StorageSpaces module for Windows PowerShell

By Bruce Langworthy

The StorageSpaces module is a PowerShell module which utilizes the Storage module for Windows PowerShell to provide a streamlined management experience for Storage Spaces. This module provides the ability to easily manage Storage Spaces for single-machine deployments, as well as providing cluster-aware management when using Storage Spaces in a Windows Failover Cluster.

# Table of Contents

[Table of Contents 2](#_Toc335107800)

[Table of Figures: 3](#_Toc335107801)

[Why StorageSpaces module? 4](#_Toc335107802)

[Cmdlets contained in the StorageSpaces module module: 6](#_Toc335107803)

[Obtaining and installing the StorageSpaces module module 7](#_Toc335107804)

[Where do I download StorageSpaces module from? 7](#_Toc335107805)

[Installation Steps for StorageSpaces module 7](#_Toc335107806)

[Obtaining help for cmdlets in the StorageSpaces module module: 8](#_Toc335107807)

[The StorageSpaces module module does not contain “built-in” help. Instead this content is provided on the Internet, and may be updated via PowerShell from a computer which has Internet access. 8](#_Toc335107808)

[Steps for building a Failover Cluster with Spaces 9](#_Toc335107809)

[Understanding the cmdlets and capabilities provided by the StorageSpaces module module 11](#_Toc335107810)

[Management of a single computer with StorageSpaces module 11](#_Toc335107811)

[Management of Storage Spaces on a Failover Cluster with StorageSpaces module 12](#_Toc335107812)

[Examples of StorageSpaces module Functionality 12](#_Toc335107813)

[Single-Machine deployment of Storage Spaces 13](#_Toc335107814)

[Deploying a Failover Cluster with Storage Spaces 15](#_Toc335107815)

[Creating a detailed configuration report for Storage Spaces 17](#_Toc335107816)

[Using Get-SpacesVolume 19](#_Toc335107817)

[Checking overall pool health 22](#_Toc335107818)

[Reviewing reliability data for physical disks in a Storage Spaces pool. 23](#_Toc335107819)

[Enabling advanced logging of changes to the Storage Spaces state; 24](#_Toc335107820)

[Additional Resources: 26](#_Toc335107821)

[Appendix A: Terminology for Storage Management 27](#_Toc335107822)

# Table of Figures:

[Figure 1: Create and format a Storage Space using the Storage module for Windows PowerShell 5](#_Toc335107882)

[Figure 2: Create and format a Storage Space using the StorageSpaces module module for Windows PowerShell. 5](#_Toc335107883)

[Figure 3: Complete list of cmdlets contained in the StorageSpaces module module. 6](#_Toc335107884)

[Figure 4: Example of deploying a Failover Cluster with Storage Spaces using StorageSpaces module 9](#_Toc335107885)

[Figure 5: Creating a Cluster Shared Volume (CSV) on a clustered Storage Spaces pool. 10](#_Toc335107886)

[Figure 6: Using StorageSpaces module with Single-Machine configurations 11](#_Toc335107887)

[Figure 7: StorageSpaces module Cluster-Aware cmdlets functionality list. 12](#_Toc335107888)

[Figure 8: Single-Machine deployment of Storage Spaces using StorageSpaces module. 13](#_Toc335107889)

[Figure 9: Using New-Spaces Volume with the -Verbose switch 14](#_Toc335107890)

[Figure 10: Deployment of Storage Spaces with Failover Clustering using StorageSpaces module 15](#_Toc335107891)

[Figure 11: Example of Get-SpacesConfiguration cmdlet output 17](#_Toc335107892)

[Figure 12: Listing the Physical Disks for a specific Storage Pool 18](#_Toc335107893)

[Figure 13: Example output from Get-SpacesVolume 20](#_Toc335107894)

[Figure 14: Get-SpacesVolume on a Clustered Storage Space 21](#_Toc335107895)

[Figure 15: Test-SpacesConfiguration showing a healthy pool 22](#_Toc335107896)

[Figure 16: Test-SpacesConfiguration showing an unhealthy pool. 22](#_Toc335107897)

[Figure 17: Example of Get-SpacesPoolPhysicalDiskHWCounter output 23](#_Toc335107898)

[Figure 18: Example Event log containing an entry showing a healthstatus change for a Storage Space 24](#_Toc335107899)

[Figure 19: Event Log showing removal of a Physical Disk was removed from the system 25](#_Toc335107900)

[Figure 20: Terminology glossary 27](#_Toc335107901)

# Why StorageSpaces module ?

A number of people have asked me, “Why the StorageSpaces module , why not just use the Storage module to manage Storage Spaces?”

The answer to this question is that you can certainly use either approach, but using StorageSpaces module provides these additional benefits;

* StorageSpaces module is task-oriented, and is designed to complete configuration workflows with a minimum number of steps.
* Deployment and Management via StorageSpaces module is possible without requiring a complete understanding of the Management API in order to manage and deploy Storage Spaces.

For example, consider the end to end deployment of a Storage Space after a pool is created. The associated disk must be initialized, a partition created, and the volume formatted. While this is certainly achievable using cmdlets from the Storage module, it requires an understanding of the functionality of the Storage Management API, and all associated cmdlets in order to complete this scenario.

So, for example, creating a Storage Space which is a mirror, which is 20GB in size, and formatted as Volume Y, would require the following script in order to achieve this goal when using the **Storage** module which is built into Windows Server 2012 and Windows 8:

For an example of deploying a failover cluster using **StorageSpaces module,** please refer to the Management Storage Spaces on a Failover Cluster with StorageSpaces module section of this document.

Note: Both examples assume that a Storage Pool named “Internal” has already been created and that DriveLetter E is currently available to be assigned.

Figure : Create and format a Storage Space using the Storage module for Windows PowerShell

**# Create a New Storage Space which is mirrored, thinly provisioned, and 20GB in size.**

# New-VirtualDisk -FriendlyName MirrorTest -StoragePoolFriendlyName Internal   
# -ResiliencySettingName Mirror -ProvisioningType Thin -Size (20GB)

**# Find out which Disk object was created by the Storage Space we just created.**

Get-VirtualDisk -FriendlyName MirrorTest | Get-Disk

**# Assuming this was disk 6;**

Initialize-Disk -Number 6

**# Create a partition and assign a drive letter**

New-Partition -DiskNumber 6 -UseMaximumSize -AssignDriveLetter

**# Format the volume**

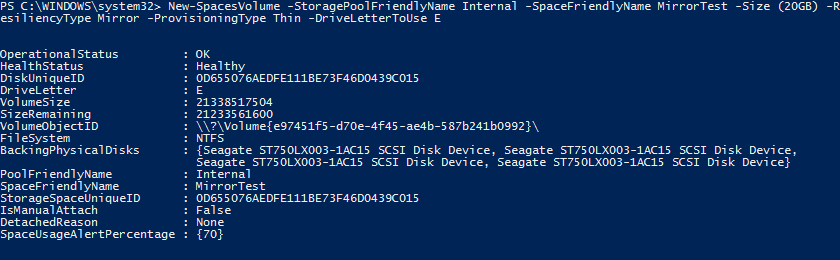
Format-Volume -DriveLetter E -NewFileSystemLabel MirrorTest

I can accomplish the same workflow as above, using a single command in the **StorageSpaces module**;

**# Create a Mirror Storage Space that is thinly provisioned, and format a   
# volume on the Storage Space and use letter E on the new Volume.**

New-SpacesVolume -StoragePoolFriendlyName Internal -SpaceFriendlyName   
 MirrorTest -Size (20GB) -ResiliencyType Mirror -ProvisioningType Thin -  
 DriveLetterToUse E

Figure : Create and format a Storage Space using the StorageSpaces module for Windows PowerShell.



## Cmdlets contained in the StorageSpaces module:

The StorageSpaces module contains the following cmdlets for Windows PowerShel.l For additional information on the capabilities of these cmdlets, please refer to the Understanding the cmdlets and capabilities provided by the StorageSpaces module section of this document.

Figure : Complete list of cmdlets contained in the StorageSpaces module .

CommandType Name ModuleName

----------- ---- ----------

Function Get-AvailableDriveLetter StorageSpaces

Function Get-SpacesConfiguration StorageSpaces

Function Repair-SpacesConfiguration StorageSpaces

Function Test-SpacesConfiguration StorageSpaces

Function Get-SpacesPhysicalDisk StorageSpaces

Function Get-SpacesPool StorageSpaces

Function New-SpacesPool StorageSpaces

Function Get-SpacesPoolPhysicalDiskHWCounter StorageSpaces

Function Get-SpacesProvider StorageSpaces

Function Get-SpacesSubsystem StorageSpaces

Function Get-SpacesVolume StorageSpaces

Function New-SpacesVolume StorageSpaces

Function Resize-SpacesVolume StorageSpaces

Function New-StorageSpacesEventLog StorageSpaces

Function Remove-StorageSpacesEventLog StorageSpaces

Function Get-StorageSpacesMpioConfiguration StorageSpaces

Function Disable-StorageSpacesMpioSupport StorageSpaces

Function Enable-StorageSpacesMpioSupport StorageSpaces

# Obtaining and installing the StorageSpaces module

## Where do I download the StorageSpaces module from?

The SpaceCommmand module is available for download on the Microsoft Script Repository. You can download the module here:

TODO: Link once posted

## Installation Steps for StorageSpaces

* Create the following directory :
  + C:\Windows\System32\WindowsPowerShell\v1.0\Modules\StorageSpaces
* Unzip the files from the attached Zip to the directory above.
* Run the command Unblock-File against the 3 files contained in the ZIP file for Space Command;
  + StorageSpaces.psd1
  + StorageSpaces.psm1
  + Notification\_Script.Ps1
* Set the Windows PowerShell execution policy to RemoteSigned;
  + Set-ExecutionPolicy RemoteSigned
  + Note: This step is only required in order to use the New-StorageSpacesEventLog or Remove-StorageSpacesEventLog cmdlets.
* Open an admin PowerShell window, and type Import-Module StorageSpaces to get started.
* To enable remote management to other computers, and to support management of remote cluster nodes, the command Enable-PsRemotingmust be run to enable PowerShell remote management on **all** cluster nodes or remote machines to manage.
* For a complete list of commands, type Get-Command –Module StorageSpaces

## Obtaining help for cmdlets in the StorageSpaces module:

The StorageSpaces module does not contain “built-in” help. Instead this content is provided on the Internet, and may be updated via PowerShell from a computer which has Internet access.

Once a PowerShell module has been imported into the current PowerShell session, the help content may be downloaded and updated via the following command:

Update-Help

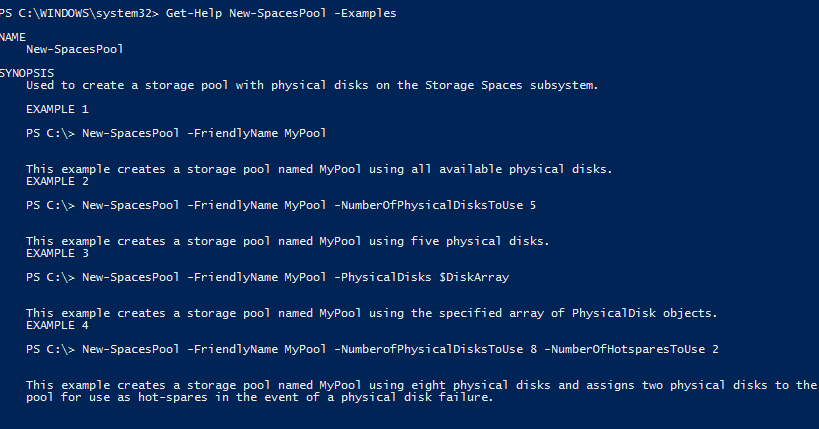
Note: the –Force parameter must be used if attempting to update more than once per 24 hour period.

The get-help cmdlet offers multiple levels of verbosity, which include

* No switch specified, only basic help is returned.
* Detailed provides detailed help
* Example provides examples of the cmdlet in use.
* Full provides all available help content for the specified cmdlet.

For example, in order to obtain script examples for the New-SpacesPool cmdlet, the following command is utilized:

Get-Help New-SpacesPool –Examples



## Steps for building a Failover Cluster with Spaces

Using the Module; Creating a Failover Cluster and storage spaces pool can be automated to the following 5 commands: (Assumes Failover Clustering feature is already installed and an adequate number of blank SAS disks for pool creation).

The workflow below is designed for creation of a new Cluster from scratch with a cluster quorum hosted on a Storage Space.

The New-SpacesPool utilizes PS remoting to build a list of Physical disks seen by both cluster nodes, and uses this list as the input for pool creation.

Figure : Example of deploying a Failover Cluster with Storage Spaces using StorageSpaces

**# Create a new Failover Cluster, and skip the Storage validation as we have   
# not configured storage yet.**

New-Cluster –Name DeepSpaceClu –Node Node1,Node2 -NoStorage

**# Create a new pool with Storage Spaces using only Physical Disks which are shared   
# (visible) to both cluster nodes**

New-SpacesPool –FriendlyName SpacesPool –PhysicalDisks (Get-SpacesPhysicalDisk   
 –Servertocompare Node1,Node2)

**# Create a Mirrored Storage Space to use as the cluster Quorum.**

New-SpacesVolume –StoragePoolFriendlyName SpacesPool –SpaceFriendlyName Quorum   
 –Size (1GB) –ResiliencyType Mirror –ProvisioningType Fixed

**# Add the new Storage Space as a Cluster Disk**

$AddedDisk = (Get-ClusterAvailableDisk | Add-ClusterDisk)

**# Set the newly added cluster disk as the Quorum**

Set-ClusterQuorum –NodeAndDiskMajority $AddedDisk.Name

After running the above commands, a Storage Space + Provisioned Disk + Cluster CSV resource can be created in one command using the following cmdlet:

**# Create a Storage Space, Provision the Disk, and Create a Cluster Shared Volume**

New-SpacesVolume –StoragePoolFriendlyName SpacesPool –SpaceFriendlyName MyMirrorCSV –Size (5GB) –ResiliencyType Mirror –ProvisioningType Fixed -CreateClusterSharedVolume

For detailed information about the current configuration of the pool and all storage spaces, use the Get-SpacesConfiguration cmdlet:

Get-SpacesConfiguration–StoragePoolFriendlyname SpacesPool

# Understanding the cmdlets and capabilities provided by the StorageSpaces module

The following two tables describe the functionality provided by the StorageSpaces module for both single computer and failover clustering configurations.

## Management of a single computer with the StorageSpaces module

Management of Storage Spaces on a single computer can be performed using the following cmdlets provided by the StorageSpaces module.

Figure : Managing StorageSpaces with Single-Machine configurations

|  |  |
| --- | --- |
| Cmdlet Name | Functionality Provided |
| Get-AvailableDriveLetter | Returns a list of either all available drive letters, or only the next letter in alphabetic order. |
| Test-SpacesConfiguration | Based on the name of a Storage Pool, this cmdlet reports whether any associated objects to the pool are unhealthy. |
| Repair-SpacesConfiguration | Performs repairs on configuration issues with Storage Spaces, such as re-attaching a pool after a clean installation of Windows Server, and requesting repairs to Storage Spaces after a drive failure or replacement. |
| Get-SpacesConfiguration | Returns detailed information about the state of a pool, all associated Storage Spaces, and their associated volumes. |
| Get-SpacesPhysicalDisk | Returns a list of PhysicalDisks which are available to Storage Spaces. Provides either a complete list of shared physical disks, or optionally, only those available for pool creation. |
| Get-SpacesPool | Returns detailed Pool information about Storage Pools created with Storage Spaces.  Note: Storage Pools created by other storage providers are not displayed. |
| New-SpacesPool | Allows easy creation of a new storage pool. |
| Get-SpacesPoolPhysicalDiskHWCounter | Reports hardware reliability data for all physical disks associated with a given storage pool. For example, temperature and read/write error counts. |
| Get-SpacesProvider | Returns the Storage Spaces storage provider for use with other cmdlets from the Storage module. |
| Get-SpacesSubsystem | Returns the storage subsystem for Storage Spaces for use with other cmdlets from the Storage module. |
| Resize-SpacesVolume | Resizes (increases) the size of both a Storage Space, and the FileSystem contained on the Storage Space. |
| New-SpacesVolume | Creates a Storage Space from an existing pool, initializes the disk, creates a partition, and formats the volume for use. |
| Get-SpacesVolume | Returns detailed information about a Storage Space, as well as associated Disk and Volume objects. |
| New-StorageSpacesEventLog | Enables detailed logging of configuration changes to Storage Spaces. |
| Remove-StorageSpacesEventLog | Removes existing logging, and disables future logging of changes to Storage Spaces unless the New-StorageSpacesEventLog cmdlet is again run. |
| Get-StorageSpacesMPIOConfiguration | Returns the current state of MPIO and whether it is configured optimally for Storage Spaces. |
| Enable-StorageSpacesMPIOSupport | Enables the MPIO Feature, and performs optimal configuration of MPIO for use with Storage Spaces. |
| Disable-StorageSpacesMPIOSupport | Prevents MPIO from claiming SAS disks, and clears the default Load Balance policy configured for disks with MSDSM. Does not uninstall the MPIO feature. |

## Management of Storage Spaces on a Failover Cluster with StorageSpaces

When used in conjunction with a Storage Spaces + Failover Clustering deployment, these cmdlets provide all of the functionality listed above, plus the following “Cluster-Aware” management functionality:

Figure : StorageSpaces Cluster-Aware cmdlets functionality list.

|  |  |
| --- | --- |
| Cmdlet | Description |
| Get-SpacesConfiguration | Remotely enumerates details about Storage Spaces owned by other cluster nodes. |
| Get-SpacesPhysicalDisk | Allows remotely running this command against N number of cluster nodes to produce a list of physical disks, where they are only listed if they are visible to all remote machines for creation of clustered storage pools with Storage Spaces. |
| Get-SpacesPool | Returns Pool information about Storage Pools created with Storage Spaces . |
| Resize-SpacesVolume | Will remotely perform resize of Storage Space and volume against the appropriate cluster node owner for the Storage Space specified. |
| New-SpacesVolume | If the Storage Pool is clustered, allows creating a Cluster Shared Volume (CSV) in addition to normal creation of a Storage Space + Formatted Volume. |
| Get-SpacesVolume | Additionally, if the Storage Space is clustered, information about the cluster node owner, and current cluster resource state are also included. (Requires the –Cluster switch on Get-SpacesVolume) |

# Examples of StorageSpaces module Functionality

StorageSpaces is a Windows PowerShell module that utilizes the **Storage** module in Windows Server 2012 and Windows 8 to provide an easier management and maintenance experience for Storage Spaces.

The StorageSpaces module is intended both to simplify management of Storage Spaces via Windows PowerShell, as well as to serve as an example of how to script storage management.

## Single-Machine deployment of Storage Spaces

The following script example performs the following tasks;

1. Creates a pool named **SpacesPool** using 4 physical disks.
2. Creates a Storage Space named **MyMusic** which is a thinly provisioned Mirror that is 500GB in size.
3. Initializes the disk exposed by the Storage Space as GPT.
4. Creates a new partition.
5. Formats the volume
6. Reports the drive letter which was assigned to the new disk.

Figure : Single-Machine deployment of Storage Spaces using StorageSpaces.

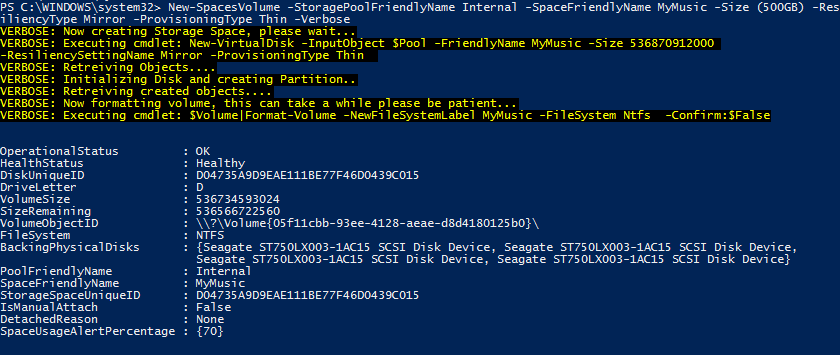
**# Create a Storage Pool using Storage Spaces**

New-SpacesPool –FriendlyName SpacesPool –NumberOfPhysicalDisksToUse 4

**# Create a Storage Space and provision the disk and volume associated with   
 the Storage Space**

New-SpacesVolume –StoragePoolFriendlyName Internal –SpaceFriendlyName   
 MyMusic –Size (500GB) –ResiliencyType Mirror –ProvisioningType Thin

Figure : Using New-Spaces Volume with the -Verbose switch



## Deploying a Failover Cluster with Storage Spaces

The Following script example performs these tasks;

1. Imports the FailoverClusters module for Windows PowerShell
2. Creates a new Failover Cluster
3. Creates a pool named **SpacesPool** using only shared Physical Disks (Disks seen by both cluster nodes).
4. Creates a Storage Space named **Quorum** as a 1GB Mirror for use as the Cluster quorum.
5. Initializes the disk exposed by the Storage Space as GPT.
6. Creates a new partition.
7. Formats the volume
8. Adds the new Quorum disk as a cluster disk.
9. Makes the newly added disk the cluster quorum.

Deployment of a failover cluster using only the Storage module

**# Create a new Failover Cluster, and skip the Storage validation as we have not  
# configured storage yet.**

New-Cluster –Name DeepSpaceClu –Node Node1,Node2 -NoStorage

**# Create a Storage Pool named SpacesPool for use with failover clustering**

New-StoragePool –FriendlyName SpacesPool –StorageSubsystemFriendlyName \*Space\* -PhysicalDisks (Get-PhysicalDisk –CanPool $True)

**# Create a New Storage Space which is mirrored, thinly provisioned, and 1GB in size.**

New-VirtualDisk -FriendlyName Quorum -StoragePoolFriendlyName SpacesPool -ResiliencySettingName Mirror -ProvisioningType Fixed -Size (1GB)

**# Find out which Disk object was created by the Storage Space we just created.**

Get-VirtualDisk -FriendlyName Quorum | Get-Disk

**# Assuming this was disk 6;**

Initialize-Disk -Number 6

**# Create a partition and assign a drive letter**

New-Partition -DiskNumber 6 -UseMaximumSize -AssignDriveLetter

**# Format the volume**

Format-Volume -NewFileSystemLabel Quorum

# Add the new Space to cluster and make it the quorum

Get-Disk 6 | Get-ClusterAvailableDisk | Add-ClusterDisk

$ClusterResource =Get-VirtualDisk –FriendlyName Quorum | Get-ClusterResource

Set-ClusterQuorum –NodeAndDiskMajority $ClusterResource.Name

Figure 10: Deployment of Storage Spaces with Failover Clustering using StorageSpaces

**# Create a new Failover Cluster, and skip the Storage validation as we have not   
 configured storage yet.**

New-Cluster –Name DeepSpaceClu –Node Node1,Node2 -NoStorage

**# Create a new pool with Storage Spaces using only Physical Disks which are shared   
 (visible) to both cluster nodes**

New-SpacesPool –FriendlyName SpacesPool –PhysicalDisks (Get-SpacesPhysicalDisk –  
 Servertocompare Node1,Node2)

**# Create a Mirrored Storage Space to use as the cluster Quorum.**

New-SpacesVolume –StoragePoolFriendlyName SpacesPool –SpaceFriendlyName Quorum –  
 Size (1GB) –ResiliencyType Mirror –ProvisioningType Fixed

**# Add the new Storage Space as a Cluster Disk**

$AddedDisk = (Get-ClusterAvailableDisk | Add-ClusterDisk)

**# Set the newly added cluster disk as the Quorum**

Set-ClusterQuorum –NodeAndDiskMajority $AddedDisk.Name

## Creating a detailed configuration report for Storage Spaces

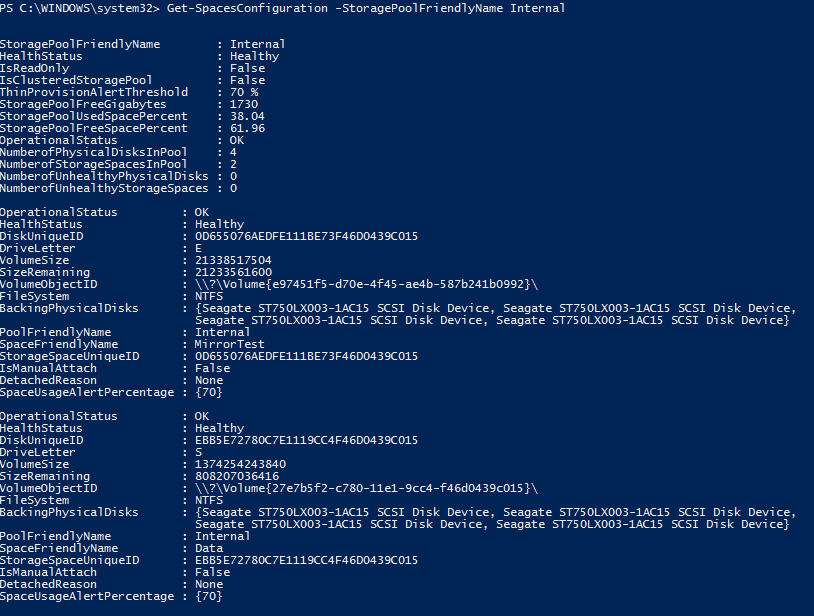
A configuration report for Storage Spaces can have many uses;

* **For Disaster Recovery** – Since the report contains details about all created Storage Spaces, as well as the pool configuration and number of Physical Disks, this report can be used as a guide for recreating a pool and all Storage Spaces for DR.
* **For Configuration Monitoring ­–** The report can also be used to monitor the current configuration of a pool and all Storage Spaces to monitor configuration changes that occur, such as the creation of additional Storage Spaces.
* **Baseline tracking ­–** Capture the initial configuration for a deployment to use as a guide for other Storage Spaces deployments.

Example output from the Get-SpacesConfiguration cmdlet, which is part of the StorageSpaces module;

Get-SpacesConfiguration –StoragePoolFriendlyName Internal

Figure : Example of Get-SpacesConfiguration cmdlet output

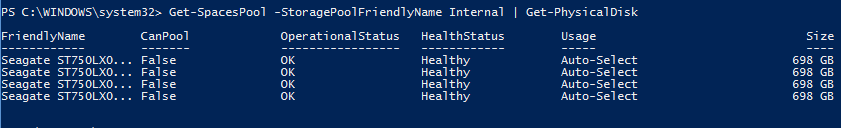


**Listing all of the physical disks in a pool**

By piping the output of Get-StoragePool to Get-PhysicalDisk, it is possible to list all physical disks associated with the given Storage Pool.

Get-StoragePool –FriendlyName Internal | Get-PhysicalDisk

Figure : Listing the Physical Disks for a specific Storage Pool



## Using Get-SpacesVolume

The Get-SpacesVolume cmdlet provides a number of additional properties about a Storage Space which are not present in the **Storage** module without running additional steps, such as;

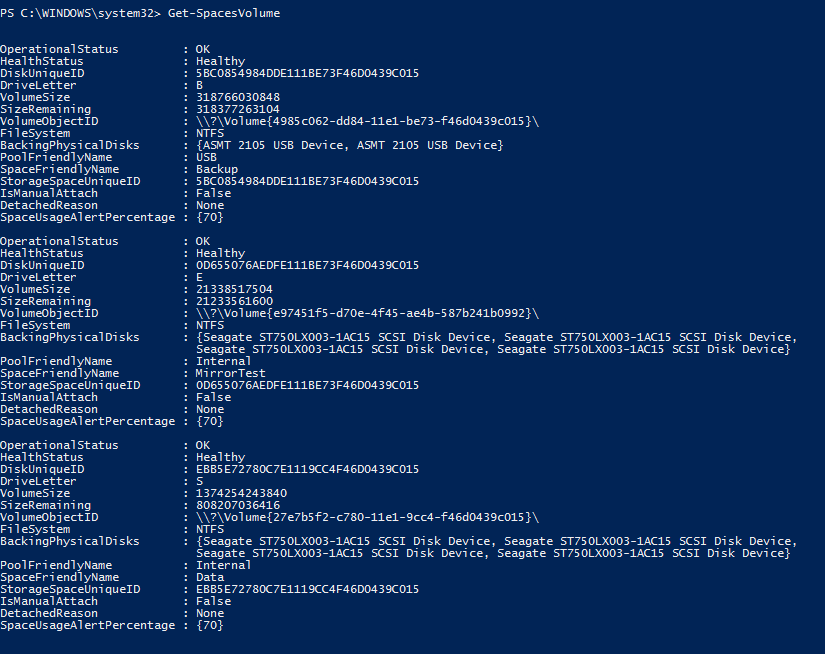
* The OperationalStatus and HealthStatus for the Storage Space and associated Disk object.
* The DriveLetter assigned to the volume located on the Storage Space.
* The FriendlyName of all Physical Disks associated with this Storage Space
* The FriendlyName of the pool that was used to create the Storage Space

If the Storage Space is clustered, the following additional details are also displayed:

* Name of the Failover Clustering resource that corresponds with this Storage Space.
* Current cluster node owner name
* Current cluster resource state

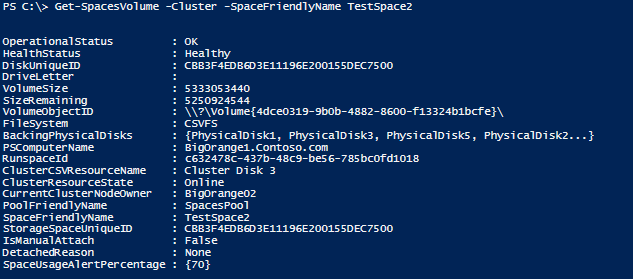
Get-SpacesVolume

Figure : Example output from Get-SpacesVolume



Using the Get-SpacesVolume cmdlet, you are able to obtain information for the Storage Space, as well as the related objects, such as the DriveLetter, and disk.

Figure : Get-SpacesVolume on a Clustered Storage Space



## Checking overall pool health

Running the Test-SpacesConifguration cmdlet will report whether there are any unhealthy resources associated with the given storage pool... For example, if the pool, any Storage Spaces, or any Physical Disks are not in a healthy state, they will be reported here.

Figure : Test-SpacesConfiguration showing a healthy pool

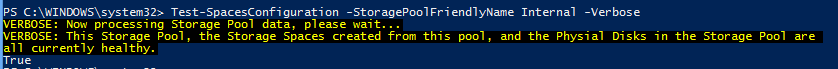


Figure : Test-SpacesConfiguration showing an unhealthy pool.



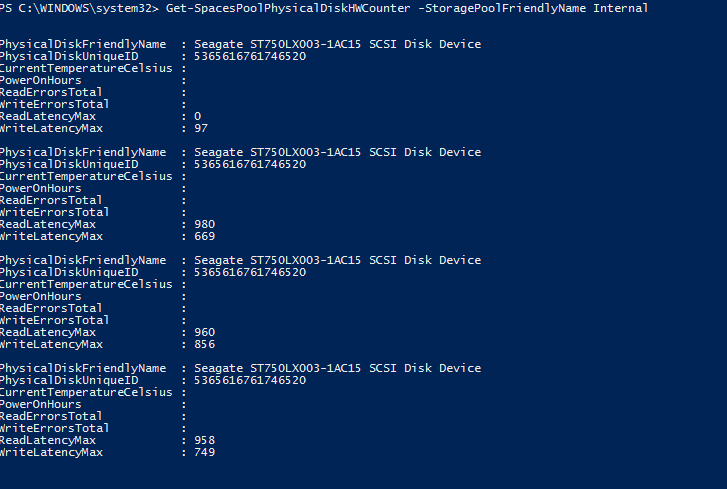
## Review reliability data for physical disks in a Storage Spaces pool.

Using the **Get-SpacesPoolPhysicalDiskHWCounter** cmdlet, it is possible to report on the physical statistics related to physical disks in the specified pool. These statistics can be useful in determining the reliability of the physical disk.

For example, if a disk reports Read Errors, Write Errors, or a high Temperature, consideration should be given to replacing the disk.

Note: The information returned by this cmdlet is limited by the information that the physical disk itself reports. For example, in the output below, the devices I am querying do not support returning information about the Temperature, Write Errors, or Read Errors, so these fields are blank.

Figure : Example of Get-SpacesPoolPhysicalDiskHWCounter output



## Enabling advanced logging of changes to the Storage Spaces state;

By using the New-StorageSpacesEventLog cmdlet, a new event log is created in Event Viewer under Applications and Services Logs named “StorageSpaces Events”. This log provides details about the actual object that experienced a notification so the specific object can be queried for details on its current state.

This log is updated by a Windows PowerShell script which is run as a scheduled job, to update the event log based on changes which have occurred, such as;

* Arrival, Departure, Alert, or modification of a Storage Space.
* Arrival, Departure, Alert, or modification of a Storage Pool.
* Arrival, Departure, Alert, or modification of Physical Disks.

Note: Once created the scheduled job which monitors for new events will re-schedule itself to persist after reboot of the system. New events are logged approximately once per minute, and will include any new events since the last time events were logged.

Figure : Example Event log containing an entry showing a healthstatus change for a Storage Space

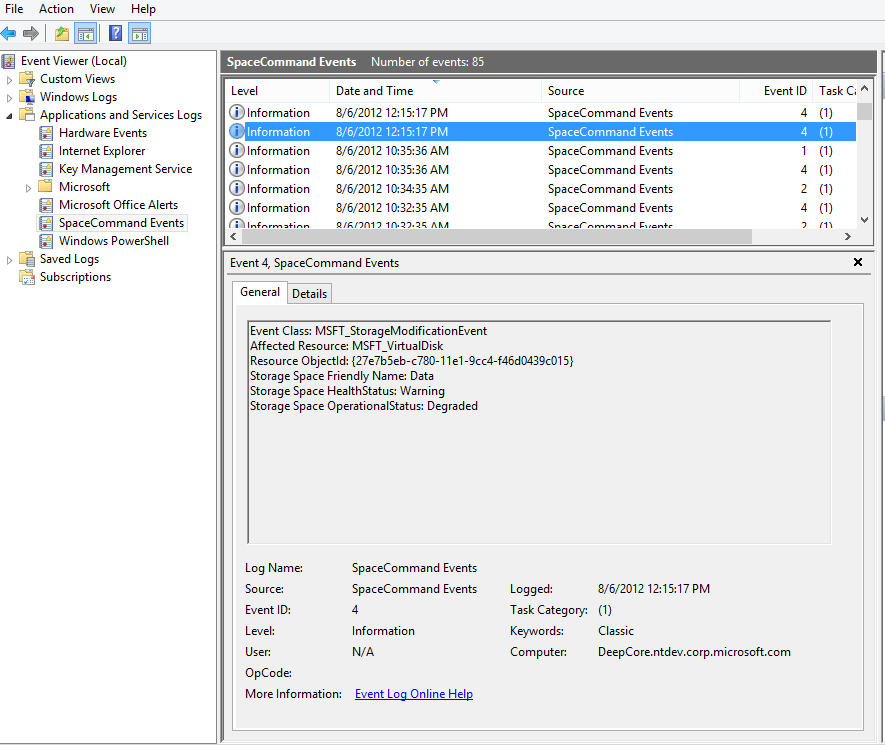
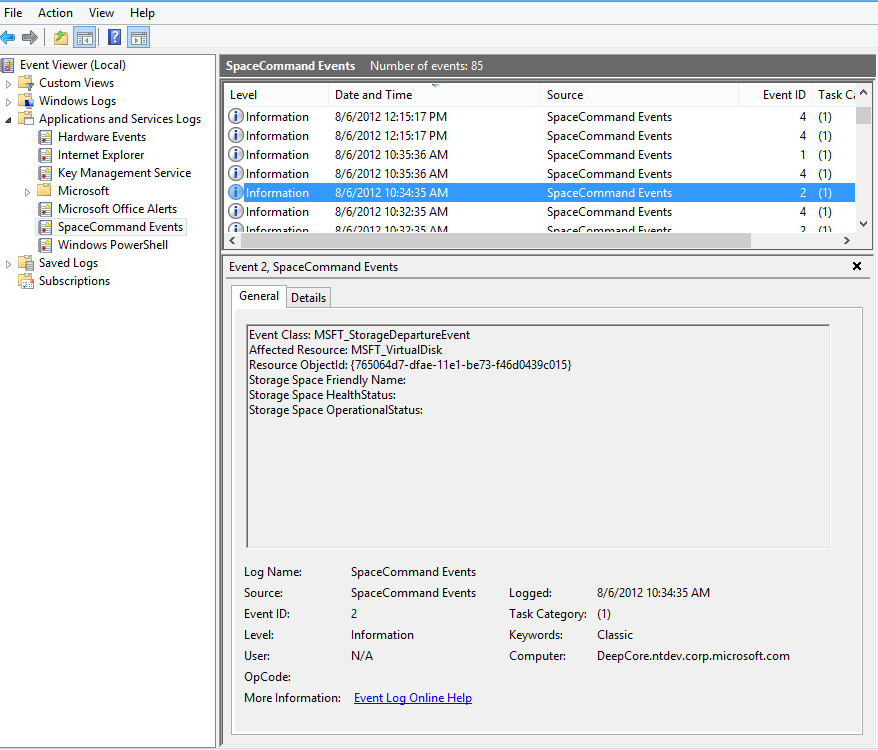


Figure : Event Log showing removal of a Physical Disk was removed from the system



# Additional Resources:

Introduction to Storage Management in Windows Server 2012  
<http://blogs.msdn.com/b/san/archive/2012/06/26/an-introduction-to-storage-management-in-windows-server-2012.aspx>

Introduction to Storage Management in Windows Server 2012 (Part 2)  
<http://blogs.msdn.com/b/san/archive/2012/07/02/introduction-to-storage-management-on-windows-server-2012-part-2.aspx>

Managing Windows storage with Windows PowerShell  
<http://blogs.msdn.com/b/san/archive/2012/07/03/managing-storage-with-windows-powershell-on-windows-server-2012.aspx>

Storage Management Windows PowerShell cmdlets  
<http://technet.microsoft.com/en-us/library/hh848705.aspx>

Managing Storage Spaces using Windows PowerShell  
<http://www.microsoft.com/en-us/download/details.aspx?id=30125>

# Appendix A: Terminology for Storage Management

Figure : Terminology glossary

|  |  |
| --- | --- |
| Term | Definition |
| Storage Management Provider (SMP) | Storage Management Providers are an interface with the Storage Management API to provide access to manage storage. There are 2 types of SMP’s supported with Windows Server 2012: SMI-S Providers Supported through the use of the Windows Standards Based Storage management Service, as a WMI provider to the storage management API. An example of a WMI-based SMP is the SMP for Storage Spaces, which is included in all Windows 8 SKUs. |
| Storage Subsystem | A StorageSubsystem is exposed by a Storage Management provider, and contains Physical Disks which are then used for the creation of a Storage Pool. |
| Physical Disk | Physical Disks are physical disk objects located in a Storage Subsystem, and are used for the creation of Storage Pools. |
| Storage Pool | A Storage Pool is a collection of Physical Disks a logical collection of Physical Disks. Storage Pools are either Concrete or Primordial. |
| Concrete Storage Pool(s) | A concrete storage pool is a storage pool containing some number of Physical Disks from a storage subsystem. A concrete pool only contains specifically allocated Physical Disks originally obtained from the Primordial Pool. |
| Primordial Storage Pool(s) | A Primordial Storage Pool is a management object which exposes all Physical Disks contained in a storage subsystem regardless of whether they have been used to form a pool. |
| Virtual Disk(s) | A virtual Disk is a virtual disk object created from a Storage Pool. Virtual disks are then exposed as a Windows Disk to a host.  Virtual Disks are known as Storage Spaces when using the Storage Spaces management provider. |
| Disk | A Disk object represents a disk as seen by a Windows host, without the use of any management providers. For example, a Disk displayed in the Disk Management console. |
| Partition | A partition created on a Disk. |
| Volume | A volume is created on a partition, and contains a Filesystem, such as NTFS. |