

Data Centre ► **Servers**

## The good and the bad in Hyper-V's PowerShell

Look ma, no GUI

By [Trevor Pott](#) 30 May 2013 at 14:13

48 

SHARE ▼



As the great virtual war between Microsoft, VMware and various also-rans has rumbled on, many column inches have been devoted to Microsoft's Hyper-V Server.

You can light up a cluster of 64 nodes and 8,000 virtual machines if you choose, it is said. Hyper-V server is free, you don't have to pay Microsoft

a dime.

That's a great marketing spiel but how usable is it in the real world?

Hyper-V Server is as fully featured as the Datacenter edition of Windows Server, at least when it comes to virtualisation, clustering and relevant storage.

The drawback is that it doesn't natively have a GUI in any real sense. It is a server core-style installation: PowerShell or remote management tools only.

Today, we explore the PowerShell side of the family.

## **Survey your domain**

Theoretically, you can set up your Hyper-V hosts and management clients in any mix-and-match combination of domain-joined or workgroup that you choose.

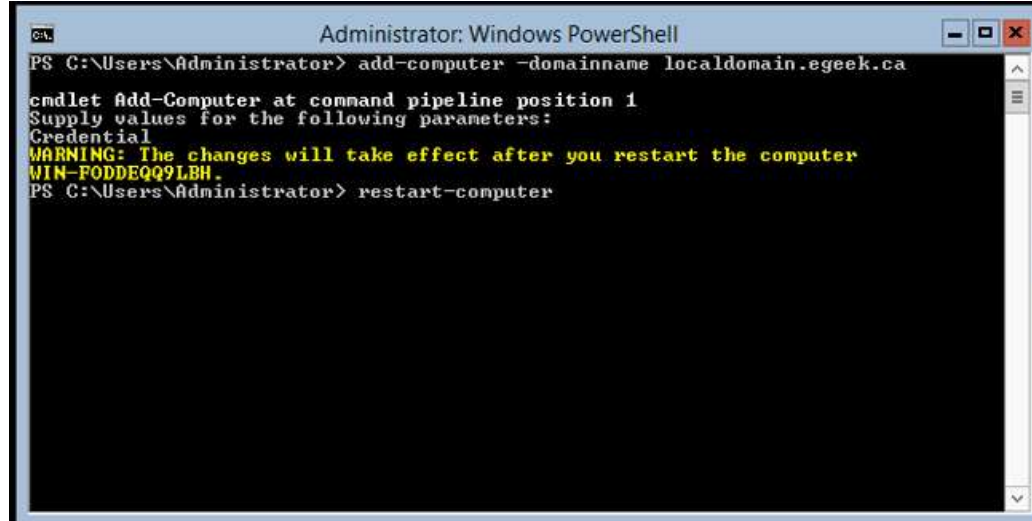
If you choose to do anything other than domain-join every single one of those systems, you deserve to be locked away in a padded room until the end of time.

Theoretically possible does not mean a good idea. If you want to start down the path of madness that is attempting to do anything involving remote PowerShell and CredSSP, I am certainly not going to help you and you deserve every ounce of the sanity-shattering cluster migraine you are about to endure.

As for the rest of us, we will set up our systems on the domain. The PowerShell required is simply two lines:

Add-Computer -DomainName [your domain]

Restart-Computer

A screenshot of a Windows PowerShell window titled "Administrator: Windows PowerShell". The window has a blue title bar and standard Windows window controls. The command prompt shows the following sequence of commands and output:

```
PS C:\Users\Administrator> add-computer -domainname localdomain.egeek.ca  
cmdlet Add-Computer at command pipeline position 1  
Supply values for the following parameters:  
Credential  
WARNING: The changes will take effect after you restart the computer  
WIN-FODDEQQ9LBH.  
PS C:\Users\Administrator> restart-computer
```

Well, that was certainly simple. I may never domain-join a computer using the GUI again. This command should help you not only with Hyper-V Server but with any Windows 7 (or later) system.

This is just part of the default PowerShell loadout and does not require downloading or importing any special modules.

## A question of trust

For anything in Hyper-V Server to work you need to set up trust between systems. Hyper-V's higher functions (Live Migration, for example) rely on a host's ability to perform tasks on another host.

Enabling the GUI remote management tools requires a similar level of trust, one that I will cover in a future article.

To get the basics of a Hyper-V cluster up and running from PowerShell we still need to set up trusts on the computer accounts for the relevant hosts, typically accessed through Active Directory Users and Computers on the Domain Controller. The services that each system will need access to are Microsoft Virtual System Migration Service and CIFS.

Although these can be set up through the GUI, we are here for PowerShell. To get this working through PowerShell you will need to [install](#) the Active Directory.

This module talks to Active Directory Gateway Services, which is installed by default when you install Active Directory on Server 2008 R2, but if you are running Server 2003 or Server 2008 you need to install the [Active Directors Management Gateway Service](#). Make sure that the relevant service is started and that firewall rules exist.

This is a rather involved process so I prefer to "import-module activedirectory" from PowerShell on the domain controller itself.

The relevant PowerShell modules are installed by default during the promotion process. Not good practice, but I am both lazy and not keen to install anything – even PowerShell modules – onto my hosts if I don't have to.

The relevant PowerShell uses the following format:

```
Get-ADComputer [host you want to configure]| Set-ADObject -  
Add @"msDS-AllowedToDelegateTo"="Microsoft Virtual System  
Migration Service/[FQDN of host you are granting access]",  
"Microsoft Virtual System Migration Service/[hostname of  
host you are granting access]", "cifs/[FQDN of host you are
```

```
granting access]", "cifs/[hostname of host you are granting access]"} }
```

Thus in order to get both of my hosts running, I need to run this twice, alternating the system names. The first execution will grant WIN-FODDEQQ9LBH rights on WIN-FODDEQQ9LBH, while the second execution will grant WIN-FODDEQQ9LBH rights on WIN-M90RHKRHJ2L. (Windows auto-generated computer names are catchy, aren't they?)

```
Get-ADComputer WIN-M90RHKRHJ2L | Set-ADObject -Add @{"msDS-AllowedToDelegateTo"="Microsoft Virtual System Migration Service/WIN-FODDEQQ9LBH.localdomain.egeek.ca", "Microsoft Virtual System Migration Service/WIN-FODDEQQ9LBH", "cifs/WIN-FODDEQQ9LBH.localdomain.egeek.ca", "cifs/WIN-FODDEQQ9LBH" }
```

```
Get-ADComputer WIN-FODDEQQ9LBH | Set-ADObject -Add @{"msDS-AllowedToDelegateTo"="Microsoft Virtual System Migration Service/WIN-M90RHKRHJ2L.localdomain.egeek.ca", "Microsoft Virtual System Migration Service/WIN-M90RHKRHJ2L", "cifs/WIN-M90RHKRHJ2L.localdomain.egeek.ca", "cifs/WIN-M90RHKRHJ2L" }
```

## Essential networks

Virtualisation without networking starts to drift into the pointless realm of IT administrative exercises. If we are going to have virtual machines that are of any use to us we should set up some networks.

Special consideration should be given to naming your networks.


Virtualisation is pretty much crippled without the ability to cluster your

hosts and move virtual machines back and forth.

Moving virtual machines back and forth in Hyper-V requires that both the source and destination host have networks that are named the same. The software needs to know where to connect the virtual network cards when you move the virtual machine host A to host B. It relies on the VMSwitch name to do that.

Before we can set about creating virtual switches we need to know what interface numbers Windows has assigned our interfaces. Even on the identical nodes of my stalwart Supermicro [FatTwin](#), Windows still sometimes iterates the NICs differently after a fresh install.

Use the PowerShell command "Get-NetAdapter" to list the network adapters in the server and the names Windows has assigned them.



```
C:\>Administrator: Windows PowerShell
C:\Users\administrator.EGEEK>PowerShell
Windows PowerShell
Copyright (C) 2012 Microsoft Corporation. All rights reserved.

PS C:\Users\administrator.EGEEK> Get-NetAdapter

Name                InterfaceDescription          ifIndex Status
-----
Ethernet 4          Intel(R) 82574L Gigabit Network Co...#2    15 Disconnected
Ethernet 5          Intel(R) Ethernet Converged Network...#2    16 Up
Ethernet 3          Intel(R) Ethernet Converged Network ...    14 Up
Ethernet 2          Intel(R) 82574L Gigabit Network Conn...    13 Disconnected

PS C:\Users\administrator.EGEEK> New-VMSwitch -Name 'UMLAN' -NetAdapterName 'Ethernet 5'

Name SwitchType NetAdapterInterfaceDescription
-----
UMLAN External Intel(R) Ethernet Converged Network Adapter X540-T2 #2

PS C:\Users\administrator.EGEEK>
```

Creating a VMSwitch is fairly straightforward. The PowerShell command is:

```
"New-VMSwitch -Name 'VMSWITCHNAME' -NetAdapterName 'NETWORK  
ADAPTER NAME' -AllowManagementOS:${TRUE|FALSE}."
```

The "AllowManagementOS" bit is especially important: this is the interface you will use to manage your Hyper-V host, so you need to know which NIC to wire where and which IP address you should try to connect to.

The result of these commands for my two servers is as follows:

```
WIN-M90RHKRHJ2L New-VMSwitch -Name 'VMLAN' -NetAdapterName  
'Ethernet 5' New-VMSwitch -Name 'ManageLAN' -NetAdapterName  
'Ethernet 2' -AllowManagementOS:$True
```

```
WIN-FODDEQQ9LBH New-VMSwitch -Name 'VMLAN' -NetAdapterName  
'Ethernet 5' New-VMSwitch -Name 'ManageLAN' -NetAdapterName  
'Ethernet 3' -AllowManagementOS:$True
```

## Failover Clustering

No Hyper-V Server setup is complete without setting up a failover cluster so that if one of the nodes fails your virtual machines will restart on the other node. Failover clusters also make live migrations easier and are generally a good thing.

Setting up Failover Clustering is remarkably easy, although it can quickly become complex depending on how far down the rabbit hole you choose to go. From my domain controller – mostly because I want to show this can be done remotely I do a whole gaggle of hosts simultaneously – I am going to run the command to install the Failover Clustering feature onto the Hyper-V hosts.

The PowerShell command is:

```
"Invoke-Command -ComputerName Node1, Node2, NodeN -  
ScriptBlock {Install-WindowsFeature Failover-Clustering -  
IncludeManagementTools}". That makes the command for my  
environment "Invoke-Command -ComputerName WIN-M90RHKRHJ2L,  
WIN-FODDEQQ9LBH -ScriptBlock {Install-WindowsFeature  
Failover-Clustering -IncludeManagementTools}"
```

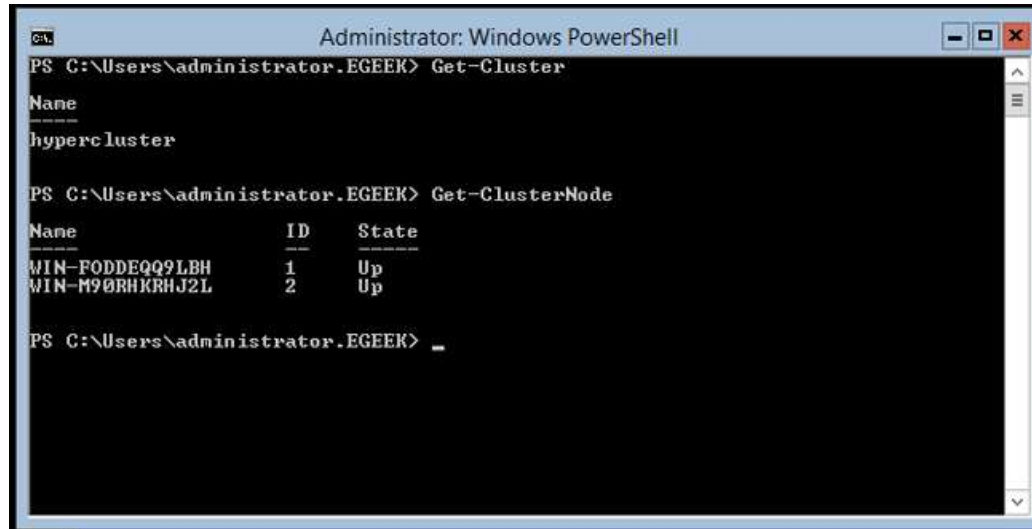
Now PowerShell can read XML natively, so if you don't want to sit there typing out Node1, Node2 and so on all the way to Node64 then you can simply [use XML](#) in place of that chunk of the command. This makes it pretty simple to deploy the Failover Clustering role remotely to your entire data centre.

Now go back to your nodes and run Restart-Computer; after the role is installed they will need it.

Once you have all the nodes you want set up with the relevant role, getting the cluster running is simple. For my environment the PowerShell is "New-Cluster -Name "HyperCluster" -Node 'WIN-M90RHKRHJ2L', 'WIN-FODDEQQ9LBH' ". You can do the math on replacing the node names.

This creates the cluster. If you go poking around, you will notice that this creates a [Microsoft Failover Cluster Virtual Adapter](#) (under your network adapters) with a 169 address. That is normal so leave it alone.



A screenshot of a Windows PowerShell window titled "Administrator: Windows PowerShell". The window shows the output of the "Get-Cluster" and "Get-ClusterNode" commands. The "Get-Cluster" command returns a single entry for a cluster named "hypercluster". The "Get-ClusterNode" command returns a table with three columns: "Name", "ID", and "State". There are two nodes listed: "WIN-FODDEQQ9LBH" with ID 1 and state "Up", and "WIN-M90RHKRHJ2L" with ID 2 and state "Up".

```
C:\
Administrator: Windows PowerShell
PS C:\Users\administrator.EGEEK> Get-Cluster
Name
hypercluster

PS C:\Users\administrator.EGEEK> Get-ClusterNode
Name                ID  State
-----
WIN-FODDEQQ9LBH     1   Up
WIN-M90RHKRHJ2L     2   Up

PS C:\Users\administrator.EGEEK> _
```

## Spawning virtual machines

Creating a basic virtual machine is equally simple, though given the number of possible options it can soon turn into quite a lengthy bit of PowerShell.

Let's say that I want to create a reasonably comfortable virtual desktop instance for myself – 8GB of RAM and 40GB of disk – with the network attached to the VMLAN virtual switch.

I will need some form of storage to put this on. In this case, I am going to use a central SMB 3.0 share called VMs located on the server SMB. The virtual machine is called VulturesForever and the PowerShell looks like this:

```
New-VM -Name VulturesForever -MemoryStartupBytes 8192MB -
Path "\\SMB\VMs\VulturesForever" -NewVHDPATH
"\\SMB\VMs\VulturesForever\OS.vhdx" -NewVHDSIZEBytes 40GB -
```

```
SwitchName VMLAN Add-VMToCluster -VMName VulturesFoeveer -  
Cluster HyperCluster
```

## Cluster Shared Volumes

The PowerShell creation, care and feeding of Cluster Shared Volumes is beyond the scope of this article. But there are a few things you should know if you choose to pursue Cluster Shared Volumes as your storage option.

It is entirely possible to set up Cluster Shared Volumes on the same systems as the Hyper-V compute nodes we have set up here.

With something like the FatTwin that I use, it would be a very usable configuration. If you choose to go down that path, you should take some time to understand the concept of [Quroum Sensing](#) within Windows, the roles of the Witness Disk, and why it might be in your best interests to take a more active role in that rather than let Windows do everything dynamically. (This also covers why having a odd number of nodes is better than an even number of nodes.)

## Scary but clever

When someone lays it all out for you, PowerShell looks like an easy and effective way to administer your Hyper-V servers. Don't be fooled: it both is and it isn't all that it is cracked up to be.

If you happen to be running a large data centre or are a dedicated Hyper-V administrator, then PowerShell is amazing. It can perform complex tasks on a huge array of systems and virtual machines in the blink of an eye.

The flip side of the coin is that PowerShell is not the right choice for the casual administrator. When presented with a GUI a halfway competent administrator familiar with the fundamentals of virtualisation can take over and make a Hyper-V cluster work.

Both the native GUI tools in Windows as well as System Center Virtual Machine Manager have made it far enough that they shouldn't bother someone coming over from a competing product.

PowerShell, however, requires research. We are on PowerShell 3.0 now with PowerShell 4.0 around the corner. The number of commandlets has exploded and there is no way to keep them all memorised.

That means careful research and testing of each command string and script before use – great for automation but totally different from the real-time hands-on functionality provided by the GUI.

PowerShell need not be scary, but it is not the right fit for the administrator who is just looking to get the cluster set up and who will poke it maybe twice a year after that.

In those cases, get the full product with the GUI. The last thing you want to be doing in the middle of an "oh crap why isn't it working" outage is trying to figure out the PowerShell commands required to sort it all out.

Otherwise, give it a go. There is some really neat stuff here and the automation possibilities are endless. ®

**Sponsored:** [Detecting cyber attacks as a small to medium business](#)



**Sign up to our Newsletter** - Get IT in your inbox daily

---

**MORE** [Microsoft](#) [Powershell](#) [Hyper-V](#)

---



## More from The Register

---



### **Microsoft emits another peep at PowerShell 7 with new toys and the return of an old friend**

Commands in (conditional) Chains – the latest Seattle rock sensation?



### **Microsoft doles out PowerShell 7 preview. It works. People like it. We can't find a reason to be sarcastic about it**

Popular admin tool shifts to .Net Core 3.0 amid talk of future features



### **Microsoft: Oh Christmas Tree, Oh Christmas Tree, my PowerShell has gone RC**

[ROUNDUP](#) Redmond celebrates holidays with new toys and a tease of 2020



### **Microsoft stocking fillers: Powershell 7, maybe even next year's Windows 10. But forget about Surface Earbuds**

[ROUNDUP](#) Also: Teams and Slack get the handbags out



### **Latest sneak peek at PowerShell 7 ups the telemetry but... hey... is that an off switch?**

[UPDATED](#) Where is Microsoft and what have you done with them?



### **Microsoft adds Windows module support to PowerShell Core while Amazon unleashes it on Lambda**

Open-source command line botherer says hi! to 1,900+ Windows modules, cmdlets

## **Whitepapers**

---



## **Reduce Redis Enterprise Deployment Cost, Complexity with Intel Optane DC Persistent Memory**

Intel and Redis Labs have prepared this kit to help you reduce Redis Enterprise deployments cost and complexity with 2nd Generation Intel® Xeon® Scalable processors and Intel® Optane™ DC persistent memory.



## **Evolving Datacenters without Complexity**

In this session, we'll talk about how IT leaders are advancing the capabilities of their datacenters to rise to today's challenges. Our guest speaker, Chris Bradford, Product Manager at DataStax will bring first-hand expertise to a discussion with The Register host Elena Perez.



## **Detecting cyber attacks as a small to medium business**

If security by obscurity is no longer an option, and inaction is a risk in itself, what can smaller enterprises do to protect themselves? Endpoint Detection and Response (EDR) solutions can go a long way towards minimising the level of threat, but they need to be chosen and used in the right way.



## **SANS Institute: Cloud Security Survey Results**

Over 47 percent of surveyed organizations store sensitive business intelligence and IP in the cloud ... yet in 2018, a quarter of respondents realized security events due to poor configuration and insecure APIs.

---

**You Might Also Like**

Recommended by

|



Sponsored

### How To Efficiently Empty Your Bowels Every Morning -

Gundry MD | Supplement



### Correction: Last month, we called Zuckerberg a



### Boeing aircraft sales slump to historic lows after



Sponsored

### Unsold 2018 SUVs Going For Pennies On The Dollar.

Yahoo! Search



Sponsored

### Read This Before You Renew Amazon Prime

Wikibuy



Sponsored

### Plastic Surgeon Tells: "Doing This Every Morning Can

Beverly Hills MD



### There are already Chinese components in



Sponsored

### Clogged Gutters? We Have The Solution For Your

leaffilterguards.com

## Sponsored links

Get The Register's Headlines in your inbox daily - quick signup!

---

## ABOUT US

[Who we are](#)

[Under the hood](#)

[Contact us](#)

[Advertise with us](#)

## MORE CONTENT

[Latest News](#)

[Popular Stories](#)

[Forums](#)

[Whitepapers](#)

[Webinars](#)

## SITUATION PUBLISHING

[The Next Platform](#)

[DevClass](#)

[Blocks and Files](#)

[Continuous Lifecycle London](#)

[M-cubed](#)



**The Register** - Independent news and views for the tech community. Part of Situation Publishing

## SIGN UP TO OUR NEWSLETTERS

Join our daily or weekly newsletters, subscribe to a specific section or set [News alerts](#)

SUBSCRIBE