LUNs trespass to the other storage processor. The paths on the other storage processor are promoted from non-optimized to optimized paths. Figure 11 shows LUN1 is owned by Storage Processor A, which makes those ports the optimized paths. As seen, PowerPath tests all paths to the LUN.

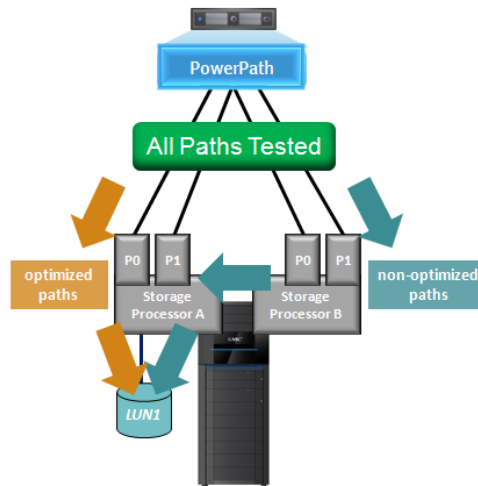


Figure 11. PowerPath path testing on a VNX array

Administrators want to know that when a failure does occur that the paths are available. Discovering that failover paths are not available at the time of a failure usually results in data unavailability. PowerPath performs path test operations on both the optimized and non-optimized paths. If the non-optimized paths are dead for whatever reason, the administrator will know about the issue through the PowerPath UI. Corrective action can be taken before an unexpected failure or scheduled maintenance (for example, NDU). Native MPIO solutions that do not provide this level of testing increase the chances of data unavailability when failures occur.

Virtualized and physical environments

Data centers have begun adopting virtualization to improve the efficiency and availability of IT resources and applications. Virtualization technology has been implemented on the host, in the network, and in the storage array. On the host, virtual machines may take the place of 10, 20, or more physical servers. In the network, virtual LANs and logical SANs have abstracted the layer 2 networking. In the storage array, virtual device pools provide flexibility for data to be moved among tiers.

With the consolidation of physical servers and their native applications onto a single platform, the I/O characteristics of the virtualized server become even more varied. Administrators expect to get the most out of data paths in a dynamic virtual environment. The I/O data path is no different from any other part of the virtual SAN. It must be optimized in order to maximize utilization for the virtual machine applications.

PowerPath/VE for vSphere and for Hyper-V are part of the PowerPath family of software products. They are designed to optimize the data paths on a virtual host in much the same way as PowerPath does on physical hosts. The multipathing algorithms for PowerPath and PowerPath/VE are identical.

By nature, a virtualized environment has an increased level of complexity yet at the same time requires a greater degree of flexibility. With more applications running on one server, there is a greater likelihood of varying I/O profiles with different throughput characteristics. Combined with the flexibility of free-flowing virtual machines from host to host within a cluster, static load patterns on the SAN and the storage ports are not guaranteed to be the same from day to day. Native multipathing solutions have simpler load-balancing algorithms that are not optimized to handle this kind of I/O traffic.

PowerPath/VE reduces complexity. There is no need to qualify PowerPath/VE with hypervisor capabilities. It manages all of the complexity of complicated device mapping and features like vMotion®, Distributed Resource Scheduler, and VAAI. Also, the process of qualify PowerPath/VE with a hypervisor is easy. There is no need to qualify PowerPath/VE with the individual guest operating systems because it resides at the hypervisor layer.

Individual virtualized hosts have complex I/O profiles. Clustered virtualized hosts add variability in load patterns for those complex profiles. When added to a SAN that has tens, hundreds, or thousands of virtual and physical hosts, the potential problems are magnified by several orders of magnitude. A level of I/O stability can be achieved by using a multipathing solution that cuts across all operating systems. That way, administrators can expect the same level of path optimization, predictability, and reliability with PowerPath/VE for virtualized servers as they have come to expect with PowerPath.

Conclusion

EMC PowerPath and PowerPath/VE provide automated and optimized load balancing, fast and non-disruptive path failover and failback, and intelligent path testing. PowerPath has over 30,000 unique customers and over 1 million licenses sold. It has established itself as the premier multipathing solution for SAN and server administrators in small and large SAN environments. Over the years, operating system vendors have either introduced native multipathing or improved upon existing multipathing. Each vendor has a unique solution that introduces challenges to overall SAN load balancing and path management. Because PowerPath operates the same across platforms in normal and fault conditions, the SAN becomes more stable and predictable. With auto-configured, optimized load-balancing policies, administrators do not have to spend valuable time deciding which is the best policy to implement for each platform. With an optimized policy, PowerPath adapts to changes in the SAN without the need to reconfigure when I/O profiles change, virtual machines are added and removed, or new storage is added. PowerPath continues to make improvements in management, operating system integration, and advanced capabilities.