

ITAS141 Lab5

By Ethan Holmes

Contents

Objective:	3
Part 2: GUI Storage Faces	3
Part 3: PowerShell Storage Spaces	10
Part 4: Storage Cleanup	12
Final Thoughts:	13

Objective:

In this lab, the objective was to learn about storage settings and configuring various storage settings through a GUI as well as a command line, learning how to initialize these disks and create Volumes.

Part 2: GUI Storage Faces

To start, after adding my 3, 30GB disks to my SA1 Machine in vSphere, I went to my Storage Pool in Server Manager to create a storage pool using 2 of the 3 disks that I created.

Name	Type	Managed by	Available to	Read-Write Server	Capacity	Free Space	Percent Allocated	Status
Windows Storage (2)								
Primordial	Available Disks	ethanholmesSA1	ethanholmesSA1	ethanholmesSA1				
Pool1	Storage Pool	ethanholmesSA1	ethanholmesSA1	ethanholmesSA1	58.5 GB	52.0 GB	<div></div>	

Figure 1: Created Storage Pool

After, I created a Volume on the Pool that I just created that was 5gb in size with a Simple layout.

VIRTUAL DISKS										
Pool1 on ethanholmesSA1										TASKS ▼
Filter 🔍 ⌵ ⌵ ⌵										
Name	Status	Layout	Provisioning	Capacity	Allocated	Volume	Clustered	Tiered	Write-Back Cache	
Virt1		Simple	Fixed	6.00 GB	6.00 GB					

Figure 2: Created Virtual Disk

I then created a Virtual Disk named “Virt2” that had a Mirrored layout, giving it the rest of the space available, 25GBs. Because it’s a mirrored disk, I lose out on some space on this disk as its reserved for Parity.

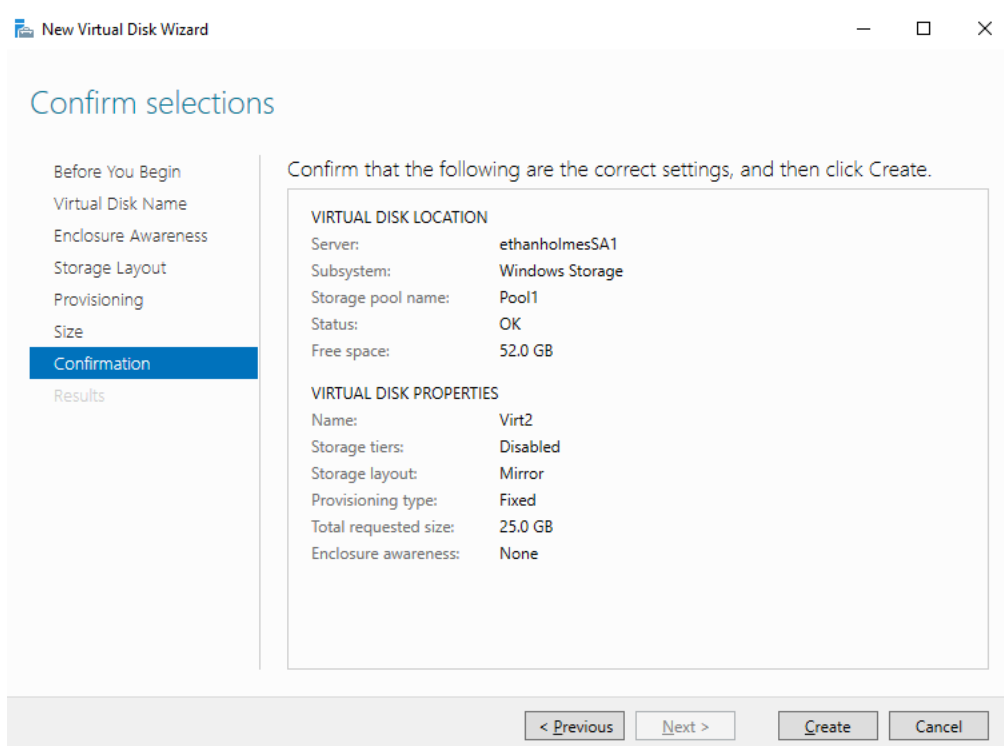


Figure 3: Confirming and creating the storage pool

Again, I followed the Microsoft wizard, creating another volume on the Storage Pool as seen in the figure below

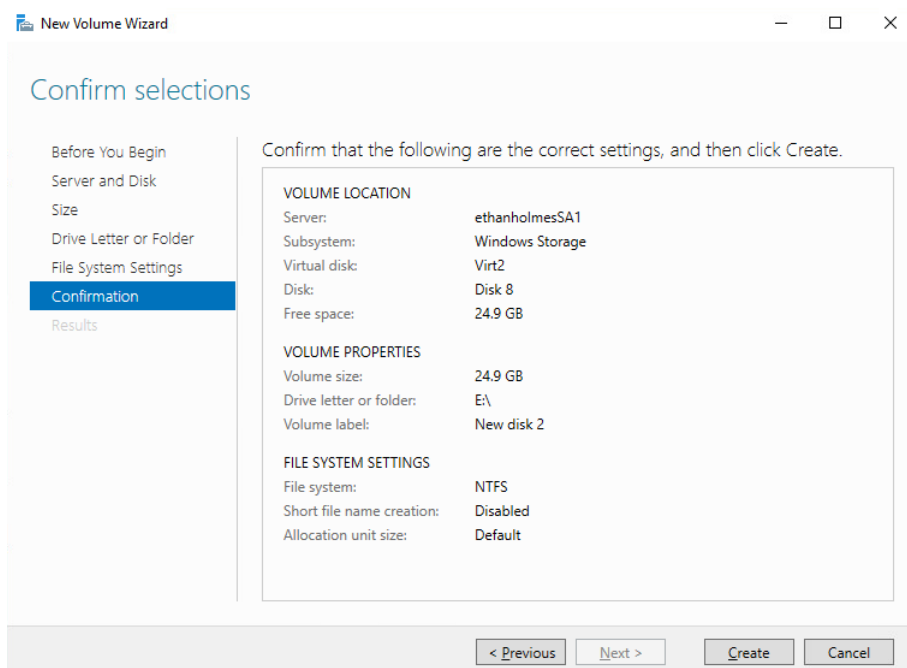


Figure 4: Creating Volume on the 2nd virtual disk

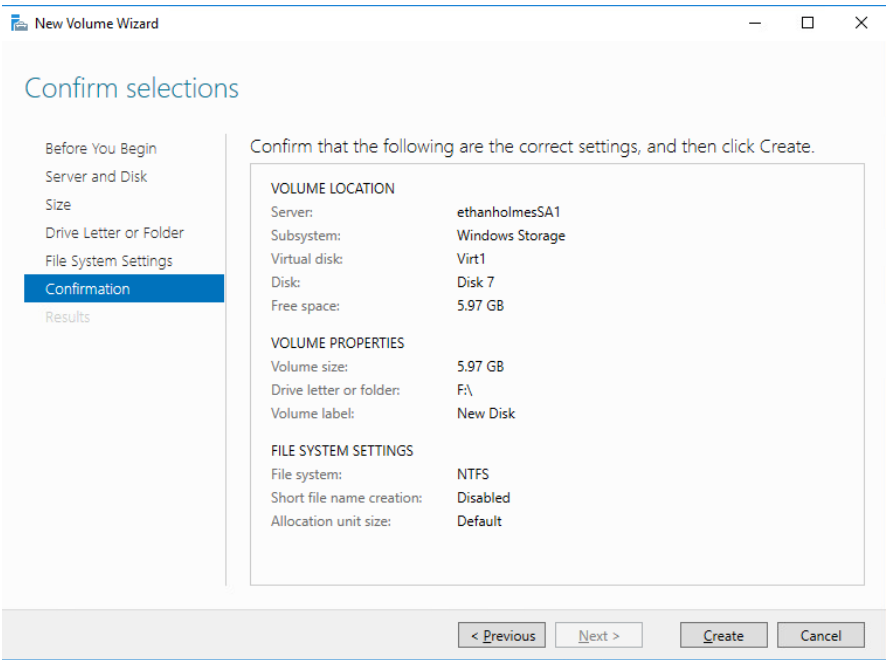


Figure 5: first disks Volumes

Here we have the 2 disks completed in the storage pool, with the capacity used, and the view from disk management.

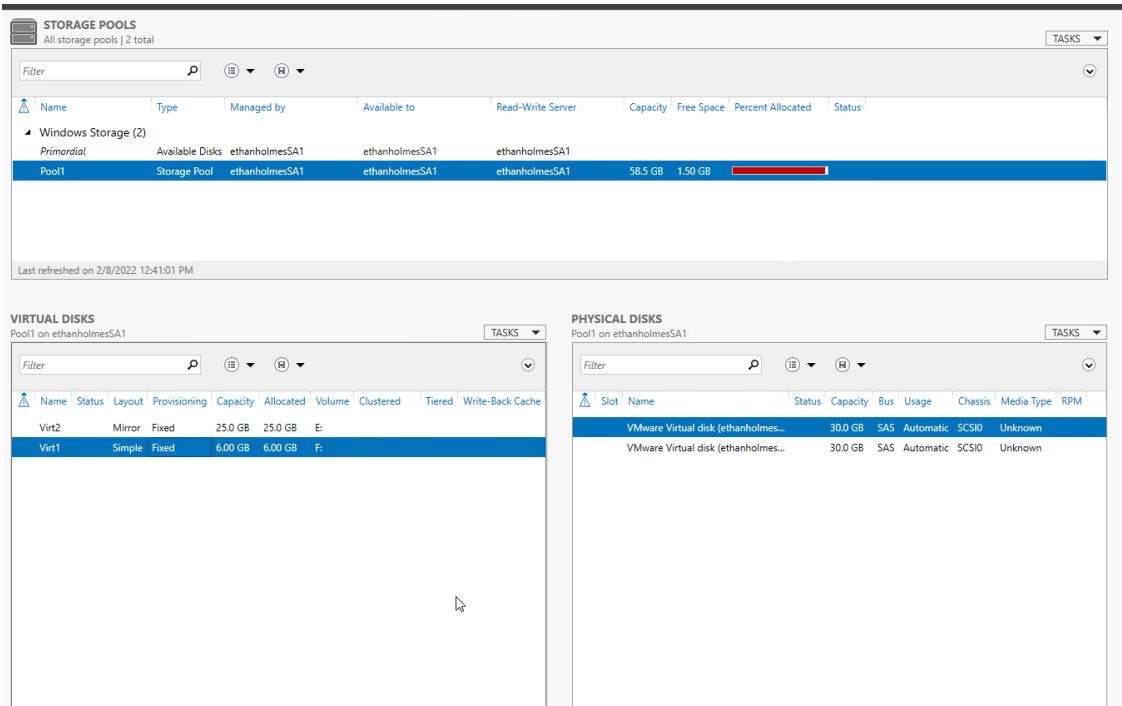


Figure 6: Storage pool with 2 Virtual Disks that have volumes on them

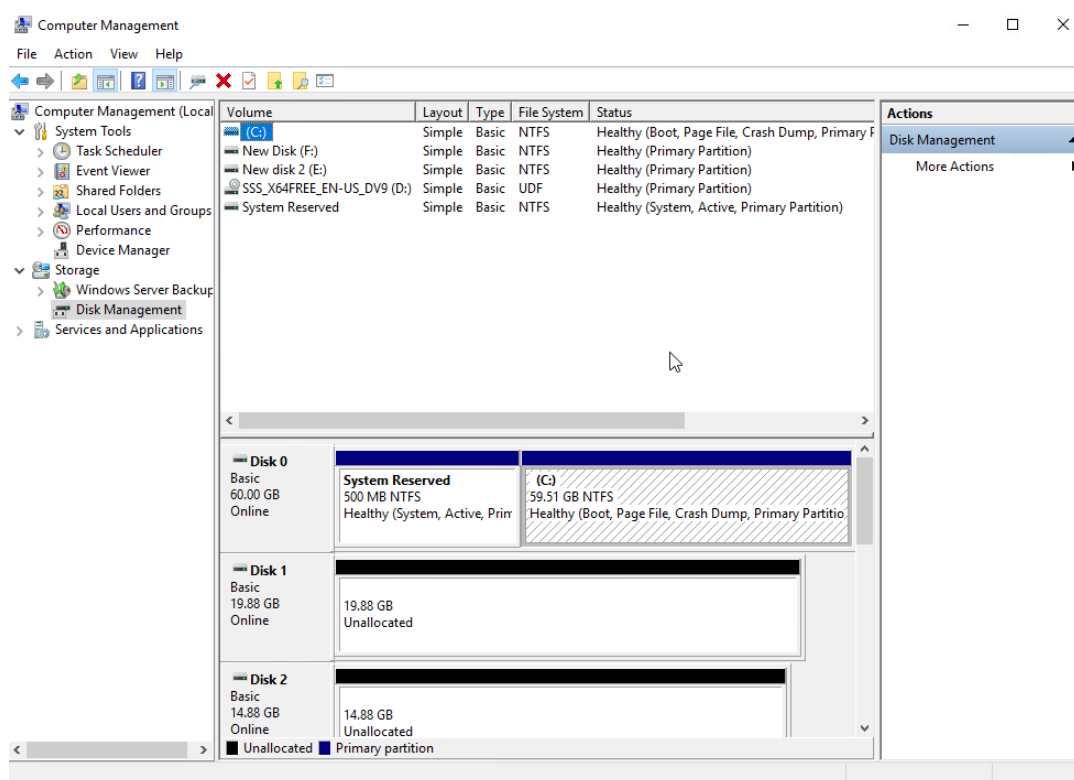


Figure 7: Showing them in Disk Management through MMC

After this, we are asked to attempt to extend our Virt1 disk

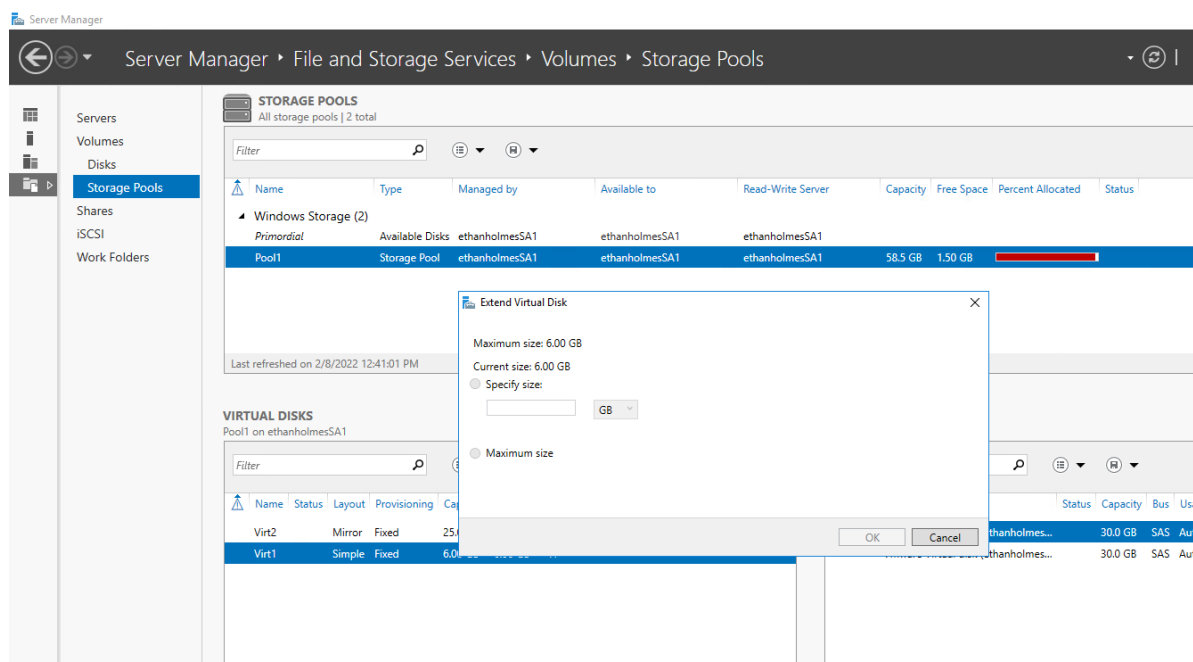


Figure 8: Unable to extend our Disk due to Fixed provisioning and not enough space

This is not possible however, as we don't have enough free space to be able to extend the disk (also because these were originally partitioned as fixed, I fixed this after). As I mentioned above, because of the mirrored disk that we made Virt2, it needs more space for the disk to replicate data. We can always expand our storage pool to allocate more space. For this, after right-clicking the Storage pool, I click "Add Physical Disk" and add the 3rd 30gb disk that we made earlier.

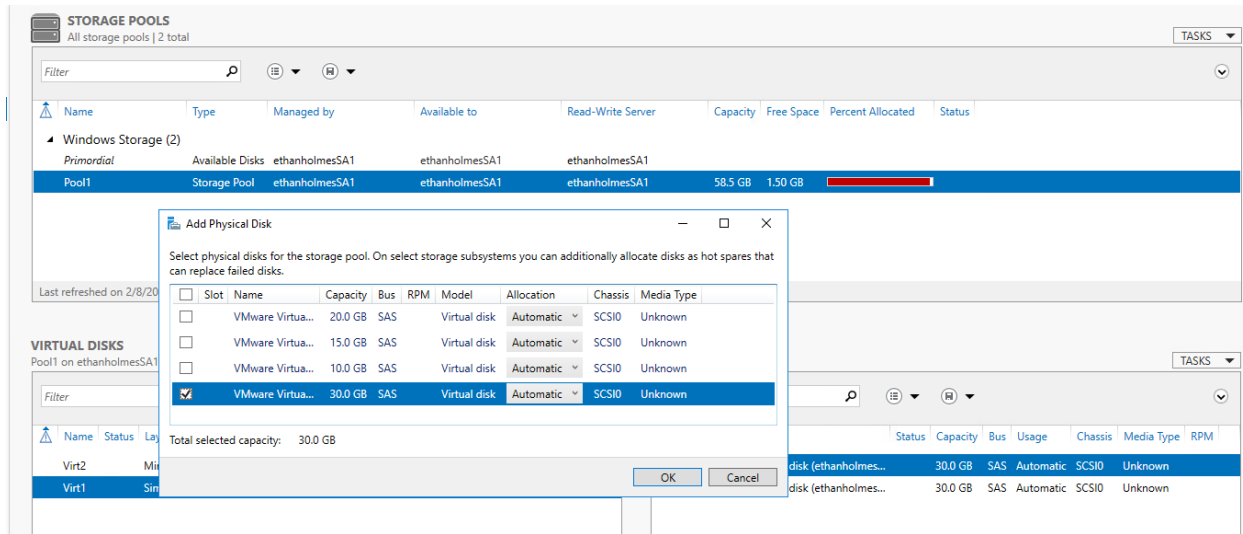


Figure 9: Extending the storage pool

After adding the 3rd 30gb size disk, we can expand our storage up to the 8gb that is asked in the lab

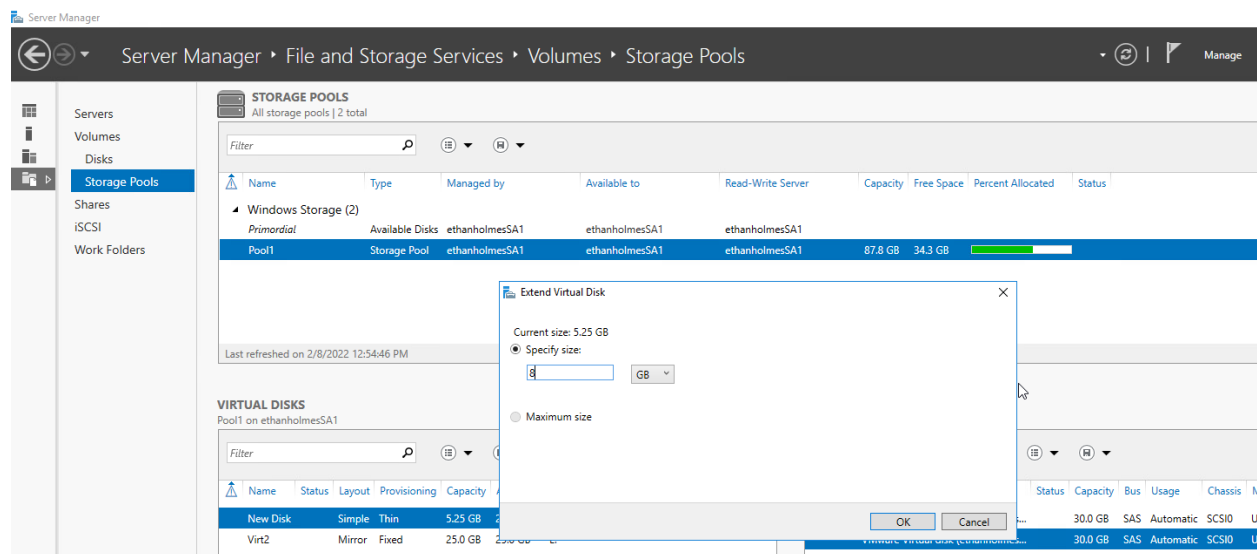
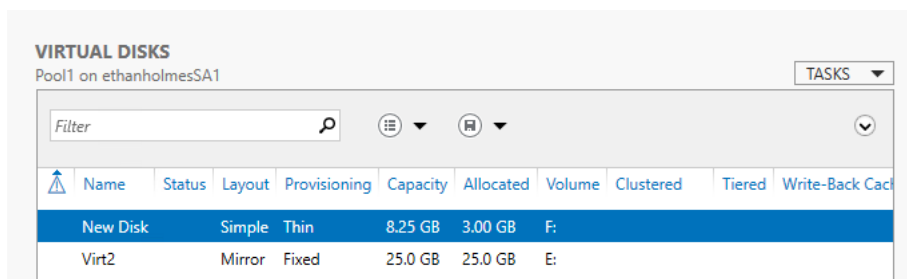


Figure 10: Extending the disk size now that we have enough space



Name	Status	Layout	Provisioning	Capacity	Allocated	Volume	Clustered	Tiered	Write-Back Cache
New Disk	Simple	Thin		8.25 GB	3.00 GB	F:			
Virt2	Mirror	Fixed		25.0 GB	25.0 GB	E:			

Figure 11: Confirming new disk size

However just because the disk is expanded, does not mean we have access to the additional 3gb that we have given it, for this we need to extend the volume in disk management. Right click the Volume and click “Extend Volume” to start the process

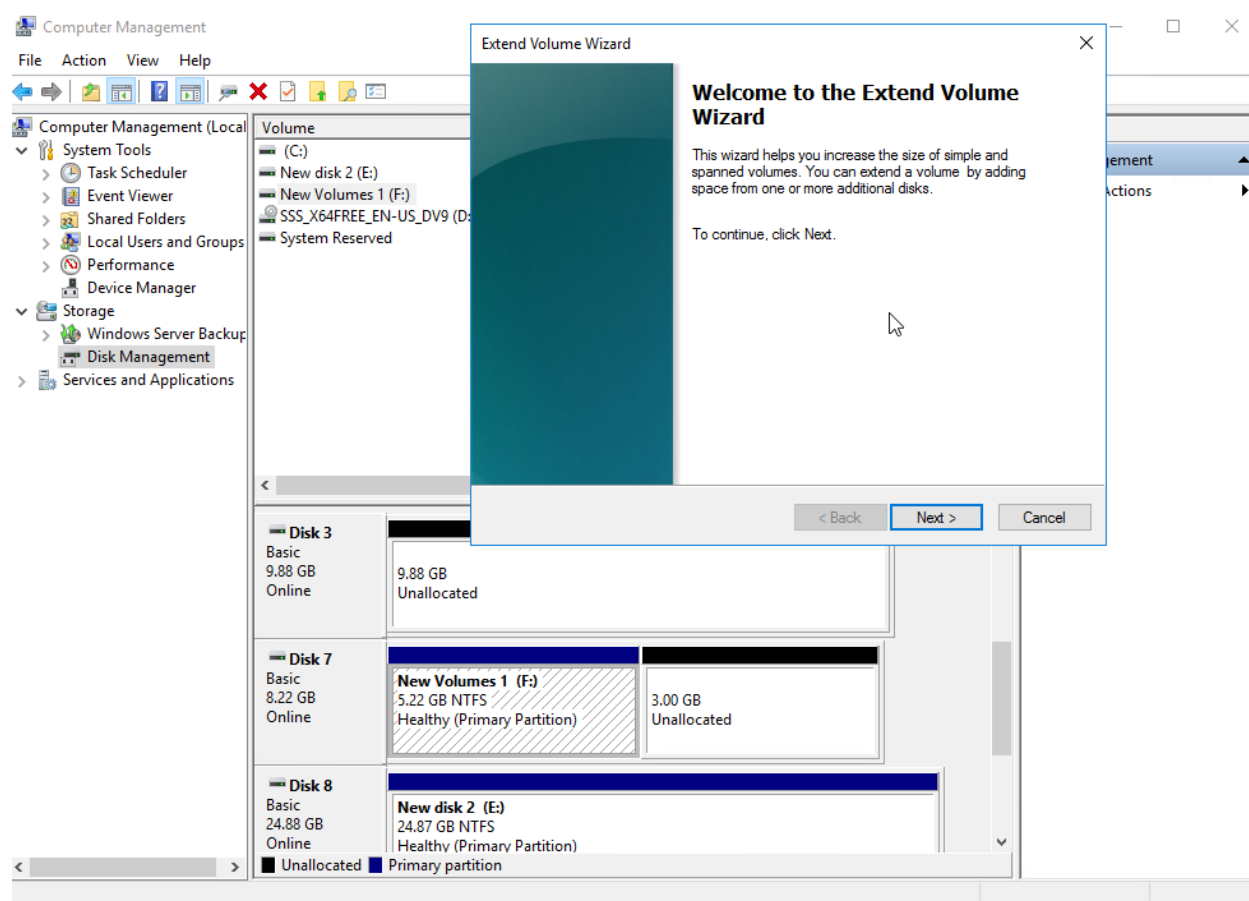


Figure 12: Extending the Volume size

We can click next, as the additional space is already selected in our disk

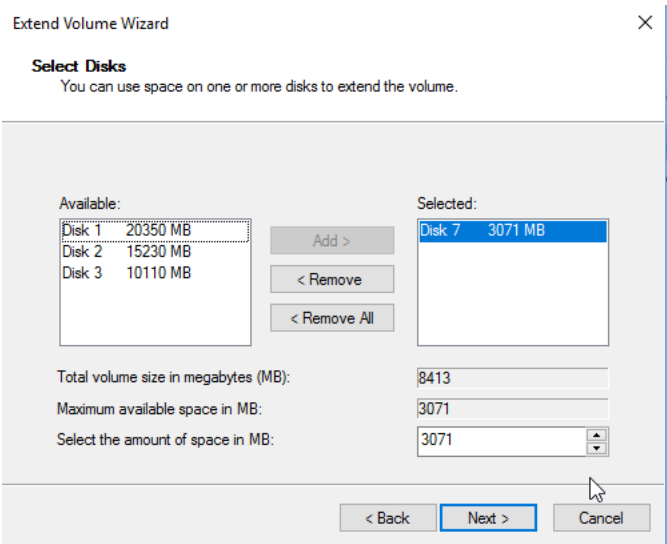


Figure 13: Adding the additional 3gb from Disk 7 to the volume

Just follow the on-screen instructions to finish the process, and we now have an extended volume from 5gb to 8gb

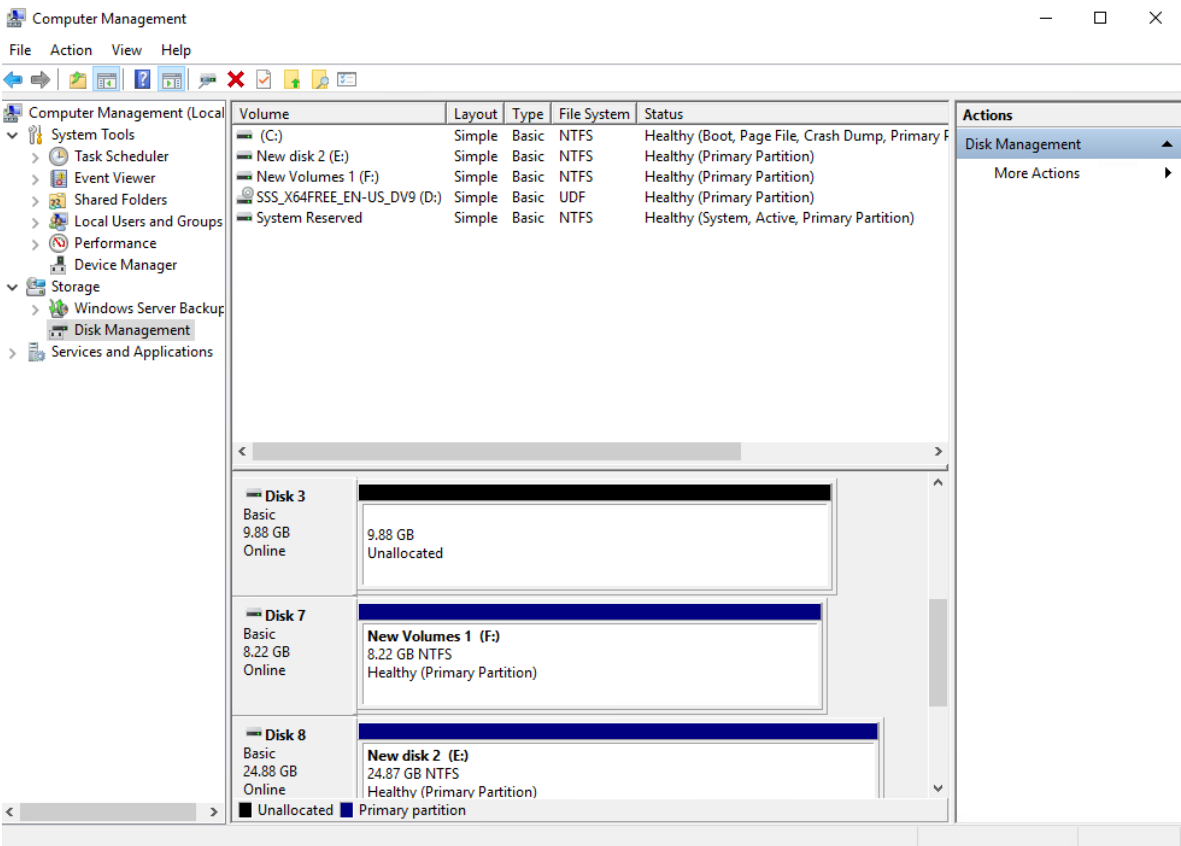
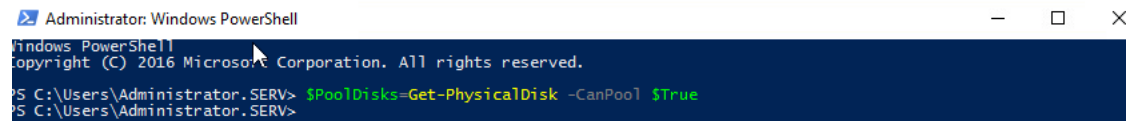


Figure 14: Confirming the new volume size

Part 3: PowerShell Storage Spaces

For this part, we need to create a storage pool and create volumes using PowerShell rather than a GUI, to start this, In PowerShell I start with the following command

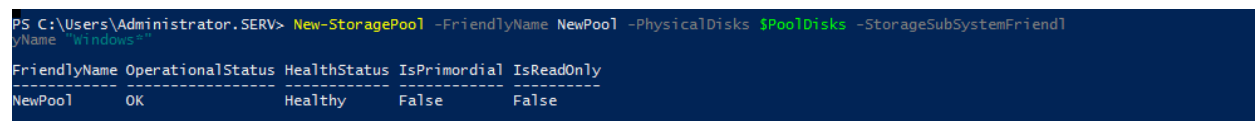


```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator.SERV> $PoolDisks=Get-PhysicalDisk -CanPool $True
PS C:\Users\Administrator.SERV>
```

Figure 15: Initializing new disks for storage pool

This is to prepare disks to be added to a storage pool, next I will use the command as shown in the figure below to create the storage pool

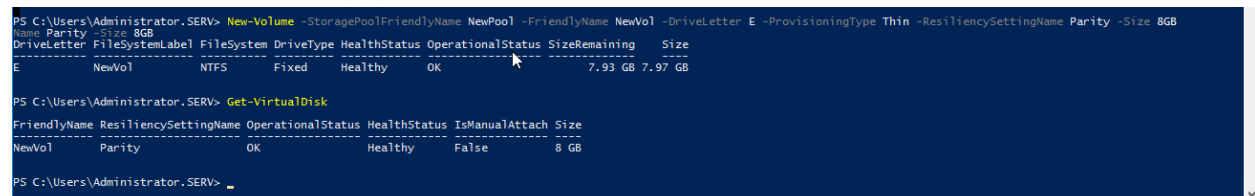


```
PS C:\Users\Administrator.SERV> New-StoragePool -FriendlyName NewPool -PhysicalDisks $PoolDisks -StorageSubSystemFriendlyName "Windows"
```

FriendlyName	OperationalStatus	HealthStatus	IsPrimordial	IsReadOnly
NewPool	OK	Healthy	False	False

Figure 16: Creating the storage pool

After creating the storage pool, we then need to create a Parity Virtual Disk that is fixed with a size of 8GB, to do this, I used the following command



```
PS C:\Users\Administrator.SERV> New-Volume -StoragePoolFriendlyName NewPool -FriendlyName NewVol1 -DriveLetter E -ProvisioningType Thin -ResiliencySettingName Parity -Size 8GB
```

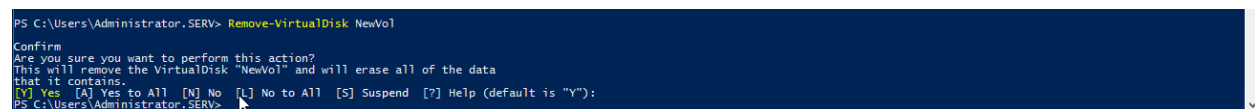
Name	Parity	Size	8GB				
DriveLetter	FileSystemLabel	FileSystem	DriveType	HealthStatus	OperationalStatus	SizeRemaining	Size
E	NewVol1	NTFS	Fixed	Healthy	OK	7.93 GB	7.97 GB


```
PS C:\Users\Administrator.SERV> Get-VirtualDisk
```

FriendlyName	ResiliencySettingName	OperationalStatus	HealthStatus	IsManualAttach	Size
NewVol1	Parity	OK	Healthy	False	8 GB

Figure 17: Creating the new volume on a storage pool and checking its settings

Using "Get-VirtualDisk" we can see that Parity has been set. After creating this, we then need to delete the disk.

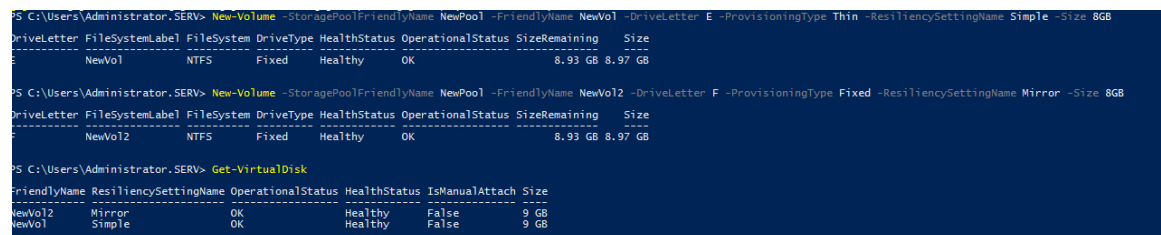


```
PS C:\Users\Administrator.SERV> Remove-VirtualDisk NewVol1
```

Confirm
Are you sure you want to perform this action?
This will remove the VirtualDisk "NewVol1" and will erase all of the data that it contains.
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"):

Figure 18: Deleting the virtual disk

The last part of this lab, we need to create 2 more virtual disks, but this time with different settings.



```
PS C:\Users\Administrator.SERV> New-Volume -StoragePoolFriendlyName NewPool -FriendlyName NewVol1 -DriveLetter E -ProvisioningType Thin -ResiliencySettingName Simple -Size 8GB
```

DriveLetter	FileSystemLabel	FileSystem	DriveType	HealthStatus	OperationalStatus	SizeRemaining	Size
E	NewVol1	NTFS	Fixed	Healthy	OK	8.93 GB	8.97 GB


```
PS C:\Users\Administrator.SERV> New-Volume -StoragePoolFriendlyName NewPool -FriendlyName NewVol2 -DriveLetter F -ProvisioningType Fixed -ResiliencySettingName Mirror -Size 8GB
```

DriveLetter	FileSystemLabel	FileSystem	DriveType	HealthStatus	OperationalStatus	SizeRemaining	Size
F	NewVol2	NTFS	Fixed	Healthy	OK	8.93 GB	8.97 GB


```
PS C:\Users\Administrator.SERV> Get-VirtualDisk
```

FriendlyName	ResiliencySettingName	OperationalStatus	HealthStatus	IsManualAttach	Size
NewVol2	Mirror	OK	Healthy	False	9 GB
NewVol1	Simple	OK	Healthy	False	9 GB

Figure 19: Creating the 2 new volumes on the disk

In the Figure above, we see the 2 Volumes that I created, and we see that in “NewVol” Matches the “Simple” resiliency setting. To show that the disk is thin provisioned, we can use “Get-VirtualDisk | FL” to get more data about the disks.

```
ObjectID      : {1}\\ETHANHOLMESSAI\root\Microsoft\Windows\Storage\Providers_v2\SPACES_VirtualDisk.ObjectId="{eea79ba2-7359-11ec-9db6-806e6f6e6963}:VD:{69fde373-088f-4f36-8942-de3b9251ae9c}[dfff9b940-9212-4b73-83bd-3234ed389f8c]"
PassThroughClass :
PassThroughIds   :
PassThroughNamespace :
PassThroughServer :
UniqueId        : 4089f90f1292734883d03234ED389F8C
Access          : Read/Write
AllocatedSize    : 4831838208
AllocationUnitSize : 268435456
ColumnIsolation  : PhysicalDisk
DetachedReason    : None
FaultDomainAwareness : PhysicalDisk
FootprintOnPool   : 4831838208
FriendlyName     : NewVol
HealthStatus      : Healthy
Interleave       : 262144
IsDeduplicationEnabled : False
IsEnclosureAware : False
IsManualAttach    : False
IsSnapshot        : False
IsTiered          : False
LogicalSectorSize : 512
MediaType        : Unspecified
Name             :
NameFormat       :
NumberOfAvailableCopies :
NumberOfColumns  : 6
NumberOfDataCopies : 1
NumberOfGroups    : 1
OperationalStatus : OK
OtherOperationalStatusDescription :
OtherUsageDescription :
ParityLayout      :
PhysicalDiskRedundancy : 0
ProvisioningType   : Thin
RequestNoSinglePointOfFailure : False
ResiliencySettingName : Simple
Size              : 9663676416
UniqueIdFormat     : Vendor Specific
UniqueIdFormatDescription :
Usage             : Other
WriteCacheSize     : 0
PSComputerName    :
```

Figure 20: Confirming the settings on NewVol

```
PS C:\Users\Administrator.SERV> Get-VirtualDisk | FL

ObjectID      : {1}\\ETHANHOLMESSAI\root\Microsoft\Windows\Storage\Providers_v2\SPACES_VirtualDisk.ObjectId="{eea79ba2-7359-11ec-9db6-806e6f6e6963}:VD:{69fde373-088f-4f36-8942-de3b9251ae9c}[90c277de-c76e-4b2d-a968-2d941e4d6d41]"
PassThroughClass :
PassThroughIds   :
PassThroughNamespace :
PassThroughServer :
UniqueId        : DE77C2906EC72D48A9682D941E4D6D41
Access          : Read/Write
AllocatedSize    : 9663676416
AllocationUnitSize : 1073741824
ColumnIsolation  : PhysicalDisk
DetachedReason    : None
FaultDomainAwareness : PhysicalDisk
FootprintOnPool   : 19864223744
FriendlyName     : NewVol2
HealthStatus      : Healthy
Interleave       : 262144
IsDeduplicationEnabled : False
IsEnclosureAware : False
IsManualAttach    : False
IsSnapshot        : False
IsTiered          : False
LogicalSectorSize : 512
MediaType        : Unspecified
Name             :
NameFormat       :
NumberOfAvailableCopies :
NumberOfColumns  : 3
NumberOfDataCopies : 2
NumberOfGroups    : 1
OperationalStatus : OK
OtherOperationalStatusDescription :
OtherUsageDescription :
ParityLayout      :
PhysicalDiskRedundancy : 1
ProvisioningType   : Fixed
RequestNoSinglePointOfFailure : False
ResiliencySettingName : Mirror
Size              : 9663676416
UniqueIdFormat     : Vendor Specific
UniqueIdFormatDescription :
Usage             : Other
WriteCacheSize     : 0
PSComputerName    :
```

Figure 21: Confirming the settings on NewVol2

Part 4: Storage Cleanup

Here, we just need to show the removal of the virtual disks from vSphere and the “original” state of our machine.

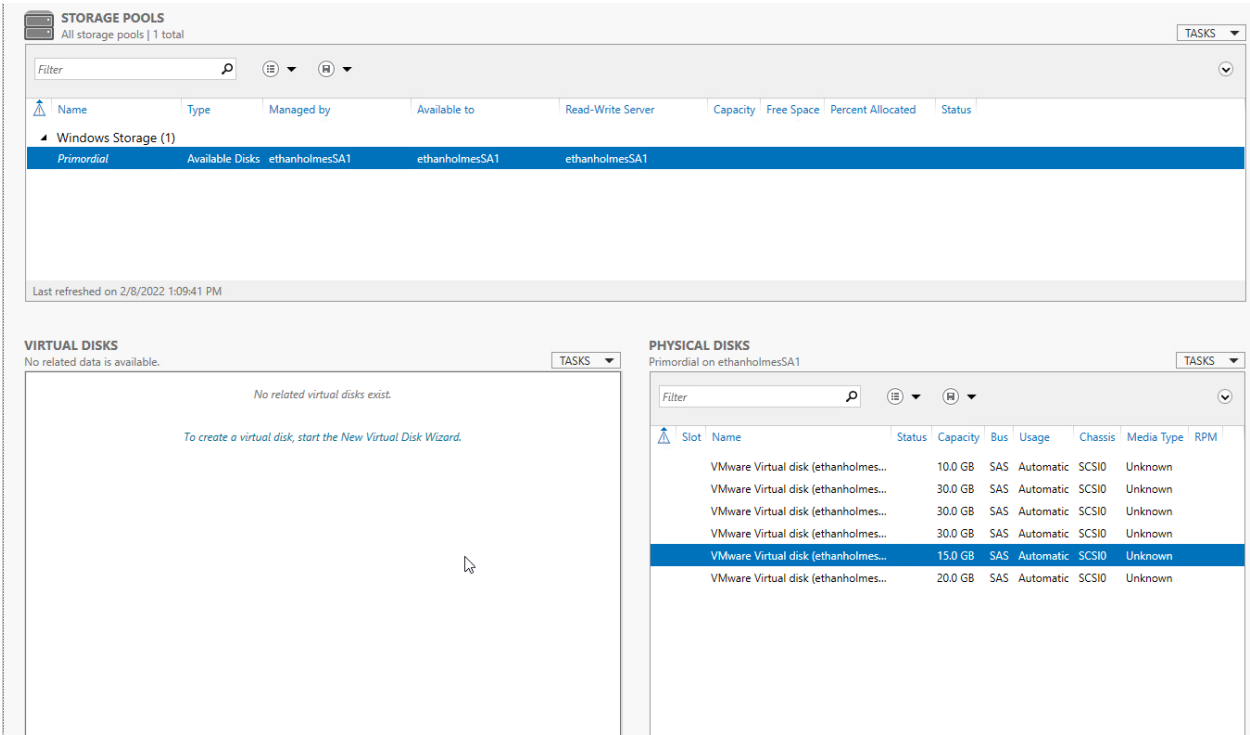


Figure 22: Pre-Cleanup disks in use

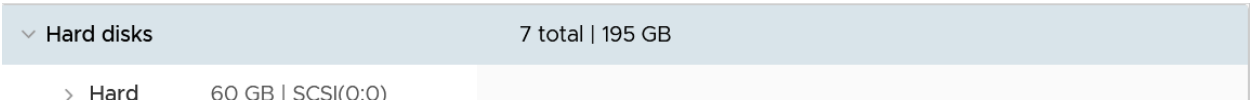


Figure 23: vSphere with 7 hard disks

Here is what it looked like pre-cleanup

Post Cleanup

> Hard disk 1	60 GB
> Hard disk 2	20 GB

Figure 24: vSphere with 2 hard disks

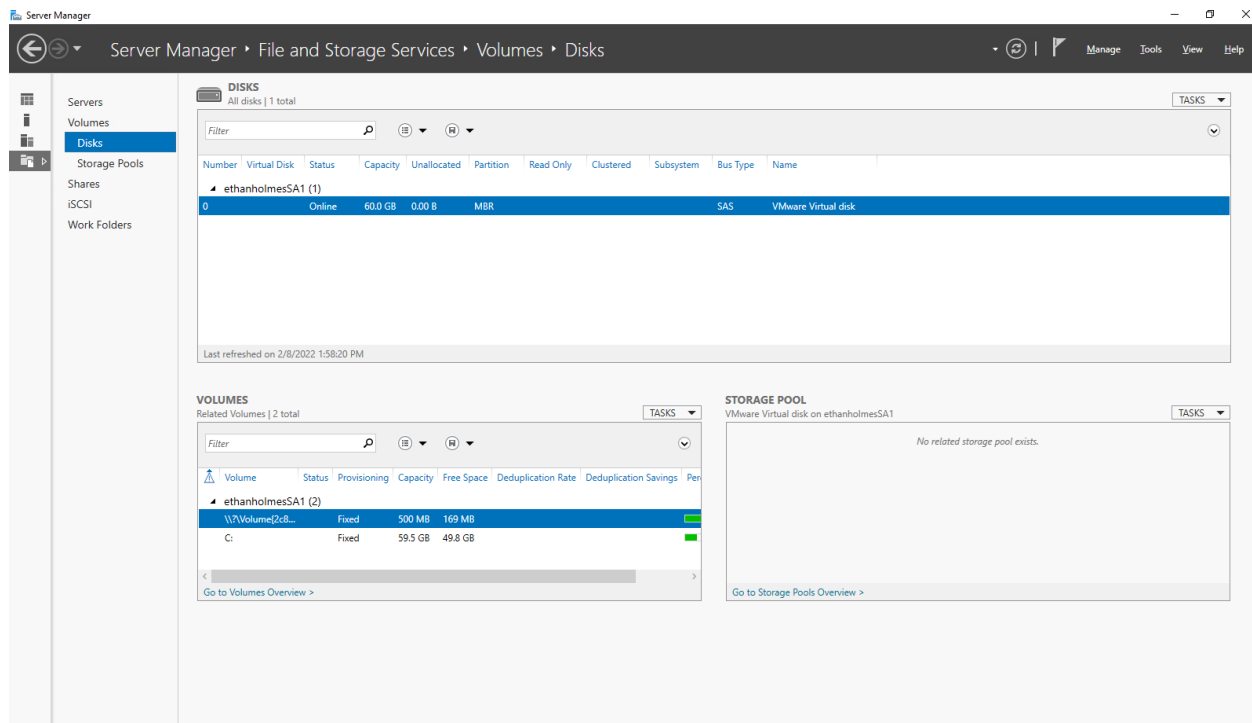


Figure 25: Cleaned up Windows disks in server manager

Final Thoughts:

In this lab I learned lots about setting up storage spaces, both doing it through PowerShell and through GUI. While GUI was easier, I think PowerShell is clearly advantages in many situations, and potentially in speed once you learn commands and can type them out at a faster pace than I did.