Step By Step: How To Create A Two-Way Mirrored Storage Space via PowerShell? #StorageSpaces #PowerShell

charbelnemnom.com/step-by-step-how-to-create-a-two-way-mirrored-storage-space-via-powershell-storagespaces-powershell

February 25, 2021

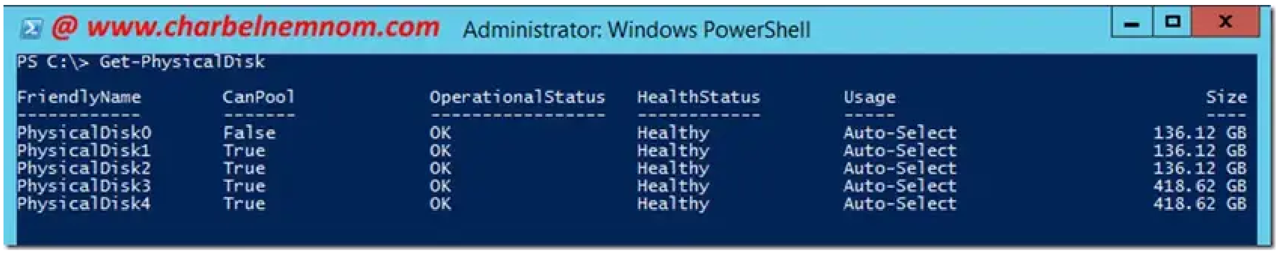
In this blog post we will continue our step by step series on Microsoft Storage Spaces, if you missed the previous post on **How To Replace A Faulty Disk In Two-Way Mirrored Storage Tiered Space**, then make sure you check it ***here***.

In today's post I will walkthrough how to create a Two-way mirror Storage Space via PowerShell.

So without further ado, let's get started.

First we will check the available physical disks in the system:

PS C:\>Get-PhysicalDisk

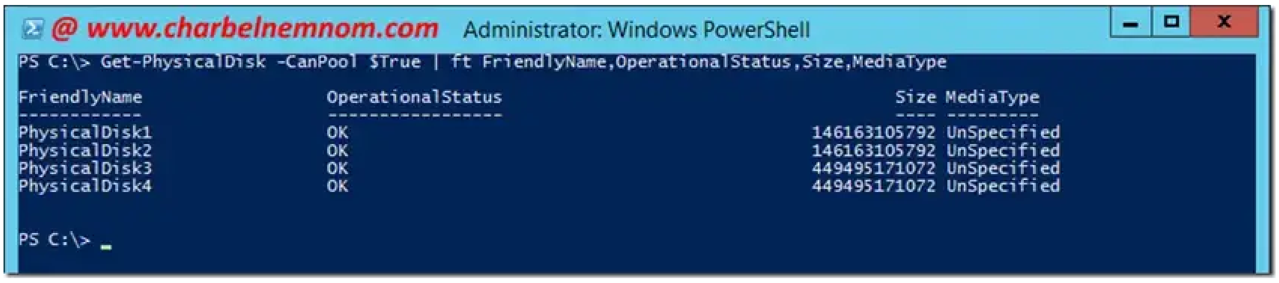


```
PS C:\> Get-PhysicalDisk

FriendlyName        CanPool            OperationalStatus   HealthStatus        Usage                          Size
------------        -------            -----------------   ------------        -----                          ----
PhysicalDisk0       False              OK                  Healthy             Auto-Select               136.12 GB
PhysicalDisk1       True               OK                  Healthy             Auto-Select               136.12 GB
PhysicalDisk2       True               OK                  Healthy             Auto-Select               136.12 GB
PhysicalDisk3       True               OK                  Healthy             Auto-Select               418.62 GB
PhysicalDisk4       True               OK                  Healthy             Auto-Select               418.62 GB
```

Next we will list the physical disks that can be pooled into our Storage Pool filtered by *Friendly Name*, *Operation Status*, *Size* and *Media Type*.

PS C:\>Get-PhysicalDisk –CanPool $True | ft FriendlyName,OperationalStatus,Size,MediaType



```
PS C:\> Get-PhysicalDisk -CanPool $True | ft FriendlyName,OperationalStatus,Size,MediaType

FriendlyName                  OperationalStatus                              Size MediaType
------------                  -----------------                              ---- ---------
PhysicalDisk1                 OK                                     146163105792 UnSpecified
PhysicalDisk2                 OK                                     146163105792 UnSpecified
PhysicalDisk3                 OK                                     449495171072 UnSpecified
PhysicalDisk4                 OK                                     449495171072 UnSpecified

PS C:\> _
```

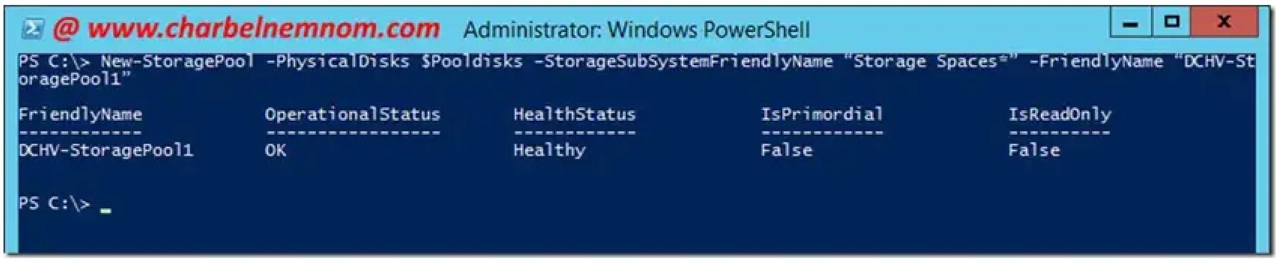
As you can see we have 4 disks that can be pooled (2X136GB and 2X418GB).

We will store all physical disks that can be pooled into a variable, **$Pooldisks**

PS C:\>$Pooldisks = Get-Physicaldisk | ? {$_.canpool -eq $true}

Next we will create a new Storage Pool using the disks in variable **$Pooldisks** with a name of “**DCHV-StoragePool1**”

PS C:\>New-StoragePool -PhysicalDisks $Pooldisks –StorageSubSystemFriendlyName “Storage Spaces*” -FriendlyName “DCHV-StoragePool1”

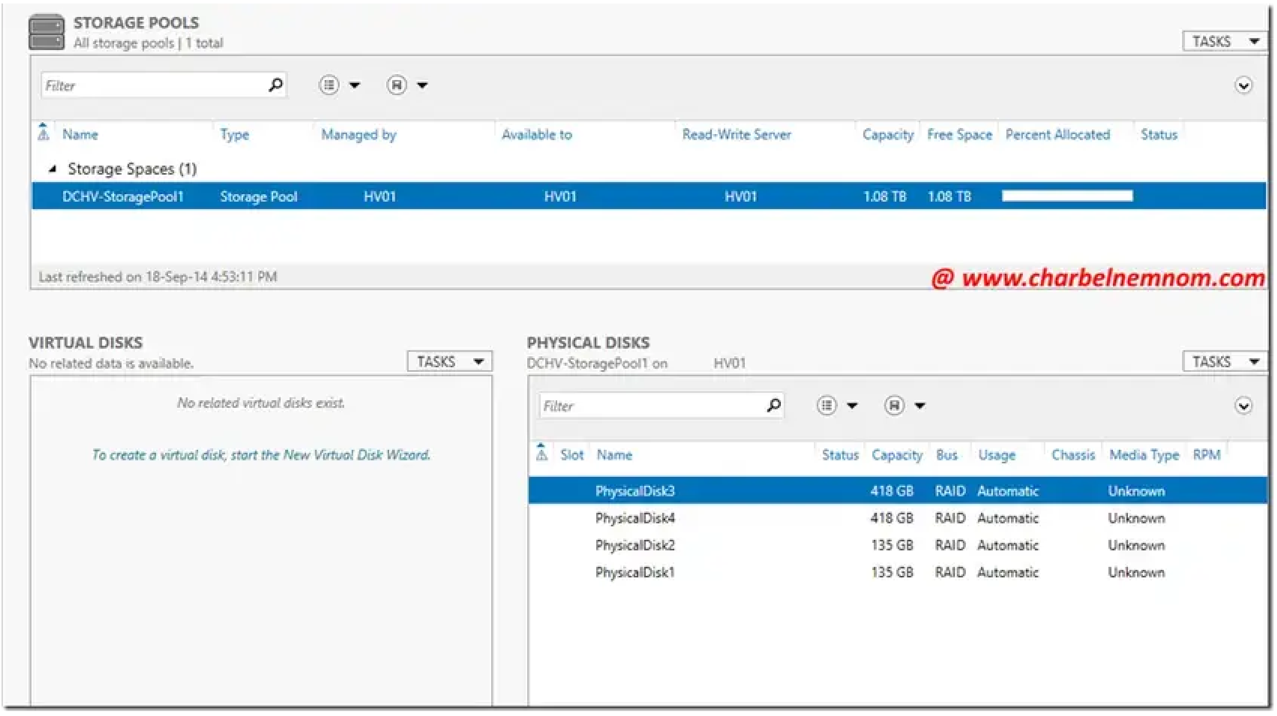


```
PS C:\> New-StoragePool -PhysicalDisks $Pooldisks -StorageSubSystemFriendlyName "Storage Spaces*" -FriendlyName "DCHV-StoragePool1"

FriendlyName        OperationalStatus   HealthStatus        IsPrimordial        IsReadOnly
------------        -----------------   ------------        ------------        ----------
DCHV-StoragePool1   OK                  Healthy             False               False

PS C:\> _
```

Here is the result in the UI:

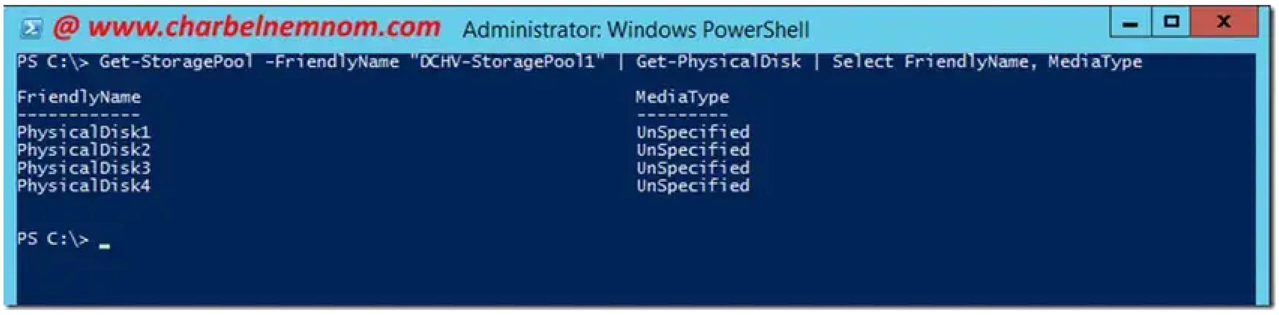

STORAGE POOLS
All storage pools | 1 total
TASKS
Filter
Last refreshed on 18-Sep-14 4:53:11 PM
@ www.charbelnemnom.com
VIRTUAL DISKS
No related data is available.
TASKS
No related virtual disks exist.
To create a virtual disk, start the New Virtual Disk Wizard.
PHYSICAL DISKS
DCHV-StoragePool1 on HV01
TASKS
Filter

Name	Type	Managed by	Available to	Read-Write Server	Capacity	Free Space	Percent Allocated	Status
Storage Spaces (1)								
DCHV-StoragePool1	Storage Pool	HV01	HV01	HV01	1.08 TB	1.08 TB		

Slot	Name	Status	Capacity	Bus	Usage	Chassis	Media Type	RPM
	PhysicalDisk3		418 GB	RAID	Automatic		Unknown	
	PhysicalDisk4		418 GB	RAID	Automatic		Unknown	
	PhysicalDisk2		135 GB	RAID	Automatic		Unknown	
	PhysicalDisk1		135 GB	RAID	Automatic		Unknown	

Now let’s view the disks in the Storage Pool that we just created.

PS C:\>Get-StoragePool -FriendlyName “DCHV-StoragePool1” | Get-PhysicalDisk | Select FriendlyName, MediaType



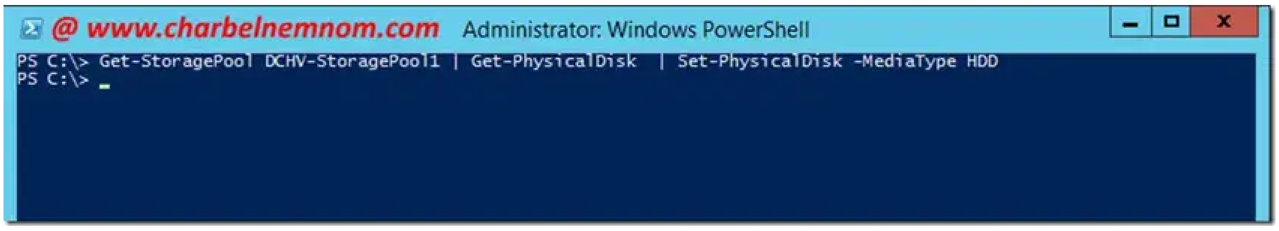
```
PS C:\> Get-StoragePool -FriendlyName "DCHV-StoragePool1" | Get-PhysicalDisk | Select FriendlyName, MediaType
FriendlyName                                                MediaType
------------                                                ---------
PhysicalDisk1                                               UnSpecified
PhysicalDisk2                                               UnSpecified
PhysicalDisk3                                               UnSpecified
PhysicalDisk4                                               UnSpecified

PS C:\> _
```

As you can see the **MediaType** is shown as **UnSpecified**, so before we continue we must set the MediaType properly.

And since I am using only Normal Hard Drives in this demonstration, we will set the MediaType to HDD, and if you are creating a tiered Storage Space, then make sure to set each type properly (HDD / SSD).

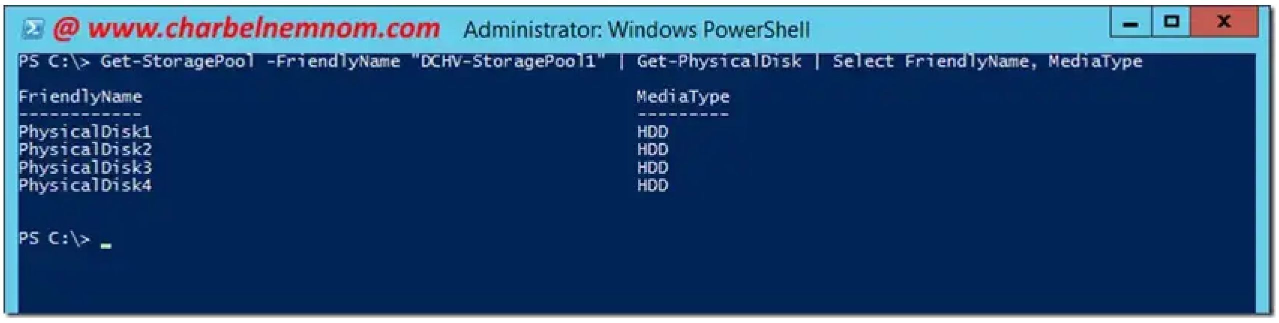
PS C:\>Get-StoragePool DCHV-StoragePool1 | Get-PhysicalDisk | Set-PhysicalDisk -MediaType HDD



```
PS C:\> Get-StoragePool DCHV-StoragePool1 | Get-PhysicalDisk  | Set-PhysicalDisk -MediaType HDD
PS C:\> _
```

Let’s view the disks in the Storage Pool after we specified the media type to HDD:

PS C:\>Get-StoragePool -FriendlyName “DCHV-StoragePool1” | Get-PhysicalDisk | Select FriendlyName, MediaType



```
PS C:\> Get-StoragePool -FriendlyName "DCHV-StoragePool1" | Get-PhysicalDisk | Select FriendlyName, MediaType

FriendlyName                                            MediaType
------------                                            ---------
PhysicalDisk1                                           HDD
PhysicalDisk2                                           HDD
PhysicalDisk3                                           HDD
PhysicalDisk4                                           HDD


PS C:\> _
```

Last but not least, we will create a **New-VirtualDisk** with **ResiliencySettingName** Mirror, then the **NumberOfDataCopies** is equal to (2=Two-way mirror Space, 3=Three-way mirror Space). In this demo we can only create Two-way mirror space since we have only 4 disks, however for Three-way mirror we need 5 disks. A two-way mirror will allow you to suffer the loss of a single disk with no problems while a three-way mirror will allow you to lose two disks.

Next we move to choose the **ProvisioningType (Fixed or Thin),** we will choose Fixed instead of Thin provisioning, and then we specify Maximum disk size. It’s very important to mention that thin provisioning will not prevent storage shortages. Applications will break if storage is not added to the thinly provisioned volumes in time.

The final and confused parameters are the **NumberOfColumns** and the **Interleave**.

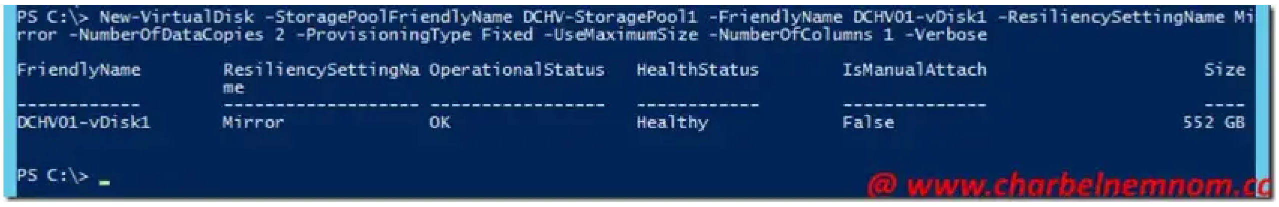
The more columns means more performance because multiple disks will be engaged at once in Read/Write operations, but it’s also limited in flexibility with expanding existing virtual disks, especially in tiered scenarios. So what’s the best size for columns for Two-way mirror Storage Space?

Typically the column count will be equal to the number of physical disks of the storage space (for simple spaces) or half of the number of disks (for mirror spaces). The column count can be lower than the number of physical disks but never higher.

The more is better in terms of performance, but the less is better in terms of flexibility for future expansion. There is no simple answer here, so it depends!

Here is a Two-way mirror virtual disk with Number of Columns=1.

***PS C:*\>New-VirtualDisk -StoragePoolFriendlyName DCHV-StoragePool1 -FriendlyName DCHV01-vDisk1 -ResiliencySettingName Mirror -NumberOfDataCopies 2 -ProvisioningType Fixed -UseMaximumSize -NumberOfColumns 1 -Verbose**



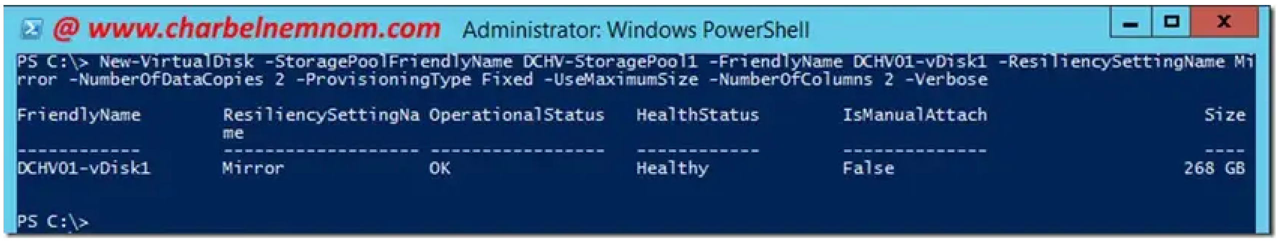
```
PS C:\> New-VirtualDisk -StoragePoolFriendlyName DCHV-StoragePool1 -FriendlyName DCHV01-vDisk1 -ResiliencySettingName Mi
rror -NumberOfDataCopies 2 -ProvisioningType Fixed -UseMaximumSize -NumberOfColumns 1 -Verbose

FriendlyName        ResiliencySettingNa OperationalStatus   HealthStatus        IsManualAttach                      Size
                    me
------------        ------------------- -----------------   ------------        --------------                      ----
DCHV01-vDisk1       Mirror              OK                  Healthy             False                             552 GB

PS C:\> _
```

Here is a Two-way mirror virtual disk with Number of Columns=2.

***PS C:*\>New-VirtualDisk -StoragePoolFriendlyName DCHV-StoragePool1 -FriendlyName DCHV01-vDisk1 -ResiliencySettingName Mirror -NumberOfDataCopies 2 -ProvisioningType Fixed -UseMaximumSize -NumberOfColumns 2 -Verbose**



```
PS C:\> New-VirtualDisk -StoragePoolFriendlyName DCHV-StoragePool1 -FriendlyName DCHV01-vDisk1 -ResiliencySettingName Mi
rror -NumberOfDataCopies 2 -ProvisioningType Fixed -UseMaximumSize -NumberOfColumns 2 -Verbose

FriendlyName        ResiliencySettingNa OperationalStatus   HealthStatus        IsManualAttach                      Size
                    me
------------        ------------------- -----------------   ------------        --------------                      ----
DCHV01-vDisk1       Mirror              OK                  Healthy             False                             268 GB

PS C:\>
```

As you can see with Column#1, we have more capacity space than Column#2. If you use 1 column, your space will only be as fast as one individual disk.

The **Interleave** parameter represents the amount of data written to a single column per stripe. The default Interleave value is 262,144 bytes (256 KB).

The final step is to initialize the volume and create the new partition.

PS C:\>Get-VirtualDisk DCHV01-vDisk1 | Get-Disk | Set-Disk -IsReadOnly 0
PS C:\>Get-VirtualDisk DCHV01-vDisk1 | Get-Disk | Set-Disk -IsOffline 0
PS C:\>Get-VirtualDisk DCHV01-vDisk1 | Get-Disk | Initialize-Disk -PartitionStyle GPT
PS C:\>Get-VirtualDisk DCHV01-vDisk1 | Get-Disk | New-Partition -DriveLetter "D" -UseMaximumSize
PS C:\>Initialize-Volume -DriveLetter "D" -FileSystem NTFS -Confirm:$false -NewFileSystemLabel "Hyper-V"

Microsoft has a great **Storage Space performance paper** which goes in more detail and is worth a read.

Until then… N’joy your day!
/Charbel