Inhalt

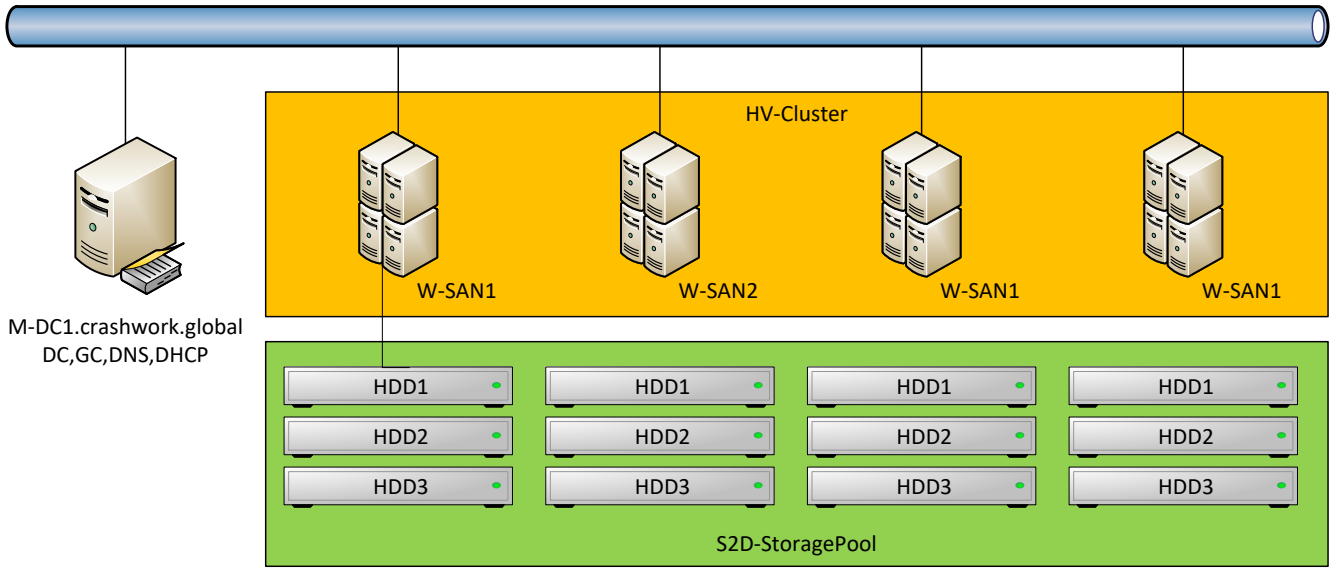
1. Szenario 1
2. Aufbau 1
Vorbereitung des Clusters 1
Aufbau des Clusters 2
Aufbau einer VM im Cluster 5
3. Testphase – was kann der Hyper-Converged-Cluster? 6
VM-LiveMigration 6
Ausfall eines Knotens ohne VM 7
Ausfall des Knotens mit der VM 9
Erweiterung des Clusters um einen Knoten - horizontale Skalierung 12

1. Szenario

Meine Test-Umgebung läuft unter Hyper-V auf einem Windows 10 System. Für die Simulation der Hyper-V-Hosts verwende ich Nested-Virtualization.

An einem virtuellen Switch hängen folgende VMs:

- M-DC1 – der Domänen-Controller für den Cluster. Diese VM nutze ich auch als RemoteManagement-Server
- 4 Server mit der Rolle Hyper-V. Jeder hat zusätzlich zur Betriebssystem-Festplatte 3 weitere lokal angeschlossene Festplatten gleicher Größe



In diesem Test zeige ich, wie man mit Windows Server 2016 Datacenter (RTM) einen Hyper-Converged-Cluster aufbaut. Dieser ist gleichzeitig ein Failover-Cluster für Hyper-V und ein Storage-Space-Direct-Cluster, der das gemeinsame Datenspeichersystem (üblicherweise ein SAN) ersetzt: alle VMs liegen also hochverfügbar als Datendateien über die lokalen Festplatten verteilt, während deren WorkLoad (CPU, RAM, Netzwerk, …) auf einem der Server betrieben wird.

2. Aufbau

Vorbereitung des Clusters

Alle Server sind bereits mit festen IP-Adressen konfiguriert und Mitglied der Domäne.

Alle Aktionen werden durch PowerShell-Befehle vom M-DC1 remote umgesetzt. Damit diese möglichst einfach funktionieren, verwende ich einige Variablen:

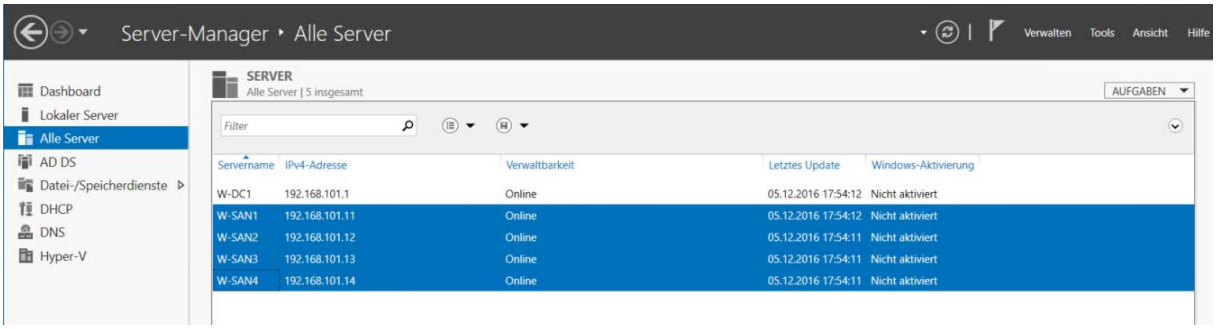
```
$ClusterNodes = 'W-SAN1','W-SAN2','W-SAN3','W-SAN4'
$ClusterName  = 'HC-Cluster'
$ClusterIP    = '192.168.101.10/24'
```

Zuerst werden noch die Rollen und Features installiert:

- Auf dem DC (unserem Steuerungsrechner) werden die Remoteverwaltungstools für den Cluster und Hyper-V installiert
- Auf allen Clusterknoten werden die Rolle Hyper-V und das Feature FailoverClustering installiert

```
Add-WindowsFeature RSAT-Clustering,RSAT-Hyper-V-Tools -IncludeAllSubFeature
Invoke-Command -ComputerName $ClusterNodes -ScriptBlock {Add-WindowsFeature Failover-
   Clustering,Hyper-V -IncludeAllSubFeature -IncludeManagementTools}
```

Zur besseren grafischen Übersicht wird der ServerManager verwendet:



Die Hyper-V-Hosts benötigen noch einen virtuellen Switch, der mit der externen Netzwerkkarte verbunden ist:

```
Invoke-Command -ComputerName $ClusterNodes -ScriptBlock {
    New-VMSwitch -Name LabNet -SwitchType Private
    Set-VMSwitch -Name LabNet -NetAdapterName 'ethernet'
}
```

Dies wird dazu führen, dass die Server kurz nicht über das Netz erreichbar sind.

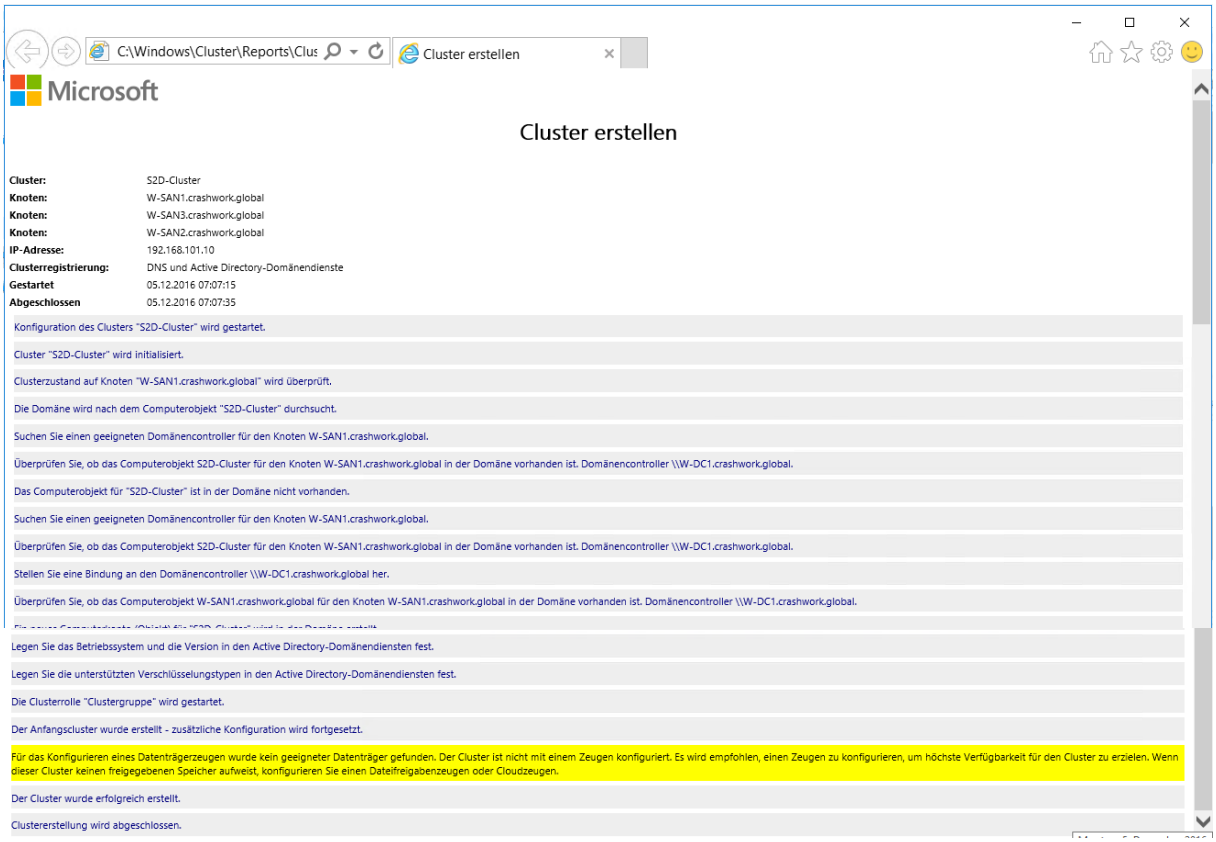
Aufbau des Clusters

Der Cluster kann nun mit diesem Befehl aufgebaut werden. Ich verwende die traditionelle Form des Clusters und nicht den CAAP (Cluster without Administrative Access Points), damit ich den Failover-Cluster-Manager verwenden kann:

```
New-Cluster -Name $ClusterName -Node ($ClusterNodes | Select-Object -SkipLast 1) -NoStorage -
StaticAddress $ClusterIP
```

Der Befehl wird mit einer Warnung beendet:

Schaut man nun in den HTML-Bericht, dann wird das Problem deutlich: es fehlen Ressourcen für eine vernünftige Quorum-Konfiguration.



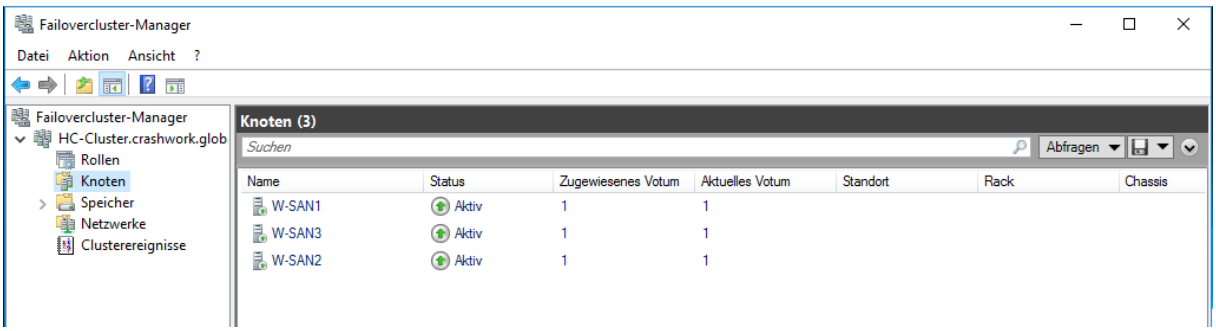
Dies kann so korrigiert werden: auf dem M-DC1 wird eine Freigabe erstellt und alle Clusterknoten erhalten Vollzugriff. Anschließend wird diese Freigabe als Witness-Share im Cluster verwendet:

```
New-Item -Path "c:\$ClusterName" -ItemType Directory | Out-Null
$Identities = $ClusterNodes | ForEach-Object {$_ + '$'}
$Identities += $env:USERNAME
New-SmbShare -Name $ClusterName -Path "c:\$ClusterName" -FullAccess $Identities
Set-ClusterQuorum -FileShareWitness "\\$($env:COMPUTERNAME)\$ClusterName" -Cluster $ClusterName
```

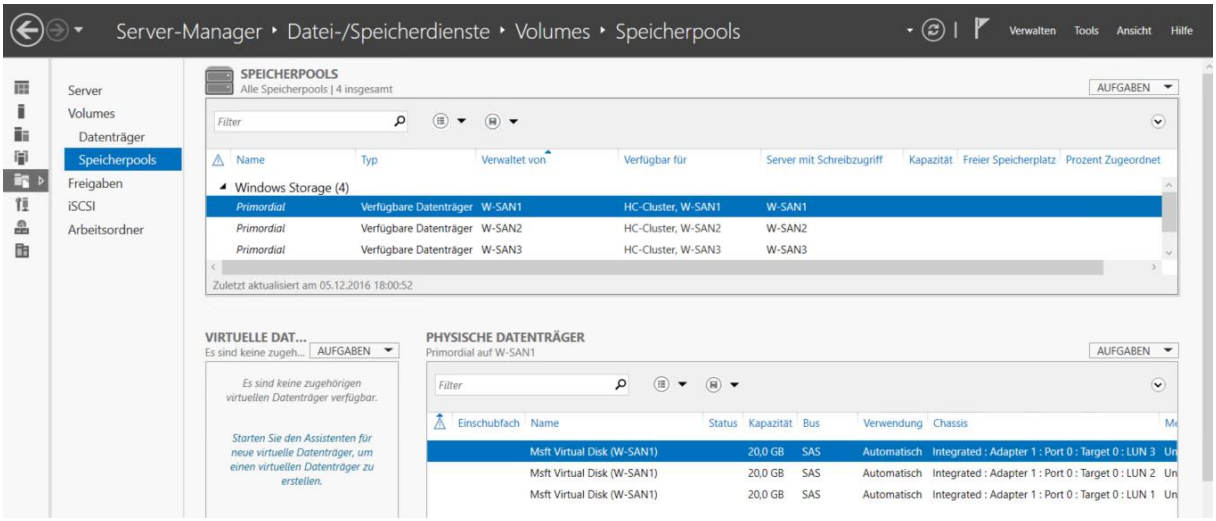
```
Name        ScopeName Path          Description
----        --------- ----          -----------
HC-Cluster *          c:\HC-Cluster

Cluster         : HC-Cluster
QuorumResource  : Dateifreigabenzeuge
QuorumType      : Majority
```

Der Cluster ist nun einsatzbereit:



Im Servermanager sind unter den Speicherpools für jeden Server im Cluster die lokalen Festplatten sichtbar. Aktuell werden diese noch nicht vom Cluster verwaltet:



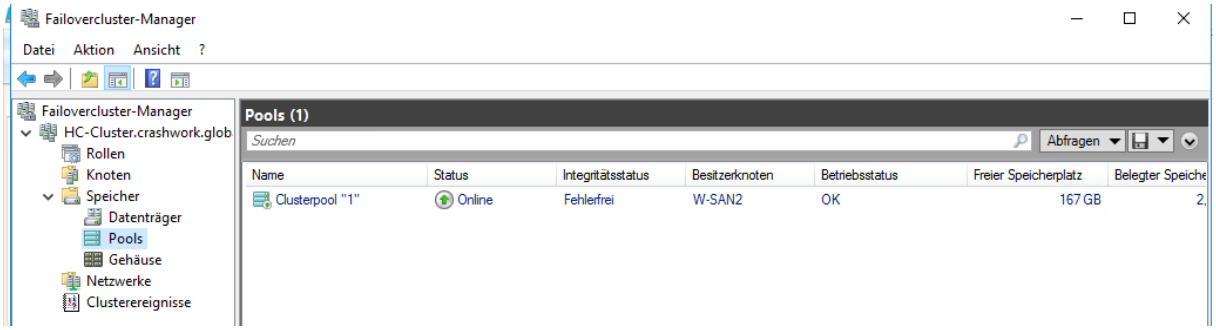
Die Storage-Space-Direct Konfiguration ist über die PowerShell möglich:

```
Enable-ClusterStorageSpacesDirect -CimSession $ClusterName
```

Auch dieser Befehl wird durch eine Ausgabe von Warnungen beendet. Der Befehl würde gerne schnellere SSD für das Caching auf jedem Server verwenden. Sind diese nicht vorhanden, dann erscheinen diese Meldungen

```
PS C:\> Enable-ClusterStorageSpacesDirect -CimSession $ClusterName
WARNUNG: HC-Cluster: 2016/12/05-18:03:47.318 Knoten W-SAN1: Es wurden keine Datenträger für die Verwendung mit dem Cache
 gefunden.
WARNUNG: HC-Cluster: 2016/12/05-18:03:47.338 Knoten W-SAN3: Es wurden keine Datenträger für die Verwendung mit dem Cache
 gefunden.
WARNUNG: HC-Cluster: 2016/12/05-18:03:47.346 Knoten W-SAN2: Es wurden keine Datenträger für die Verwendung mit dem Cache
 gefunden.
WARNUNG: HC-Cluster: 2016/12/05-18:03:47.454 C:\Windows\Cluster\Reports\Enable-ClusterS2D on 2016.12.05-18.03.47.444.htm
```

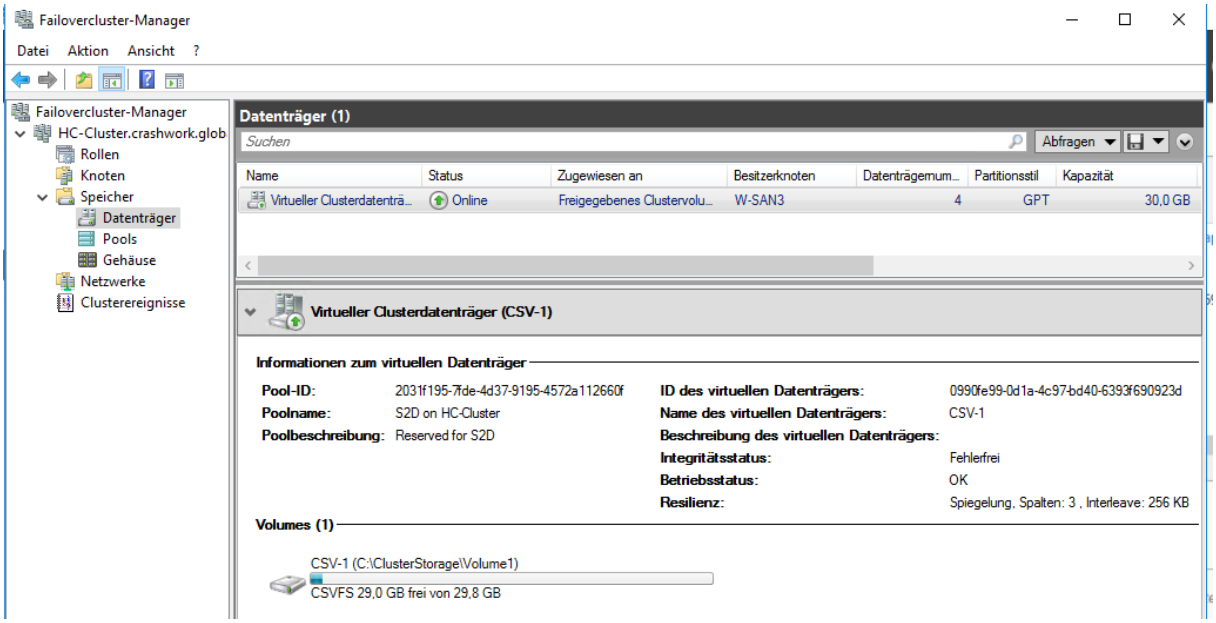
Dennoch wird der neue gemeinsame Pool im Cluster integriert – die Performance wird dann entsprechend niedriger ausfallen:



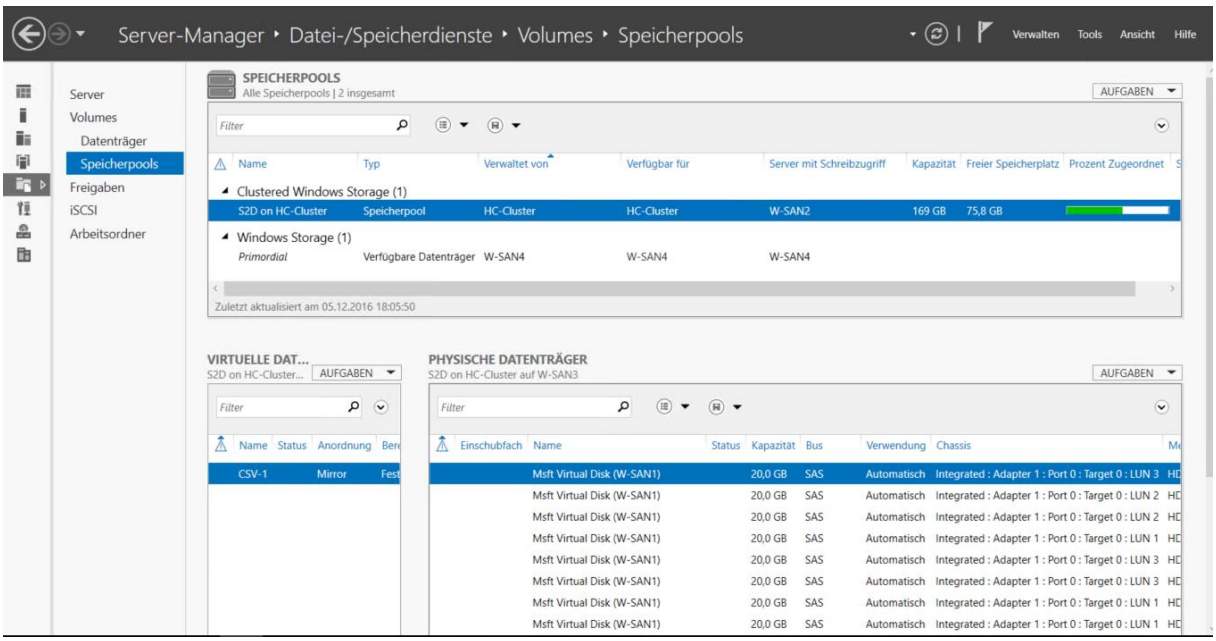
Auf dem Pool benötigen wir nun wie bei einem „traditionellen“ Speicherpool entsprechende virtuelle Disks mit einer Partitionierung und Formatierung – abgesichert durch einen ResilienceType (Mirror oder Parity). So kann ein Volume erstellt werden:

```
New-Volume -StoragePoolFriendlyName "S2D*" -FriendlyName 'CSV-1' -FileSystem CSVFS ReFS -Size 30GB -
CimSession $ClusterName
```

Das Volume wird dann im Cluster-Manager angezeigt und auf allen Cluster-Knoten als Cluster-Shared-Volume eingebunden:



Auch im ServerManager wird beim Cluster nun der Speicherpool mit allen lokalen Festplatten aller Server angezeigt:



Aufbau einer VM im Cluster

Für die Testphase kopiere ich eine VHDX-Datei mit einem vorbereiteten Betriebssystem (Windows Server 2008 R2 als CoreServer – mit Sysprep vorbereitet) in das neue Cluster-Shared-Volume:

```
Copy-Item -Path \\M-DC1\VHDs\HDD0.vhd -Destination '\\W-SAN1\C$\ClusterStorage\Volume1\Test-VM1\hdd0.vhd'
```

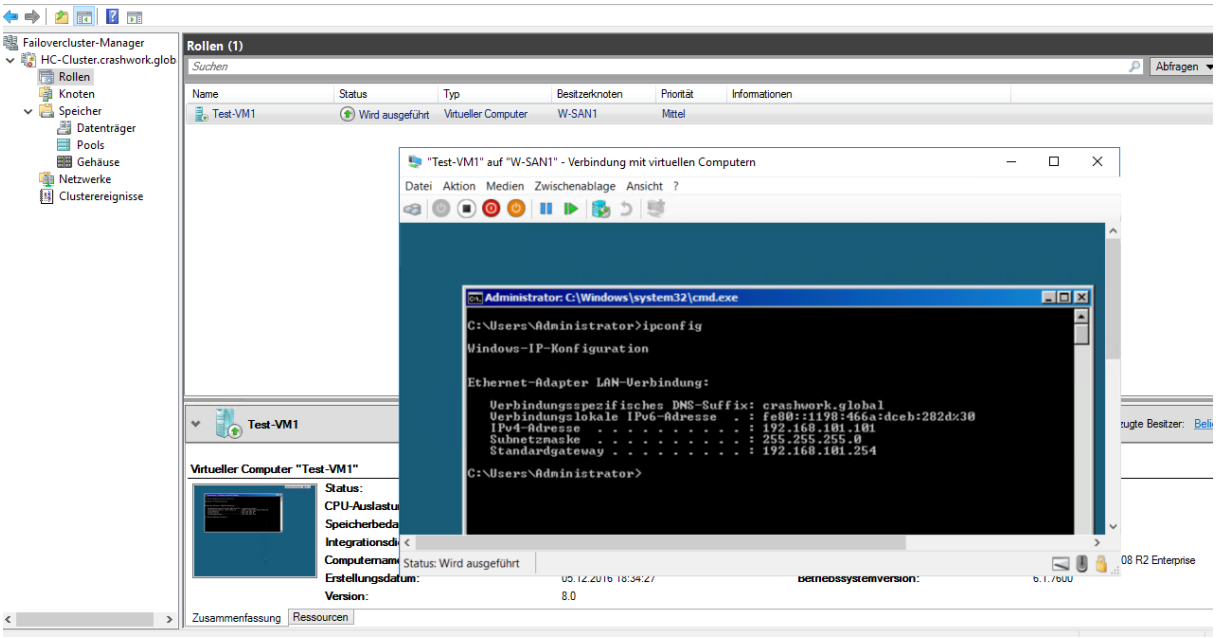
Im CSV wird nun eine neue VM mit der VHD-Datei erstellt:

```
New-VM `
    -Name Test-VM1 `
    -MemoryStartupBytes 1GB `
    -SwitchName LabNet `
    -Path c:\ClusterStorage\Volume1 `
    -VHDPath c:\ClusterStorage\Volume1\Test-VM1\HDD0.vhd `
    -CimSession W-SAN1
```

Die VM muss nun noch dem Cluster als Rolle hinzugefügt werden. Dann kann Sie gestartet werden:

```
Add-VMToCluster -VMName Test-VM1 -Cluster $ClusterName
Start-VM -CimSession $ClusterName -Name Test-VM1
```

Das ist die VM. Das Konsolen-Fenster wird natürlich aus dem Failover-Cluster-Manager geöffnet:

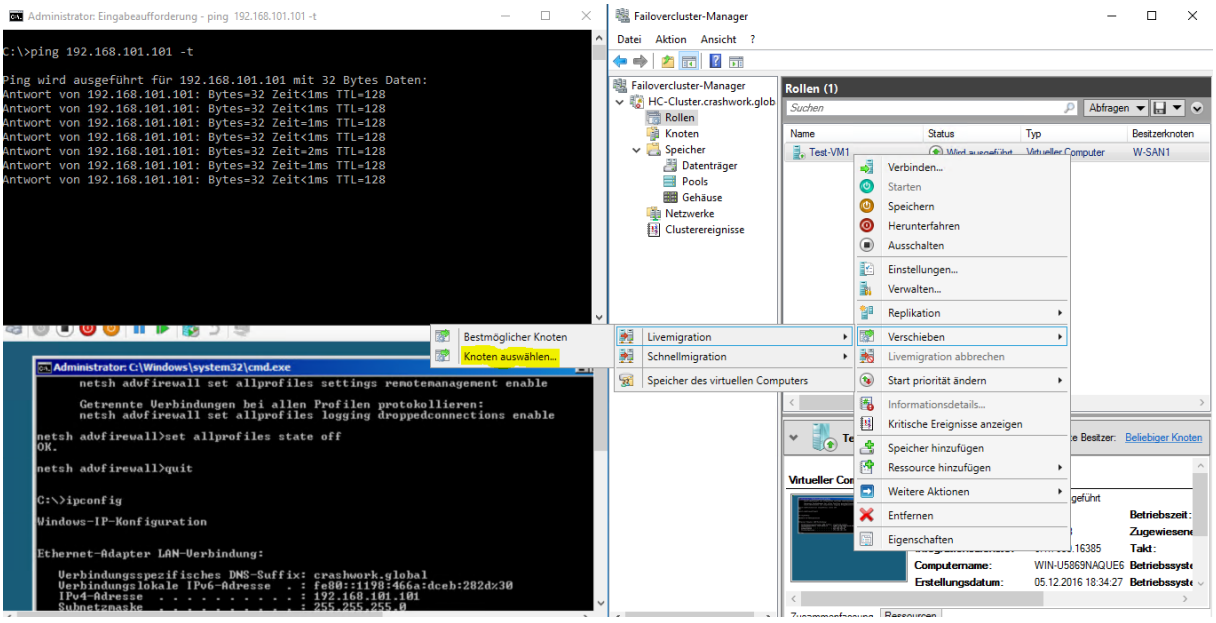


Der Cluster betreibt nun eine VM – ohne gemeinsamen Storage. Die Bytes der VM werden durch den Spiegel des virtuellen Datenträgers auf mehrere Server im Hintergrund verteilt.

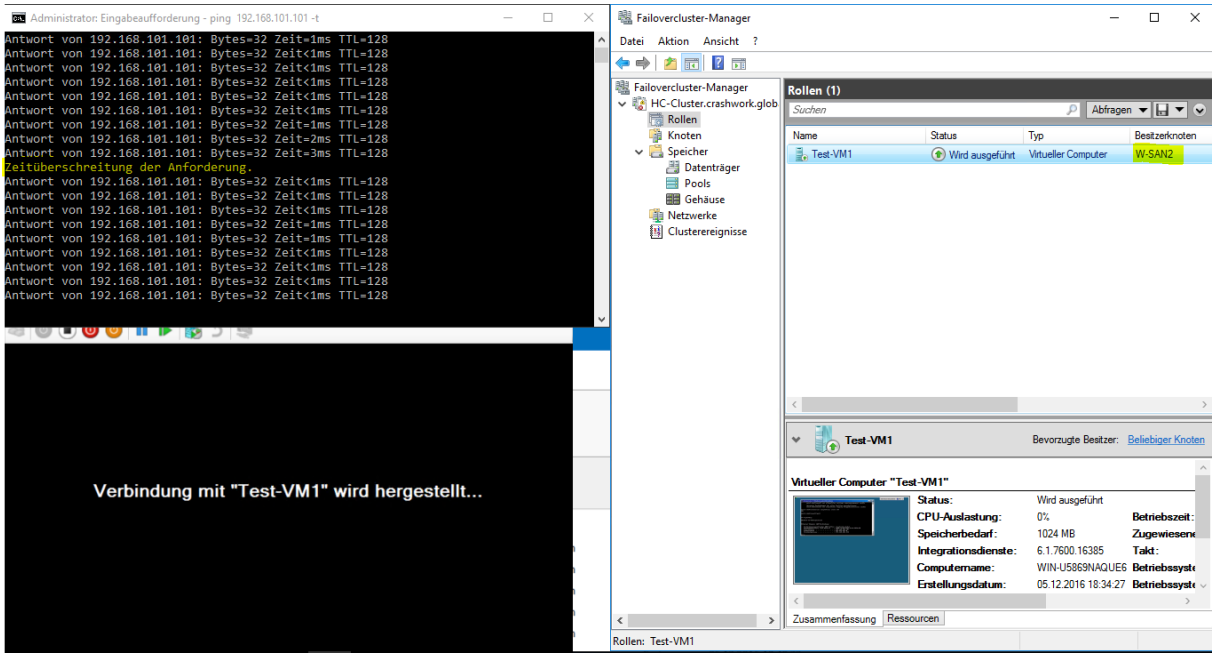
3. Testphase – was kann der Hyper-Converged-Cluster?

VM-LiveMigration

Virtuelle Maschinen können im Cluster im laufenden Betrieb von einem Knoten zu einem anderen übergeben werden. Die Konfigurationsdateien und auch die VHD-Dateien verbleiben dabei natürlich immer im Cluster-Shared-Volume – es werden nur die Workloads (CPU, RAM, Netzwerkbindungen, …) übertragen:



Um den Wechsel des Knotens zu demonstrieren sende ich einen Dauerping vom DC an die VM (oben links), verwende das VM-Konsolenfenster (unten links), um die VM zu betrachten und sehe die VM im Failover-Cluster-Manager (rechts). Noch läuft sie auf W-SAN1. Über die Konsole wechsel ich nun den Hyper-V-Host zu W-SAN2:



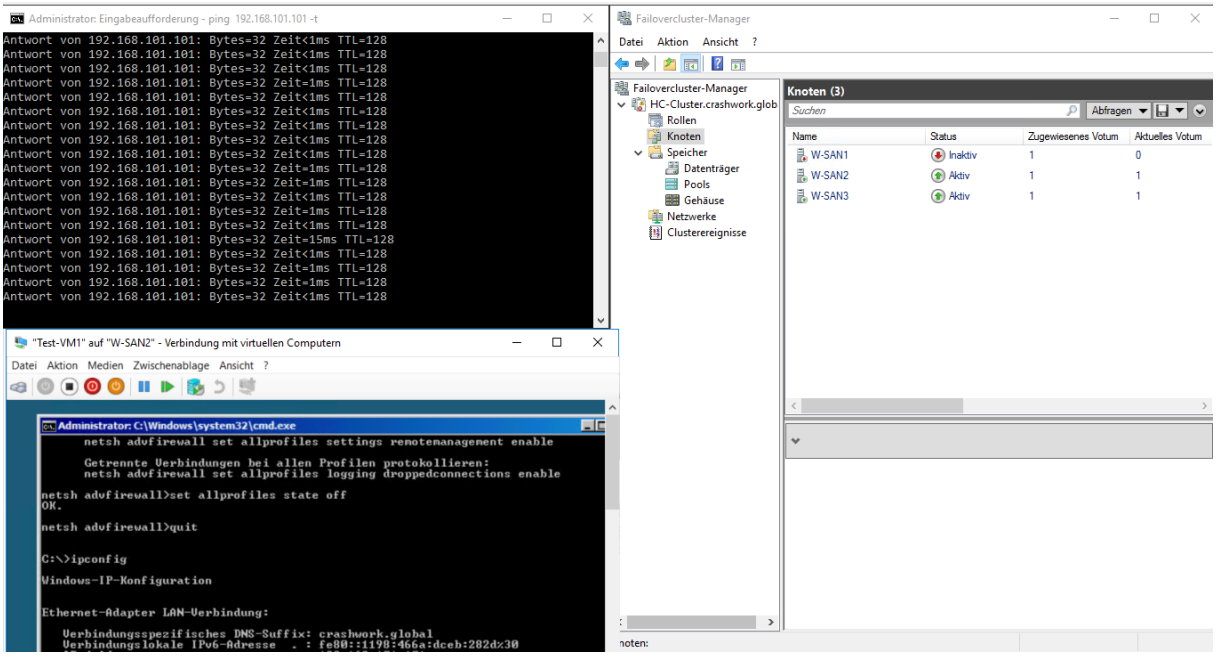
Man erkennt, dass die Verbindung des Dauerpings kurz unterbrochen wurde – das ist normal, da der Switch erst die neue Position der Netzwerkkarte der VM erlernen muss. TCP-Verbindungen werden einfach wiederholt und laufen weiter.

Das Konsolenfenster ist bei meinem Test nicht mit der VM mitgezogen. Durch einen neuen Verbindungsaufbau konnte ich wieder auf den Desktop der VM zugreifen.

Im Cluster-Manager wird die VM nun von W-SAN2 geführt.

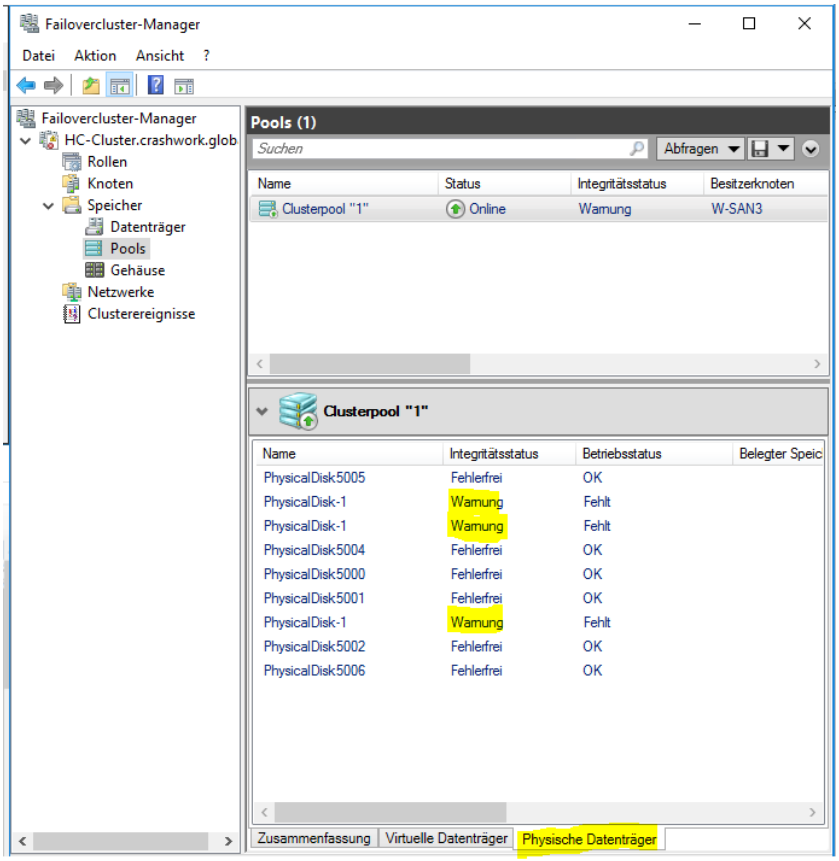
Ausfall eines Knotens ohne VM

Simuliert wird der Ausfall eines Cluster-Knotens durch hartes Ausschalten. Die VM läuft dabei auf einem anderen Knoten. Da sich der Datenspeicher aber über alle Knoten verteilt ist, besteht die Gefahr, dass die VM ausfällt.

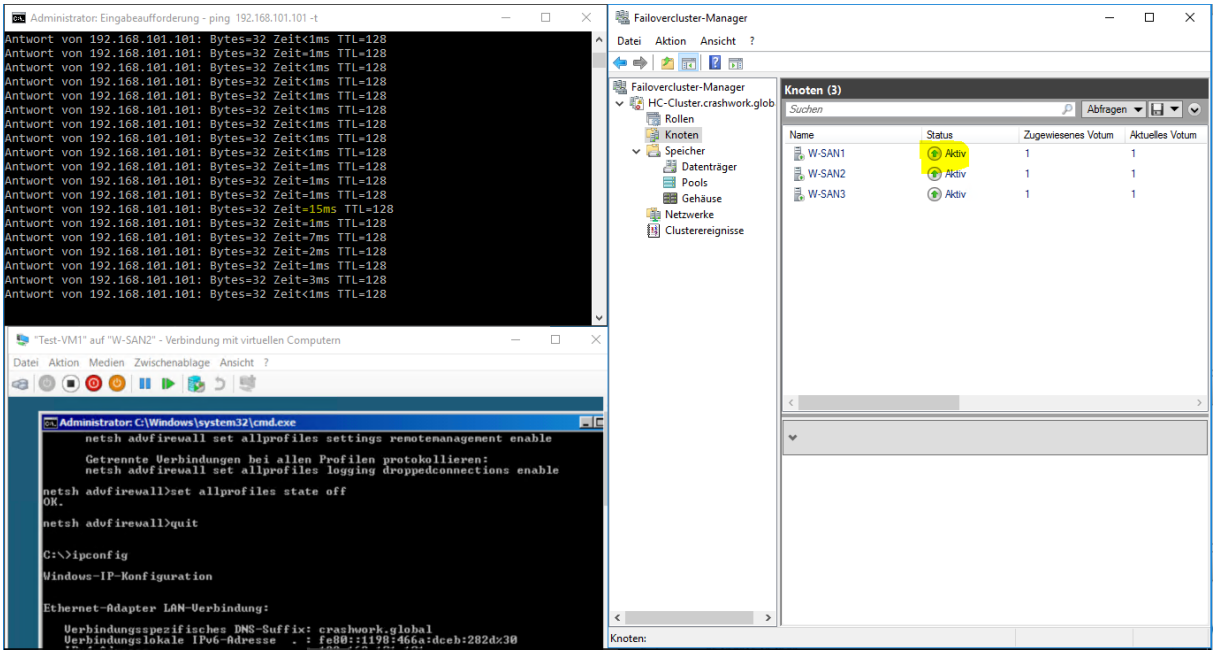


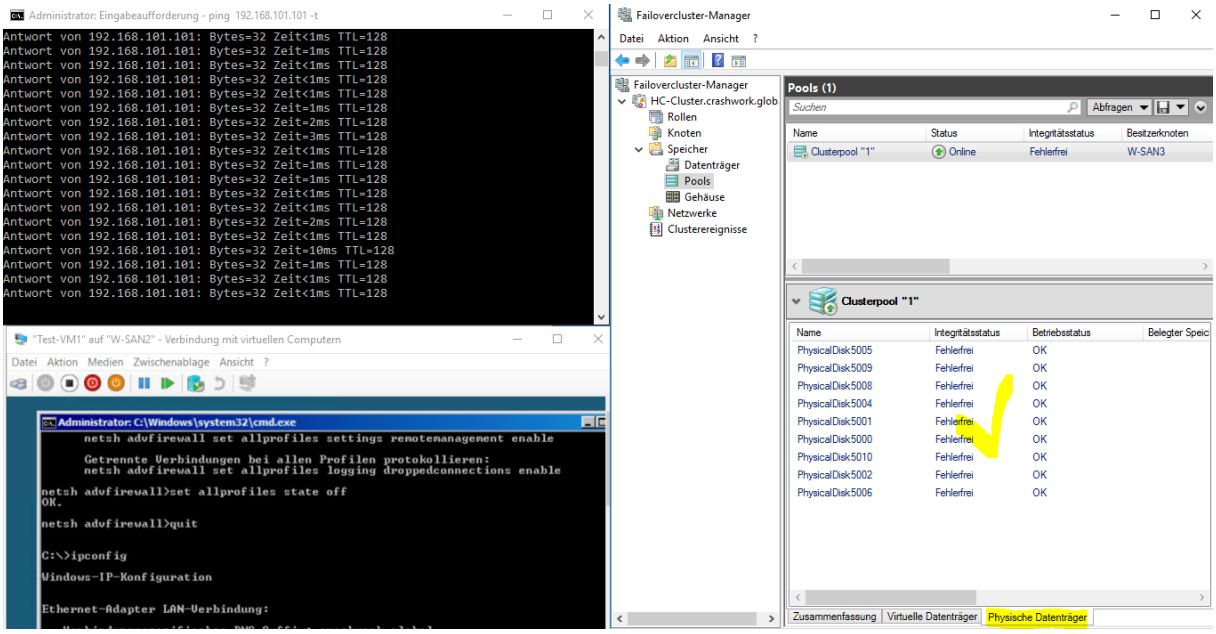
Wie im vorherigen Test ist im Bild oben links der Dauerping vom DC zur VM aktiv, unten links ist die VM-Konsole aktiv und rechts läuft der Failover-Cluster-Manager. Dieser stellt zügig den Ausfall des Knotens W-SAN1 fest. Die VM läuft stabil auf W-SAN2 weiter.

Im Cluster-Manager ist erkennbar, dass 3 der 9 lokalen Festplatten fehlen:



Wird der Cluster-Knoten wieder gestartet, dann tritt er dem Cluster bei und der Normalzustand wird wieder hergestellt:

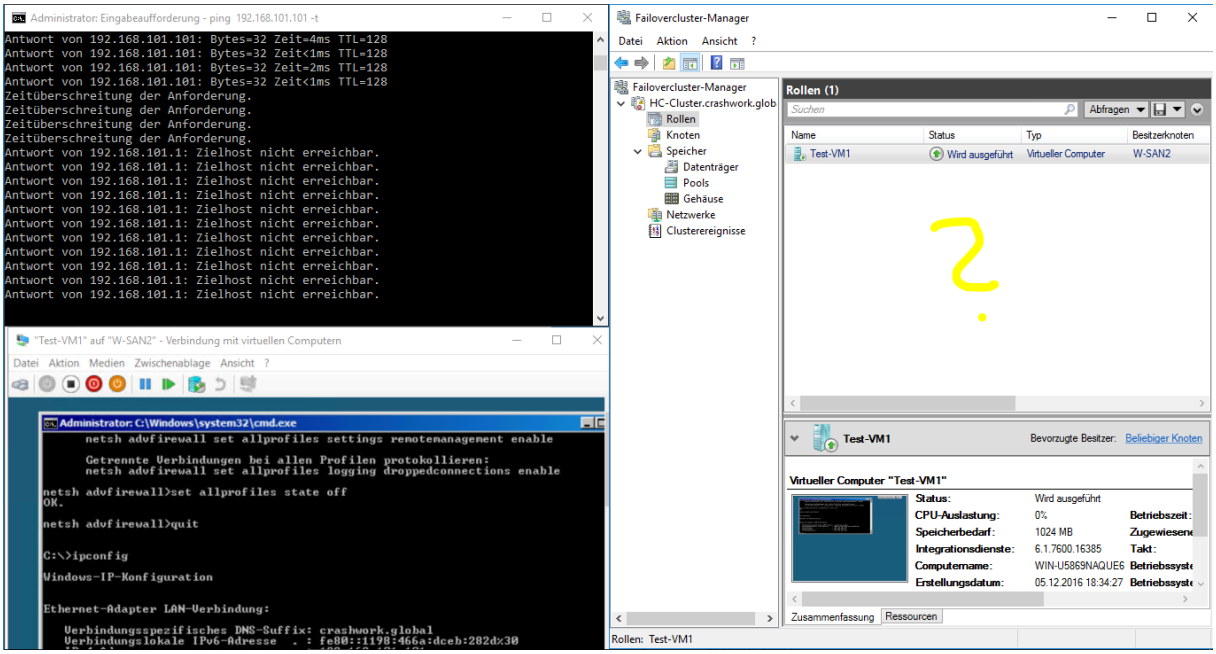


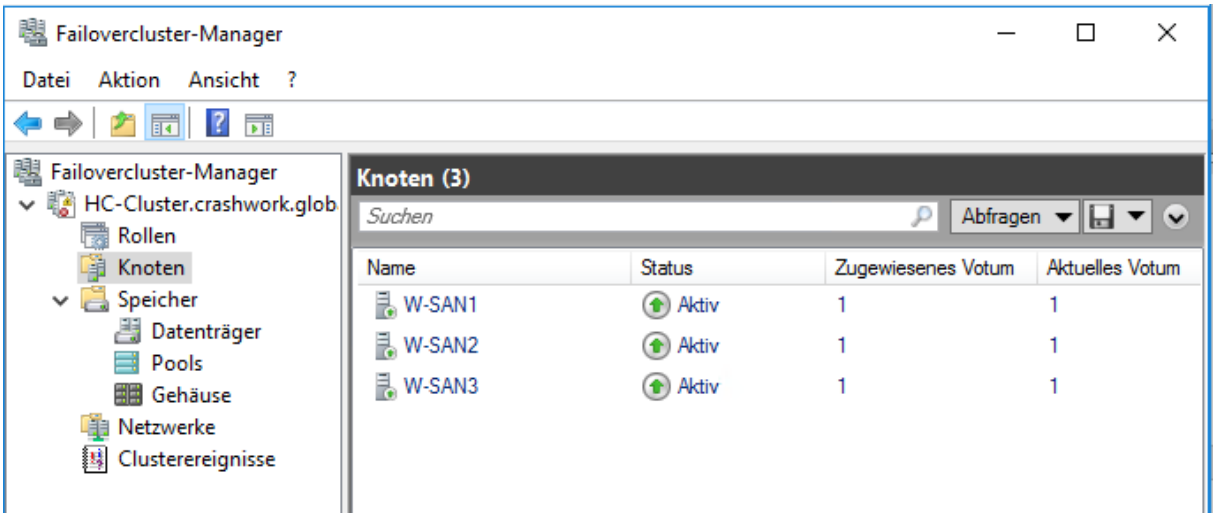


Ausfall des Knotens mit der VM

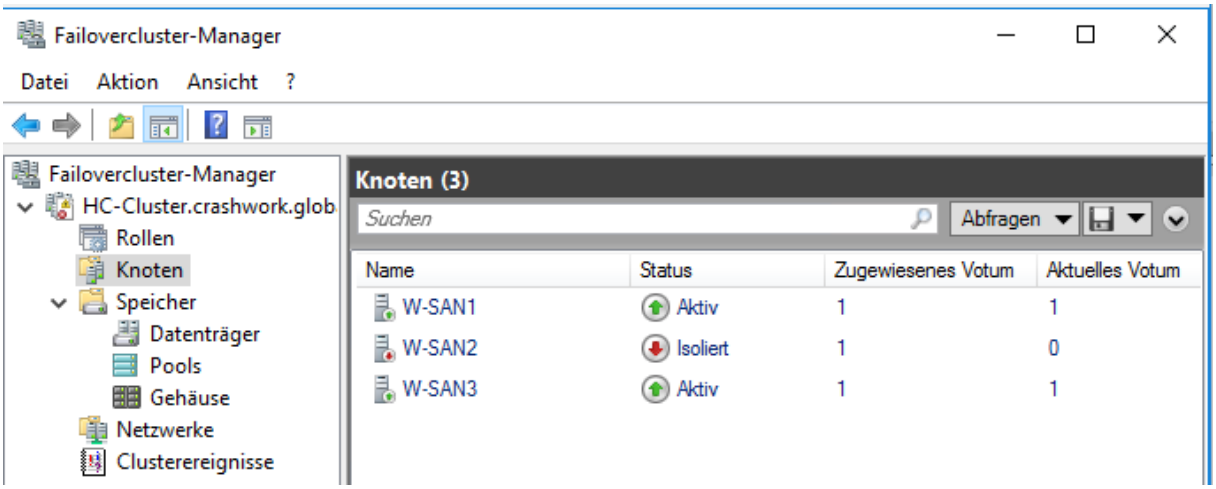
Jetzt wird der Cluster-Knoten hart ausgeschaltet, der die VM betreibt. Der Cluster verliert dabei nicht nur dem Workload der VM (CPU, RAM, Netzwerkverbindungen – diese Lasten stellt aktuell nur der eine Knoten zur Verfügung!), sondern auch einen Teil der lokalen Festplatten im Cluster-Speicherpool

Im Bild oben links läuft wieder unser Dauerping vom DC zur VM. Unten rechts ist die VM-Konsole sichtbar. Rechts im Bild läuft wieder der Cluster-Manager. Nach dem Ausschalten des Cluster-Knotens verliert der Dauerping sofort die Verbindung. Die Konsole „läuft“ weiter – reagiert aber nicht mehr auf Eingaben. Die Aktualisierung im Cluster-Manager verliert an Priorität: der Cluster versucht nun, einen Ausgleich mit einem Failover durchzuführen:

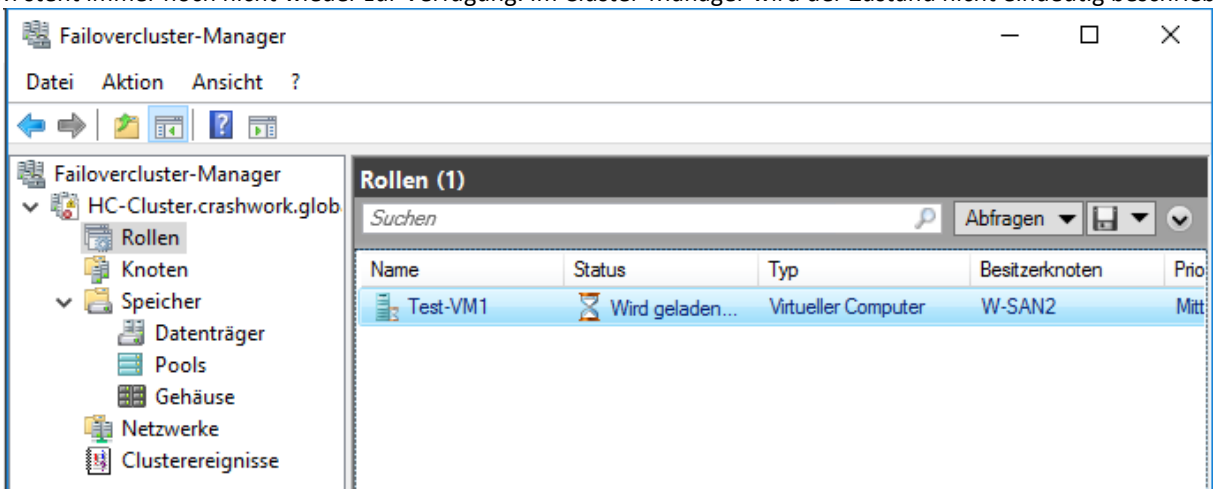




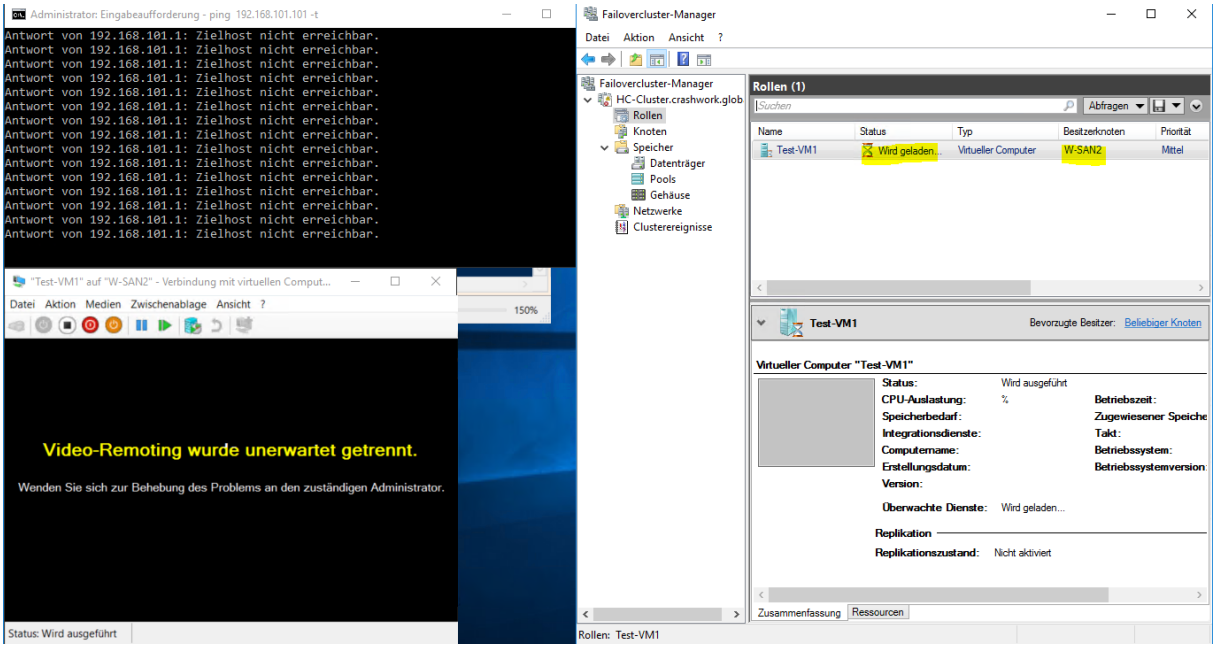
Ca. 30 Sekunden später wird der ausgefallene Cluster-Knoten als isoliert gekennzeichnet:

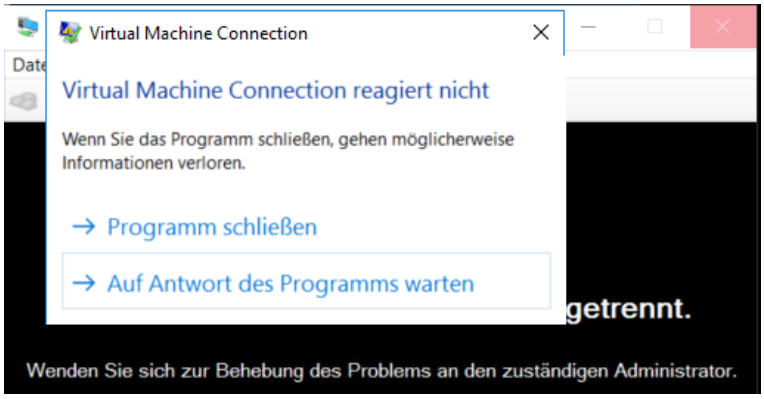


Die VM steht immer noch nicht wieder zur Verfügung. Im Cluster-Manager wird der Zustand nicht eindeutig beschrieben:

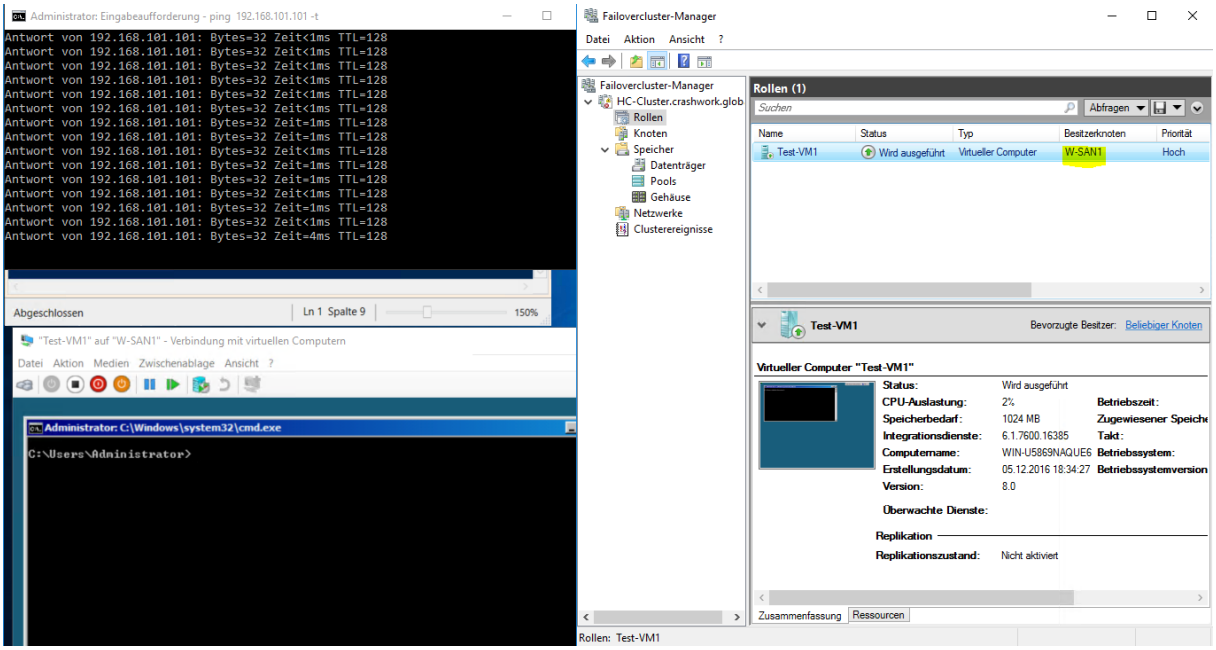


Nun bricht auch die VM-Konsole die „Verbindung“ ab:



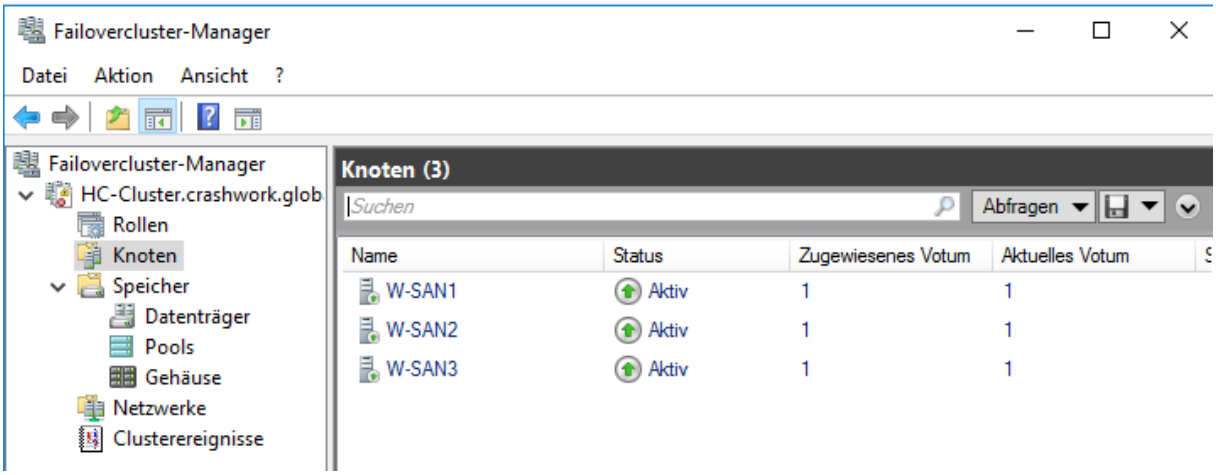


Erst etwa 2 Minuten später ist die VM auf einem anderen Cluster-Knoten neu gestartet worden:



Die VM-Konsole musste neu gestartet werden.

Nun kann der ausgefallene Knoten wieder gestartet werden. Er tritt dem Cluster bei. Im Hintergrund wird der Speicher-Pool wieder ausgeglichen. Die VM verbleibt auf dem Server W-SAN1 – es ist kein Failback konfiguriert:



Erweiterung des Clusters um einen Knoten - horizontale Skalierung

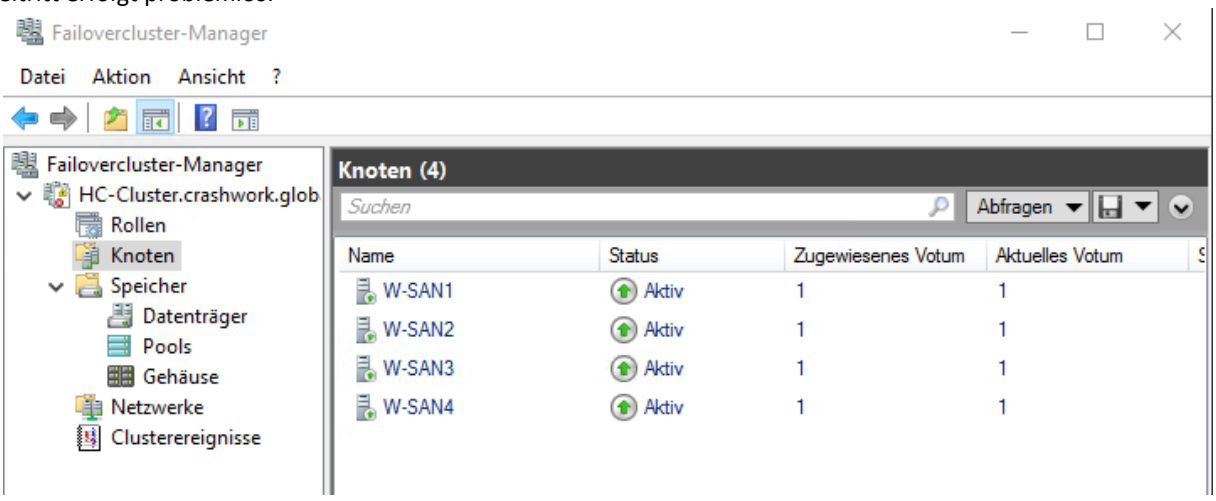
Wenn die Belastung im Cluster steigt oder die Anforderungen an die Ausfallsicherheit erhöht werden, kann eine Erweiterung um zusätzliche Cluster-Knoten sinnvoll sein. Diese müssen dem Hardware-Aufbau der anderen Knoten entsprechen. In unserem Hyper-Converged-Cluster müssen sie also auch lokalen Speicher mitbringen.

Der Server W-SAN4 ist identisch wie die anderen Server konfiguriert: er führt ein kompatibles Hyper-V aus und verfügt über 3 lokale Festplatten gleicher Größe und Bauart.

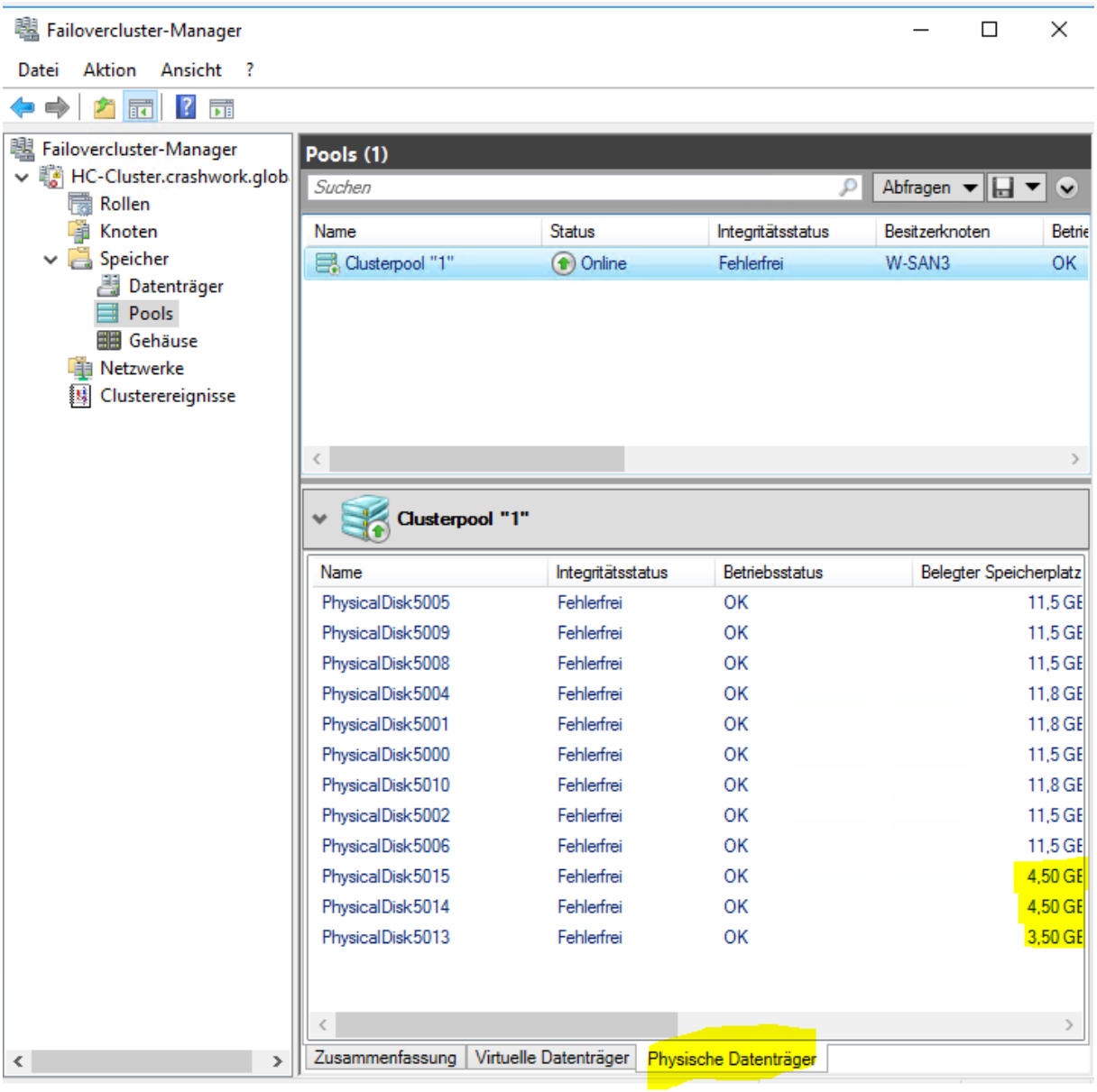
Der Server kann so in den Cluster aufgenommen werden

```
$ClusterNode = 'W-SAN4'
Add-ClusterNode -Name $ClusterNode -Cluster $ClusterName
Get-ClusterStorageSpacesDirect -CimSession $ClusterName
```

Der Beitritt erfolgt problemlos:



Auch erkennt der Cluster, dass es freie Festplatten im neuen Server gibt. Diese werden dem Speicher-Pool zugewiesen:

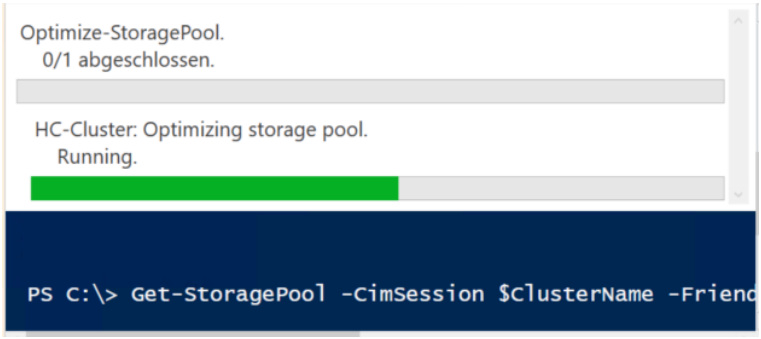


Mit der PowerShell kann ich die Größe des Inhaltes der Festplatten auslesen. Man erkennt deutlich, dass die neuen Festplatten nahezu leer sind, während die VM im Cluster-Shared-Volume die bereits vorhandenen Festplatten gefüllt hat:

```
Zeit         : 07:53:19
W-SAN1_HDD1 : 970
W-SAN1_HDD2 : 966
W-SAN1_HDD3 : 960
W-SAN2_HDD1 : 966
W-SAN2_HDD2 : 962
W-SAN2_HDD3 : 968
W-SAN3_HDD1 : 960
W-SAN3_HDD2 : 1000
W-SAN3_HDD3 : 966
W-SAN4_HDD1 : 36
W-SAN4_HDD2 : 36
W-SAN4_HDD3 : 36
```

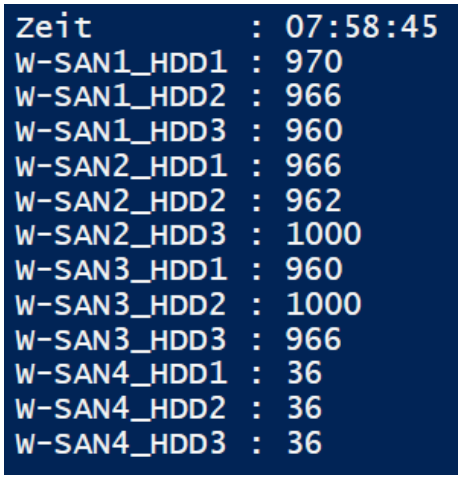
Nun kann/sollte der Speicher-Pool optimiert werden, damit alle Cluster-Knoten die Last gemeinsam tragen (Leistungssteigerung) und die Daten besser verteilt sind (Ausfallsicherheit erhöhen). Dies geht mit der Powershell:

```
Get-StoragePool -CimSession $ClusterName -FriendlyName s2d* | Optimize-StoragePool
```

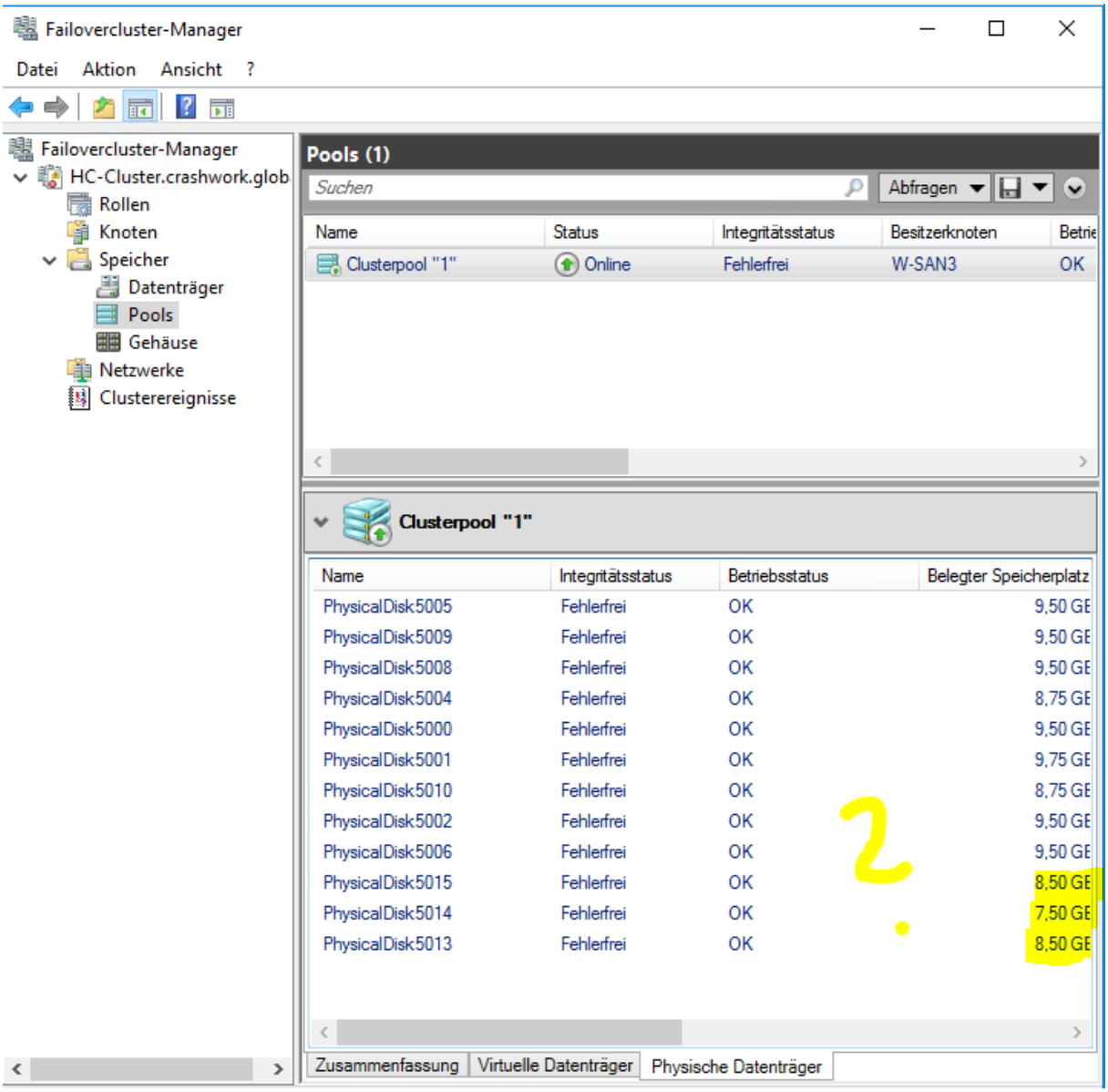


Nun wird die Last auf den Knoten deutlich. Nur einer führt eine kleine VM ohne Last aus! Der Prozentwert bezieht sich hier auf die CPU:

Die Optimierung kann einige Zeit in Anspruch nehmen. Auch nach 5 Minuten ist nichts passiert:



Im Cluster-Manager wird bereits Speicherplatz auf den neuen Festplatten allokiert:



Dennoch verbleibt in meiner Simulation jedes Bit an seinem Speicherplatz. Der Cluster hat wohl entschieden, das die Daten bereits optimal verteilt sind bzw. eine Umverteilung nur Last ohne Verbesserung mitbringt.