PowerCLI Core Documentation

Release latest

Apr 25, 2017

Introduction

CHAPTER 1

PowerCLI Core

Welcome!

PowerCLI Core uses Microsoft PowerShell Core and .Net Core to enable users of VMware Photon OS, Linux, Mac and Docker to now use the same cmdlets which were previously only available on windows.

PowerCLI Core enables a multi-platform scripting language which will allow you to manage your VMware infrastructure on any OS. Scripts written previously against the windows version are now made portable to a number of operating systems and can simply be loaded and run on these new OS versions without change.

PowerCLI Core can be downloaded from the VMware Flings site: https://labs.vmware.com/flings/powercli-core and used with he below instructions to be deployed.

PowerCLI Core vs PowerCLI for Windows

This initial version provides access to the core vSphere module including over 280 cmdlets allowing you to manage most of the major features of vCenter and ESXi. The below table shows the difference between the windows version and what is currently offered for PowerCLI Core:

Module	Description	PowerCLI for Windows	PowerCLI Core
Core	vCenter and ESXi Cmdlets		
VDS	vSphere Distributed Switch Cmdlets		
CIS	vSphere REST API Cmdlets		
Storage	Storage Cmdlets		X
License	License View Cmdlet		X
VUM	Update Manager Cmdlets		X
Auto Deploy	Auto Deploy Cmdlets		X
Image Builder	Image Builder Cmdlets		X
VCD	vCloud Director Cmdlets		X
vCloud Air	vCloud Air Cmdlets		X

Changelog

Date	Tag	Change
04-18-17	Latest	Added CIS REST API Cmdlets
01-11-17	Latest	Added PowerVRA Module for managing vRealize Automation
01-05-17	Latest	Added PowerNSX Module for managing NSXv

CHAPTER 2

Docker Installation

Docker Image

Step 1 Run the following to download the container from the docker hub:

```
docker pull vmware/powerclicore
```

CHAPTER 3

Mac Installation

Install

These are the detailed instructions.

Step 1 - Download and Install PowerShell for Mac OS X using the instructions and packages from **here_** this will also include the install of homebrew

Step 2 - Make sure you did not miss this step from the PowerShell installation instruction:

```
brew install openssl
brew install curl --with-openssl
```

Step 3 - Create the following directory if it does not exists by running the following command:

```
mkdir -p ~/.local/share/powershell/Modules
```

Step 4 - Extract the PowerCLI modules into the directory you created above by running the following command:

```
unzip PowerCLI.ViCore.zip -d ~/.local/share/powershell/Modules
unzip PowerCLI.Vds.zip -d ~/.local/share/powershell/Modules
```

CHAPTER 4

Linux Installation

Installing on VMware Photon OS 1.0

Step 1 - On the Photon machine Edit a new file in the following location /etc/yum.repos.d/powershell.repo and place the following content in it:

```
[powershell]
name=VMware Photon Linux 1.0(x86_64)
baseurl=https://vmware.bintray.com/powershell
gpgcheck=0
enabled=1
skip_if_unavailable=True
```

Step 2 - Install PowerShell onto Photon OS and create the modules folder needed later:

```
tdnf install -y powershell
mkdir -p ~/.local/share/powershell/Modules
```

Step 3 - From your download machine, copy the PowerCLI Modules from the downloaded fling zip file to the Photon machine, for example use scp as below:

```
scp PowerCLI* root@PHOTON_IP_ADDRESS:/root/.local/share/powershell/Modules
```

Step 4 - From your Photon machine, install Unzip on the photon box

```
tdnf install -y unzip
```

Step 5 - Extract the PowerCLI modules into the directory you copied them into above by running the following command:

```
cd /root/.local/share/powershell/Modules
unzip PowerCLI.ViCore.zip
unzip PowerCLI.Vds.zip
```

Installing on Ubuntu 14.04.5 Server (64-bit)

Step 1 - Download PowerShell for Linux from **here_** on your Ubuntu machine and install as below:

```
curl -SLO https://github.com/PowerShell/PowerShell/releases/download/v6.0.0-alpha.12/
↪powershell_6.0.0-alpha.12-1ubuntu1.14.04.1_amd64.deb
sudo dpkg -i powershell_6.0.0-alpha.12-1ubuntu1.14.04.1_amd64.deb
sudo apt-get install -f
```

Step 2 - Create the following directory if it does not exist by running the following command:

```
mkdir -p ~/.local/share/powershell/Modules
```

Step 3 - From your download machine, copy the PowerCLI Modules from the downloaded fling zip file to the Photon machine under your users folders (replace "username" with your username), for example use scp as below:

```
scp PowerCLI* username@UBUNTU_IP_ADDRESS:/home/username/.local/share/powershell/
↪Modules
```

Step 4 - From the Ubunutu server extract the PowerCLI modules into the directory you created above by running the following command:

```
cd ~/.local/share/powershell/Modules
unzip PowerCLI.ViCore.zip
unzip PowerCLI.Vds.zip
```

CHAPTER 5

Launching PowerCLI

Launching PowerCLI from Mac/Linux

Step 1 - Open terminal

Step 2 - Start Powershell in the terminal by running the following command:

```
powershell
```

Step 3 - Import the PowerCLI Modules into your PowerShell Session:

```
Get-Module -ListAvailable PowerCLI* | Import-Module
```

Step 3a - (Optional - Please Read) If the SSL certificates of your vCenter are not trusted by your OS, disable SSL certificate validation for PowerCLI by running:

```
Set-PowerCLIConfiguration -InvalidCertificateAction Ignore
```

Step 4 - Connect to your vCenter Server using Connect-VIServer

```
PS> Connect-VIServer -Server 192.168.1.51 -User administrator@vSphere.local -
Password VMware1!
Name Port User
---- ---- ----
192.168.1.51 443 VSPHERE.LOCAL\Administrator
```

Launching the PowerCLI Docker container

Open a Terminal and run:

```
docker run --rm -it vmware/powerclicore
```

More options for working with and running the container can be found here.

CHAPTER 6

Frequently Asked Questions

1. I am receiving an error when using Connect-VIServer as below

```
WARNING: Invalid server certificate. Use Set-PowerCLIConfiguration to set the value␣
↪for the InvalidCertificateAction option to Prompt if you'd like to connect once or␣
↪to add a permanent exception for this server.
connect-viserver : 10/17/16 3:00:15 PM        Connect-VIServer                An␣
↪error occurred while sending the request.
At line:1 char:1
connect-viserver 10.192.116.20 -User administrator@vsphere.local -Pas ...
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
CategoryInfo          : NotSpecified: (:) [Connect-VIServer], ViError
FullyQualifiedErrorId : Client20_ConnectivityServiceImpl_Reconnect_Exception,VMware.
↪VimAutomation.ViCore.Cmdlets.Commands.ConnectVIServer
```

This error is because the certificate on your vCenter server is not trusted by the machine you are making the connection from.

To fix this issue, replace the certificate chain on your machine or use the Set-PowerCLIConfiguration cmdlet to ignore certificate issues as below:

```
Set-PowerCLIConfiguration -InvalidCertificateAction Ignore
```

CHAPTER 7

Known Issues

- PowerShell Core does not provide aliases for some of the well known PowerShell cmdlets, watch out for aliases like sleep and sort as these will run native linux commands, it is recommended you use the full cmdlet names like Sort-Object and Start-Sleep for example.
- The Get-VMHostHardware cmdlet has not yet been fully ported to PowerCLI Core and will provide an error when run
- The Get-VMHostPciDevice cmdlet has not yet been fully ported to PowerCLI Core and will provide an error when run
- The Open-VMConsoleWindow cmdlet has not yet been fully ported to PowerCLI Core and will provide an error when run
- The Tag, TagCategory, TagAssignment cmdlets are not supported with vSphere 6.5
- The Content Library Cmdlets have not yet been fully ported to PowerCLI Core and will provide an error when run
- The Credential store Cmdlets have not yet been fully ported to PowerCLI Core and will provide an error when run

CHAPTER 8

Connect Commands

This page contains details on **Connect** commands.

Connect-VIServer

NAME Connect-VIServer

SYNOPSIS This cmdlet establishes a connection to a vCenter Server system.

SYNTAX Connect-VIServer [-Server] <String[]> [-Port <Int32>] [-Protocol <String>] [-Credential <PSCredential>] [-User <String>] [-Password <String>] [-Session <String>] [-NotDefault] [-SaveCredentials] [-AllLinked] [-Force] [<CommonParameters>]

Connect-VIServer -Menu [<CommonParameters>]

DESCRIPTION This cmdlet establishes a connection to a vCenter Server system. The cmdlet starts a new session or re-establishes a previous session with a vCenter Server system using the specified parameters.

When you attempt to connect to a server, the server checks for valid certificates. To set the default behavior of vSphere PowerCLI when no valid certificates are recognized, use the InvalidCertificateAction parameter of the Set-PowerCLIConfiguration cmdlet. For more information about invalid certificates, run 'Get-Help about_invalid_certificates'.

You can have more than one connections to the same server. To disconnect from a server, you need to close all active connections to this server. vSphere PowerCLI supports working with multiple default servers. If you select this option, every time when you connect to a different server using Connect-VIServer, the new server connection is stored in an array variable together with the previously connected servers, unless the NotDefault parameter is set. This variable is named $DefaultVIServers and its initial value is an empty array. When you run a cmdlet and the target servers cannot be determined from the specified parameters, the cmdlet runs against all servers stored in the array variable. To remove a server from the $DefaultVIServers variable, you can either use Disconnect-Server to close all active connections to the server, or modify the value of $DefaultVIServers manually.

If you choose to work with a single default server, when you run a cmdlet and the target servers cannot be determined from the specified parameters, the cmdlet runs against the last connected server. This server is stored in the $defaultVIServer variable, which is updated every time you establish a new connection.

To switch between single and multiple default servers working mode, use DefaultServerMode parameter of the Set-PowerCLIConfiguration cmdlet. Working with multiple default servers will be enabled by default in a future release.

PARAMETERS

-Server <String[]> Specifies the IP address or the DNS name of the vSphere server to which you want to connect. You can also specify a server by providing its IPv6 address enclosed in square brackets, for example [fe80::250:56ff:feb0:74bd%4].

-Port <Int32> Specifies the port on the server you want to use for the connection.

-Protocol <String> Specifies the Internet protocol you want to use for the connection. It can be either http or https.

-Credential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the server. For more information about the server authentication logic of PowerCLI, run “help about_server_authentication”. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-User <String> Specifies the user name you want to use for authenticating with the server. If the Credential parameter is also specified, this parameter is ignored. For more information about the server authentication logic of PowerCLI, run “help about_server_authentication”. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

Note: If the user name contains special characters, enclose the entire string in single quotes (‘).

-Password <String> Specifies the password you want to use for authenticating with the server. If the Credential parameter is also specified, this parameter is ignored. For more information about the server authentication logic of PowerCLI, run “help about_server_authentication”.

Note: If the password contains special characters, enclose the entire string in single quotes (‘).

-Session <String> Specifies the ID of an existing vCenter Server session you want to reestablish.

-NotDefault Indicates that you do not want to include the server to which you connect into the $defaultVIServers variable.

-SaveCredentials Indicates that you want to save the specified credentials in the local credential store.

-AllLinked Indicates whether you want to connect to vCenter Server in linked mode. If you specify $true for the AllLinked parameter and the server to which you want to connect is a part of a federation vCenter Server, you’ll be connected to all members of the linked vCenter Server.

To use this option, PowerCLI must be configured to work in multiple servers connection mode. To configure PowerCLI to support multiple servers connection, specify Multiple for the DefaultVIServerMode parameter of the Set-PowerCLI Configuration cmdlet.

-Force Suppresses all user interface prompts during the cmdlet execution. Currently these include ‘Multiple default servers’ and ‘Invalid certificate action’.

-Menu Indicates that you want to select a connection server from a list of recently connected servers. If Menu is set to $true, the cmdlet retrieves a list of the last visited servers and enters a nested command prompt, so that you can select a server from the list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Connect-VIServer -Server 10.23.112.235 -Protocol https -User admin -Password pass

Connects to a vSphere server using the User and Password parameters.

————— Example 2 —————

C:PS>Connect-VIServer Server -Credential $myCredentialsObject -Port 1234

Connects to a vSphere server by using a credential object.

————— Example 3 —————

C:PS>Connect-VIServer “Server” -SessionId $sessionId

Connect by using a server session ID. Once you connect to a server, you can save the session ID - $serverObject.SessionId, so that you can restore the existing server connection instead of reconnecting.

————— Example 4 —————

C:PS>Connect-VIServer Server

Connect by using integrated authentication. In this case, the credentials you are logged on to your machine must be the same as those for the server.

————— Example 5 —————

C:PS>Connect-VIServer “Server” -User user -Password pass -SaveCredentials

Connect to a server and save the credentials in the credential store. After the credentials are stored, you can connect to the server without specifying them. To get a previously saved credential store item, use the Get-VICredentialStoreItem cmdlet.

————— Example 6 —————

C:PS>Connect-VIServer -Menu

Connect to a server by choosing the server address from a list of previously connected servers.

————— Example 7 —————

C:PS>Connect-VIServer “Server” -AllLinked

Connect to a vSphere server which is a part of a federation vCenter Server. This will Connect you to all vSphere servers in the federation as well.

REMARKS To see the examples, type: “get-help Connect-VIServer -examples”. For more information, type: “get-help Connect-VIServer -detailed”. For technical information, type: “get-help Connect-VIServer -full”. For online help, type: “get-help Connect-VIServer -online”

CHAPTER 9

Add Commands

This page contains details on **Add** commands.

Add-PassthroughDevice

NAME Add-PassthroughDevice

SYNOPSIS This cmdlet attaches pass-through devices to the specified virtual machine.

SYNTAX Add-PassthroughDevice [-VM] <VirtualMachine[]> [-PassthroughDevice] <PassThroughDevice[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet attaches pass-through devices to the specified virtual machine. Note that the value of the ControllerKey property of the returned device might not be up to date, if a new SCSI controller creation process is running in the background.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machine to which you want to attach the passthrough devices.

-PassthroughDevice <PassThroughDevice[]> Specifies the passthrough devices you want to add to the virtual machine.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$scsiDeviceList = Get-PassthroughDevice -VMHost Host -Type Scsi

Add-PassthroughDevice -VM $vm -PassthroughDevice $scsiDeviceList[0]

Adds the first SCSI passthrough device of the Host host to the $vm virtual machine.

REMARKS To see the examples, type: "get-help Add-PassthroughDevice -examples". For more information, type: "get-help Add-PassthroughDevice -detailed". For technical information, type: "get-help Add-PassthroughDevice -full". For online help, type: "get-help Add-PassthroughDevice -online"

Add-VDSwitchPhysicalNetworkAdapter

NAME Add-VDSwitchPhysicalNetworkAdapter

SYNOPSIS This cmdlet adds host physical network adapters to a vSphere distributed switch.

SYNTAX Add-VDSwitchPhysicalNetworkAdapter [-VMHostPhysicalNic] <PhysicalNic[]> [-DistributedSwitch] <DistributedSwitch> [-VirtualNicPortgroup <VDPortgroup[]>] [-VMHostVirtualNic <HostVirtualNic[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet adds host physical network adapters to a vSphere distributed switch.

PARAMETERS

-VMHostPhysicalNic <PhysicalNic[]> Specifies the host physical network adapters that you want to add or migrate to the vSphere distributed switch.

-DistributedSwitch <DistributedSwitch> Specifies the vSphere distributed switch to which you want to add the host physical network adapter.

-VirtualNicPortgroup <VDPortgroup[]> Specifies the port groups to which to attach the host virtual network adapters. Accepts either one port group, or the same number of port groups as the number of virtual network adapters specified. If one port group is specified, all adapters are attached to that port group. If the same number of port groups as the number of virtual network adapters are specified, the first adapter is attached to the first port group, the second adapter - to the second port group, and so on.

-VMHostVirtualNic <HostVirtualNic[]> Specifies the host virtual network adapters to be migrated along with the physical adapter, so that their connectivity is preserved.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmhostNetworkAdapter = Get-VMHost "MyVMHost" | Get-VMHostNetworkAdapter -Physical -Name vmnic2 Get-VDSwitch "MyVDSwitch" | Add-VDSwitchPhysicalNetworkAdapter -VMHostPhysicalNic $vmhostNetworkAdapter

Retrieves the specified physical network adapter from the specified host and adds it to the specified vSphere distributed switch.

———— Example 2 ————

C:PS>$myVMHost = Get-VMHost "MyVMHost" $physicalNic = Get-VMHostNetworkAdapter -VMHost $myVMHost -Name "vmnic0" $virtualNic = Get-VMHostNetworkAdapter -VMHost $myVMHost -Name "vmk0" Get-VDSwitch -Name "MyVDSwitch" | Add-VDSwitchPhysicalNetworkAdapter -VMHostPhysicalNic $physicalNic -VMHostVirtualNic $virtualNic -VirtualNicPortgroup 'MyVDPortGroup'

Migrates a host physical network adapter and a virtual network adapter to a vSphere distributed switch.

REMARKS To see the examples, type: "get-help Add-VDSwitchPhysicalNetworkAdapter -examples". For more information, type: "get-help Add-VDSwitchPhysicalNetworkAdapter -detailed". For technical information, type: "get-help Add-VDSwitchPhysicalNetworkAdapter -full". For online help, type: "get-help Add-VDSwitchPhysicalNetworkAdapter -online"

Add-VDSwitchVMHost

NAME Add-VDSwitchVMHost

SYNOPSIS This cmdlet adds hosts to the specified vSphere distributed switch.

SYNTAX Add-VDSwitchVMHost -VDSwitch <VDSwitch> -VMHost <VMHost[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet adds hosts to the specified vSphere distributed switch. The physical network adapters of the hosts are not connected to the vSphere distributed switch.

PARAMETERS

-VDSwitch <VDSwitch> Specifies the vSphere distributed switch to which you want to add one or more hosts.

-VMHost <VMHost[]> Specifies the hosts that you want to add to the vSphere distributed switch.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch -Name "MyDistributedSwitch" | Add-VDSwitchVMHost -VMHost "VMHost1", "VMHost2"

Adds two hosts to the specified vSphere distributed switch.

REMARKS To see the examples, type: “get-help Add-VDSwitchVMHost -examples”. For more information, type: “get-help Add-VDSwitchVMHost -detailed”. For technical information, type: “get-help Add-VDSwitchVMHost -full”. For online help, type: “get-help Add-VDSwitchVMHost -online”

Add-VirtualSwitchPhysicalNetworkAdapter

NAME Add-VirtualSwitchPhysicalNetworkAdapter

SYNOPSIS This cmdlet adds a host physical NIC to a standard virtual switch.

SYNTAX Add-VirtualSwitchPhysicalNetworkAdapter [-VMHostPhysicalNic] <PhysicalNic[]> [-VirtualSwitch] <VirtualSwitch> [-VirtualNicPortgroup <VirtualPortGroup[]>] [-VMHostVirtualNic <HostVirtualNic[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet adds a host physical NIC to a standard virtual switch. If VMHost virtual network adapters are specified, the cmdlet migrates them to the virtual switch as well.

Note: If VMHost virtual network adapters are specified, the cmdlet migrates them to the respective port groups or creates new ones if VirtualNicPortgroup is not specified.

PARAMETERS

-VMHostPhysicalNic <PhysicalNic[]> Specifies the host physical network adapters that you want to add or migrate to the standard virtual switch.

-VirtualSwitch <VirtualSwitch> Specifies the standard virtual switch to which you want to migrate physical or virtual network adapters.

-VirtualNicPortgroup <VirtualPortGroup[]> Specifies the port groups to which to attach the host virtual network adapters. Accepts the same number of port groups as the number of virtual network adapters specified. The first adapter is attached to the first port group, the second adapter - to the second port group, and so on.

-VMHostVirtualNic <HostVirtualNic[]> Specifies the host virtual network adapters to be migrated along with the physical adapter, so that their connectivity is preserved.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$myVMHostNetworkAdapter = Get-VMhost “MyVMHost” | Get-VMHostNetworkAdapter -Physical -Name vmnic2 Get-VirtualSwitch “MyVirtualSwitch” | Add-VirtualSwitchPhysicalNetworkAdapter -VMHostPhysicalNic $myVMHostNetworkAdapter

Adds a VMHost physical network adapter to the specified distributed switch.

———— Example 2 ————

C:PS>$myVMHost = Get-VMHost 'MyVMHost' $myVDSwitch = Get-VDSwitch 'MyVDSwitch' $physicalNic = Get-VMHostNetworkAdapter -VMHost $myVMHost -VirtualSwitch $myVDSwitch -Name 'vmnic0' $virtualNic = Get-VMHostNetworkAdapter -VMHost $myVMHost -VirtualSwitch $myVDSwitch -Name 'vmk0' $myStandardSwitch = Get-VirtualSwitch -VMHost $myVMHost -Name 'vSwitch0' Add-VirtualSwitchPhysicalNetworkAdapter -VirtualSwitch $myStandardSwitch -VMHostPhysicalNic $physicalNic -VMHostVirtualNic $virtualNic

Migrates VMHost physical and virtual network adapters from a distributed virtual switch to a standard virtual switch.

REMARKS To see the examples, type: "get-help Add-VirtualSwitchPhysicalNetworkAdapter -examples". For more information, type: "get-help Add-VirtualSwitchPhysicalNetworkAdapter -detailed". For technical information, type: "get-help Add-VirtualSwitchPhysicalNetworkAdapter -full". For online help, type: "get-help Add-VirtualSwitchPhysicalNetworkAdapter -online"

Add-VMHost

NAME Add-VMHost

SYNOPSIS This cmdlet adds a host to be managed by a vCenter Server system.

SYNTAX Add-VMHost [-Name] <String> [-Port <Int32>] [-Location] <VIContainer> [-Credential <PSCredential>] [-User <String>] [-Password <String>] [-Force] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet adds a host to be managed by a vCenter Server system. The host is added to the datacenter or folder specified by the Location parameter. One of the User/Password and Credential parameters must be provided in order to authenticate with the host. If both are specified, the Credential parameter is used and the User and Password parameters are ignored.

PARAMETERS

-Name <String> Specifies the name of the host you want to add to a vCenter Server system.

-Port <Int32> Specifies the port on the host you want to use for the connection.

-Location <VIContainer> Specifies a datacenter or folder where you want to place the host.

-Credential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the virtual machine host.

-User <String> Specifies the user name you want to use for authenticating with the host.

-Password <String> Specifies the password you want to use for authenticating with the host.

-Force	Indicates that the cmdlet runs even if the authenticity of the host SSL certificate cannot be verified.
-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myServer = Connect-VIServer -Server 10.23.112.235 Add-VMHost -Server $myServer -Name MyVMHost1 -Location MyDatacenter1 -User MyUsername1 -Password MyPassword1

Adds a VM host to a specified vCenter Server system and provides a username and password for authentication.

————— Example 2 —————

C:PS>$myCredentials = Get-VICredentialStoreItem -File “C:MyCredentials.xml” $myServer = Connect-VIServer -Server 10.23.112.235 Add-VMHost -Server $myServer -Name MyVMHost1 -Location MyDatacenter1 -Credentials $myCredentials

Adds a VM host to a vCenter Server system and specifies a PSCredential object that contains authentication credentials.

————— Example 3 —————

C:PS>$myCredentials = Get-VICredentialStoreItem -File “C:MyCredentials.xml” $myServer = Connect-VIServer -Server 10.23.112.235 Add-VMHost -Server $server -Name MyVMHost1 -Location MyDatacenter1 -Credentials $myCredentials -Port MyVMHostPortNumber1 -Confirm:$false

Adds a VM host to a vCenter Server system without asking for confirmation and specifies a port on the host for connecting.

————— Example 4 —————

C:PS>$myCredentials = Get-VICredentialStoreItem -File “C:MyCredentials.xml” $myServer = Connect-VIServer -Server 10.23.112.235 Add-VMHost -Server $myServer -Name MyVMHost1 -Location MyDataCenter1 -Credentials $myCredentials -Port MyVMHostPortNumber1 -Force

Adds a VM host to a vCenter Server system even if the authenticity of the host SSL certificate cannot be verified.

REMARKS To see the examples, type: “get-help Add-VMHost -examples”. For more information, type: “get-help Add-VMHost -detailed”. For technical information, type: “get-help Add-VMHost -full”. For online help, type: “get-help Add-VMHost -online”

Add-VMHostNtpServer

NAME Add-VMHostNtpServer

SYNOPSIS This cmdlet adds the specified NTP servers to the NTP server list of the specified hosts.

SYNTAX Add-VMHostNtpServer [-NtpServer] <String[]> [-VMHost] <VMHost[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet adds the specified NTP servers to the NTP server list of the specified hosts. If a server is already in the list, a non-terminating error is generated and a duplicate is not created.

PARAMETERS

-NtpServer <String[]> Specifies the domain name or the IP address of the NTP server you want to add to the host.

-VMHost <VMHost[]> Specifies a host to which you want to add the NTP server.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Add-VmHostNtpServer -NtpServer "ntp-server-name.com" -VMHost $vmhost

Adds the NTP server with a domain name "ntp-server-name.com" to the virtual machine hosts stored in the $vmhost variable.

———— Example 2 ————

C:PS>Add-VmHostNtpServer -NtpServer "192.168.1.5" -VMHost (Get-VMHost)

Adds the NTP server with an IP address "192.168.1.5" to the virtual machine hosts pipelined through the Get-VMHost cmdlet.

REMARKS To see the examples, type: "get-help Add-VMHostNtpServer -examples". For more information, type: "get-help Add-VMHostNtpServer -detailed". For technical information, type: "get-help Add-VMHostNtpServer -full". For online help, type: "get-help Add-VMHostNtpServer -online"

CHAPTER 10

Copy Commands

This page contains details on **Copy** commands.

Copy-DatastoreItem

NAME Copy-DatastoreItem

SYNOPSIS This cmdlet copies items between datastores and between a datastore and a local file system provider.

SYNTAX Copy-DatastoreItem [-Item] <Object[]> [[-Destination] <Object>] [-Force] [-PassThru] [-Recurse] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet copies items between datastores and between a datastore and a local file system provider.

PARAMETERS

-Item <Object[]> Specifies the datastore item you want to copy. You can use a string to provide a relative path to the item in the current provider location.

-Destination <Object> Specifies the destination where you want to copy the datastore item. You can use a string to specify a relative path to the destination object in the current provider location.

-Force	Indicates whether to overwrite all items with the same name at the provided destination.
-PassThru	Indicates that the cmdlet returns the copied item.
-Recurse	Indicates that you want to copy not only the item, but its children items as well.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Copy-DatastoreItem vmstore:DatacenterStorage1MyVM* c:VMFolderMyVM

Copies the contents of a datastore folder in a local folder.

————— Example 2 —————

C:PS>Copy-DatastoreItem c:VMFolderMyVM* vmstore:DatacenterStorage1NewVM-Force

Copies the contents of a local folder into a datastore folder. If the destination folder does not exist, the Force parameter enforces its creation.

————— Example 3 —————

C:PS>Copy-DatastoreItem c:VMFolder* vmstore:DatacenterStorage1VMs-Force -Recurse

Copies recursively the contents of a local folder into a datastore folder.

————— Example 4 —————

C:PS>Copy-DatastoreItem Windows.ISO vmstore:DatacenterStorage1ISOFilesWinXPSP3.iso

Copies a file into a datastore folder and changes the file name.

REMARKS To see the examples, type: "get-help Copy-DatastoreItem -examples". For more information, type: "get-help Copy-DatastoreItem -detailed". For technical information, type: "get-help Copy-DatastoreItem -full". For online help, type: "get-help Copy-DatastoreItem -online"

Copy-HardDisk

NAME Copy-HardDisk

SYNOPSIS Copies a virtual hard disk to another destination.

SYNTAX Copy-HardDisk [-HardDisk] <HardDisk[]> [-DestinationPath] <String> [-DestinationStorageFormat <VirtualDiskStorageFormat>] [-Force] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION Copies a virtual hard disk to another destination specified by the DestinationPath parameter. DestinationPath must be a datastore path to the destination folder.

PARAMETERS

-HardDisk <HardDisk[]> Specifies the virtual hard disk you want to copy.

-DestinationPath <String> Specifies the datastore path to the folder where you want to copy the hard disk. The datastore name is included in the path in square braces.

-DestinationStorageFormat <VirtualDiskStorageFormat> Specifies the type of the hard disk copy. The valid values are Thin, Thick, and EagerZeroedThick. This parameter is only applicable when you are connected to an ESX/ESXi host.

-Force	Indicates whether to overwrite all disks with the same name at the provided destination.
-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-HardDisk -VM $vm | Copy-HardDisk "[Storage1]/"

Retrieves the hard disks of a virtual machine and copies them into the storage1 root folder.

———— Example 2 ————

C:PS>Copy-HardDisk -HardDisk $hdd -DestinationPath "[Storage1] vms/disks" -DestinationStorageFormat Thick

Copies the $hdd hard disk to the "vms/disks"location on storage1 and changes the storage format of the destination disk to Thick.

REMARKS To see the examples, type: "get-help Copy-HardDisk -examples". For more information, type: "get-help Copy-HardDisk -detailed". For technical information, type: "get-help Copy-HardDisk -full". For online help, type: "get-help Copy-HardDisk -online"

Copy-VMGuestFile

NAME Copy-VMGuestFile

SYNOPSIS This cmdlet copies files and folders from and to the guest OS of the specified virtual machines using VMware Tools.

SYNTAX Copy-VMGuestFile [-Source] <String[]> [-Destination] <String> -LocalToGuest [-Force] [-VM] <VirtualMachine[]> [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-GuestCredential <PSCredential>] [-GuestUser <String>] [-GuestPassword <SecureString>] [-ToolsWaitSecs <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Copy-VMGuestFile [-Source] <String[]> [-Destination] <String> -GuestToLocal [-Force] [-VM] <VirtualMachine[]> [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-GuestCredential <PSCredential>] [-GuestUser <String>] [-GuestPassword <SecureString>] [-ToolsWaitSecs <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet copies files and folders from and to the guest OS of the specified virtual machines using VMware Tools. If a file with the same name exists in the destination directory, it is overwritten. Use the GuestUser and GuestPassword, or GuestCredential parameters to authenticate when connecting to the VMware Tools. To authenticate with the host, use the HostUser and HostPassword, or HostCredential parameters. SSPI is not supported. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Modify privilege.

PARAMETERS

-Source <String[]> Specifies the file you want to copy. If the file is on a virtual machine, specifies the absolute file path. Relative file paths are supported only when copying files from a local storage. Wildcards are allowed only at the end of the source path. If you are copying files from the guest operating system of a virtual machine to a local directory, the Source parameter supports wildcards only on vCenter Server 5.0 and later.

-Destination <String> Specifies the destination path where you want to copy the file. If the destination points to a virtual machine, specify the absolute file path. Relative destination paths are supported only when copying files to a local storage.

-LocalToGuest	Indicates that you want to copy a file from a local directory to the guest operating system of the virtual machine.
-Force	Indicates that the non-existing directories in the specified destination path are automatically created.

-VM <VirtualMachine[]> Specifies the virtual machine where the file is located.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host where the file is to be copied. Do not use this parameter if the HostUser and HostPassword parameters are used. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host where the file is to be copied. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host where the file is to be copied. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-GuestCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the guest OS where the file to be copied is located.

-GuestUser <String> Specifies the user name you want to use for authenticating with the guest OS where the file to be copied is located.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest OS where the file to be copied is located.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from the VMware Tools. If a non-positive value is provided, the system waits infinitely long time.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.
-GuestToLocal	Indicates that you want to copy a file from the guest operating system of the virtual machine to a local directory.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Copy-VMGuestFile -Source c:text.txt -Destination c:temp-VM VM -GuestToLocal -GuestUser user -GuestPassword pass2

Copies the text.txt file from the guest OS system to the local Temp directory.

———— Example 2 ————

C:PS>$vm = Get-VM -Name VM

Get-Item "c:FolderToCopy*.*" | Copy-VMGuestFile -Destination "c:MyFolder" -VM $vm -LocalToGuest -GuestUser -GuestPassword pass2

Copies files from a local machine to a guest operating system.

REMARKS To see the examples, type: "get-help Copy-VMGuestFile -examples". For more information, type: "get-help Copy-VMGuestFile -detailed". For technical information, type: "get-help Copy-VMGuestFile -full". For online help, type: "get-help Copy-VMGuestFile -online"

CHAPTER 11

Disconnect Commands

This page contains details on **Disconnect** commands.

Disconnect-VIServer

NAME Disconnect-VIServer

SYNOPSIS This cmdlet closes the connection to a vCenter Server system.

SYNTAX Disconnect-VIServer [[-Server] <VIServer[]>] [-Force] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet closes the connection to a vCenter Server system. In PowerCLI, you can have multiple connections to a server. In order to disconnect from a server, you must close all active connections to it. By default, Disconnect-VIServer closes only the last connection to the specified server. To close all active connections to a server, use the Force parameter or run the cmdlet for each connection. When a server is disconnected, it is removed from the default servers list. For more information about default servers, see the description of Connect-VIServer.

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems you want to disconnect from.

-Force	Indicates that you want to close all active connections to the specified server and disconnect from it. If the value is $false, the cmdlet closes only the last connection to the specified server and you must run Disconnect-VIServer for each active connection to this server in order to disconnect from it.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$Server = Connect-VIServer 10.23.112.235

Disconnect-VIServer -Server $Server

Connects to a server with an IP address 10.23.112.235 and saves the returned VIServer object in the $Server variable. Then disconnects from the specified server.

———— Example 2 ————

C:PS>Disconnect-VIServer -Server $global:DefaultVIServers -Force

Closes all connections to the default servers.

———— Example 3 ————

C:PS>Disconnect-VIServer -Server * -Force

Disconnects all server connections.

REMARKS To see the examples, type: “get-help Disconnect-VIServer -examples”. For more information, type: “get-help Disconnect-VIServer -detailed”. For technical information, type: “get-help Disconnect-VIServer -full”. For online help, type: “get-help Disconnect-VIServer -online”

CHAPTER 12

Dismount Commands

This page contains details on **Dismount** commands.

Dismount-Tools

NAME Dismount-Tools

SYNOPSIS This cmdlet dismounts the VMware Tools installer CD.

SYNTAX Dismount-Tools [[-Guest] <VMGuest[]>] [<CommonParameters>]

Dismount-Tools [[-VM] <VirtualMachine[]>] [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet dismounts the VMware Tools installer CD from one or more virtual machines operating systems specified by the VM and Guest parameters. To specify a server different from the default one, use the Server parameter. The virtual machines must be powered on.

PARAMETERS

-Guest <VMGuest[]> Specifies the guest operating systems from which you want to remove the VMware Tools.

-VM <VirtualMachine[]> Specifies the virtual machines from which you want to remove the VMware Tools.

-Server <VIServer[]> Specifies the vCenter Server systems on which the search for virtual machine names passed by the VM parameter is performed. If no value is given to this parameter, the search for the virtual machine names is performed on the default server.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Dismount-Tools VM

Dismounts the VMware Tools from the VM virtual machine. The virtual machine must be powered on.

————— Example 2 —————

C:PS>Get-VMGuest VM | Dismount-Tools

Dismounts the VMware Tools from the virtual machine specified by its guest operating system. The virtual machine must be powered on.

REMARKS To see the examples, type: “get-help Dismount-Tools -examples”. For more information, type: “get-help Dismount-Tools -detailed”. For technical information, type: “get-help Dismount-Tools -full”. For online help, type: “get-help Dismount-Tools -online”

CHAPTER 13

Export Commands

This page contains details on **Export** commands.

Export-VApp

NAME Export-VApp

SYNOPSIS This cmdlet exports a vApp or a single virtual machine to the specified destination.

SYNTAX Export-VApp [[-Destination] <String>] [-VApp] <VApp[]> [-Name <String>] [-Force] [-Format <VAppStorageFormat>] [-CreateSeparateFolder] [-Description <String>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

Export-VApp [[-Destination] <String>] -VM <VirtualMachine[]> [-Name <String>] [-Force] [-Format <VAppStorageFormat>] [-CreateSeparateFolder] [-Description <String>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet exports a vApp or a single virtual machine to the specified destination. If no destination is specified, the cmdlet creates a new folder in the current working directory and exports the vApp or the virtual machine to it. The name of the new folder is the same as the name of the vApp or the virtual machine as it appears in vCenter Server.

PARAMETERS

-Destination <String> Specifies a destination path to the file system location where you want to export the vApp or the virtual machine. If the value of the Destination parameter is a folder, the vApp or the virtual machine is exported to a container folder (OVF). If the destination is a file, the vApp or the virtual machine is exported in OVA format.

-VApp <VApp[]> Specifies the vApp that you want to export.

-Name <String> Specifies a name for the exported vApp or virtual machine.

-Force	Indicates that the cmdlet overwrites the existing destination files and creates directories to complete the specified file path.

-Format <VAppStorageFormat> Specifies the file format of the specified vApp or virtual machine. The default format is OVF. The valid values are OVF and OVA.

-CreateSeparateFolder Indicates that you want to create a separate folder for each vApp or virtual machine.

-Description <String> Provides a description of the exported vApp or virtual machine.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-VM <VirtualMachine[]> Specifies the virtual machine that you want to export.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VApp -Name "MyVApp*" | Export-VApp -Destination "C:vapps"

Retrieves all vApps whose names start with "MyVApp" and exports them to the specified path.

————— Example 2 —————

C:PS>$myVApp = Get-VApp -Name "MyVApp1" Export-VApp -Destination "C:NewFolder" -VApp $myVApp -Name "EMail_vApp" -Force

Exports the vApp in the $myVApp variable to the specified location and assigns a name to the folder.

————— Example 3 —————

C:PS>$myVApp = Get-VApp -Name "MyVApp1" Export-VApp -vApp $myVApp -Destination "C:vappsVapp" -Force -CreateSeparateFolder:$false

Exports the vApp in the $myVApp variable to the specified location without creating a separate folder for each virtual appliance.

————— Example 4 —————

C:PS>$myVApp = Get-VApp -Name "MyVApp1" Export-VApp -vApp $myVApp -Destination "C:vappsmyVapp" -Format Ova

Exports a vApp in OVA format.

————— Example 5 —————

C:PS>Get-VM -Name MyVM* | Export-VApp -Destination "C:MyVMs"

Retrieves all virtual machines whose names start with "MyVM" and exports them to the specified path.

————— Example 6 —————

C:PS>$myVM = New-VM -Name MyVM1 -VMHost MyVMHost1 Export-VApp -Destination “C:MyVMs” -VM $myVM -Format Ova

Creates a new virtual machine and exports it in OVA format.

———— Example 7 ————

C:PS>$myVM = New-VM -Name “MyVM1” -VMHost MyVMHost1 Get-VM -Name MyVM | Export-VApp -Destination “C:MyVMs” Export-VApp -Destination “C:MyVMs” -VM $myVM -Force

Exports a virtual machine to the same path twice. The second time forces an override of the previously exported files.

REMARKS To see the examples, type: “get-help Export-VApp -examples”. For more information, type: “get-help Export-VApp -detailed”. For technical information, type: “get-help Export-VApp -full”. For online help, type: “get-help Export-VApp -online”

Export-VDPortGroup

NAME Export-VDPortGroup

SYNOPSIS This cmdlet exports the configuration of a specified distributed port group to a specified .zip file.

SYNTAX Export-VDPortGroup [-VDPortGroup] <VDPortgroup[]> [-Description <String>] [-Destination <String>] [-Force] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet exports the configuration of a specified distributed port group to a specified .zip file. You can export only vSphere distributed port groups.

Note: This cmdlet is supported only on vSphere 5.1 or later.

PARAMETERS

-VDPortGroup <VDPortgroup[]> Specifies the distributed port group whose configuration you want to export.

-Description <String> Specifies a description for the exported distributed port group configuration.

-Destination <String> Specifies an absolute or a relative file path to the location where you want to export the configuration of the distributed port group.

-Force	Indicates that if the specified destination file already exists, the existing file will be overwritten. Any directories required to complete the specified file path will also be created.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDPortGroup -Name ‘MyVDPortGroup’ | Export-VDPortGroup -Destination ‘C:MyVDSwitchesBackupMyVDPortGroup_21122012.zip’

Exports the configuration of the specified port group to the specified file.

———— Example 2 ————

C:PS>$myPortGroup = Get-VDPortGroup -Name 'MyVDPortGroup' Export-VDPortGroup -VDPortGroup $myPortGroup -Destination 'C:MyVDSwitchesBackupMyVDPortGroupBackup.zip' -Force

Exports the configuration of the specified port group to the specified file. If the MyVDSwitchesBackup directory does not exist, it is created. If the MyVDPortGroupBackup.zip file already exists in the specified location, it is overwritten.

REMARKS To see the examples, type: "get-help Export-VDPortGroup -examples". For more information, type: "get-help Export-VDPortGroup -detailed". For technical information, type: "get-help Export-VDPortGroup -full". For online help, type: "get-help Export-VDPortGroup -online"

Export-VDSwitch

NAME Export-VDSwitch

SYNOPSIS This cmdlet exports the configuration of a specified vSphere distributed switch to a .zip file.

SYNTAX Export-VDSwitch [-VDSwitch] <VDSwitch[]> [-WithoutPortGroups] [-Description <String>] [-Destination <String>] [-Force] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet exports the configuration of a specified vSphere distributed switch to a .zip file.

Note: This cmdlet is supported only on vCenter Server 5.1 or later.

PARAMETERS

-VDSwitch <VDSwitch[]> Specifies the vSphere distributed switch whose configuration you want to export.

-WithoutPortGroups Indicates that the configuration of the vSphere distributed switch is exported without its port group configuration.

-Description <String> Specifies a description for the exported vSphere distributed switch configuration.

-Destination <String> Specifies an absolute or a relative file path to the location where you want to export the vSphere distributed switch configuration.

-Force Indicates that if the specified destination file already exists, the existing file is overwritten. Any directories required to complete the specified file path are also created.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

——— Example 1 ———

C:PS>Get-VDSwitch -Name 'MyVDSwitch' | Export-VDSwitch -Description "My VDSwitch configuration" -Destination "c:myVDSwitchConfig.zip"

Exports the configuration of the specified vSphere distributed switch and its port groups to the specified file.

——— Example 2 ———

C:PS>Get-VDSwitch -Name 'MyVDSwitch' | Export-VDSwitch -Description "My VDSwitch configuration" -Destination "c:myVDSwitchConfig.zip" -WithoutPortGroups -Force

Exports the configuration of the specified vSphere distributed switch and its port groups to the specified file. If the myVDSwitchConfig.zip file already exists, it is overwritten.

REMARKS To see the examples, type: “get-help Export-VDSwitch -examples”. For more information, type: “get-help Export-VDSwitch -detailed”. For technical information, type: “get-help Export-VDSwitch -full”. For online help, type: “get-help Export-VDSwitch -online”

Export-VMHostProfile

NAME Export-VMHostProfile

SYNOPSIS This cmdlet exports the specified host profile to a file.

SYNTAX Export-VMHostProfile [-FilePath] <String> [-Profile] <VMHostProfile> [-Force] [-Server <VIServer>] [<CommonParameters>]

DESCRIPTION This cmdlet exports the specified host profile to a file that is in the VMware profile format (.vpf). If the value of the Force parameter is $false and the destination file exists or the target parent directory does not exist, a terminating error is generated. If the value of the Force parameter is $true, the existing destination file is overwritten and directories are created to complete the specified file path.

PARAMETERS

-FilePath <String> Specifies the path to the file where you want to export the host profile.

-Profile <VMHostProfile> Specifies the host profile you want to export.

-Force Indicates that the cmdlet overwrites the existing destination files and creates directories to complete the specified file path.

-Server <VIServer> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$profile = (Get-VMHostProfile -Name Profile)[0]

Export-VMHostProfile -FilePath export.prf -Profile $profile -Force

Exports the selected host profile to the export.prf file.

REMARKS To see the examples, type: “get-help Export-VMHostProfile -examples”. For more information, type: “get-help Export-VMHostProfile -detailed”. For technical information, type: “get-help Export-VMHostProfile -full”. For online help, type: “get-help Export-VMHostProfile -online”

CHAPTER 14

Format Commands

This page contains details on **Format** commands.

Format-VMHostDiskPartition

NAME Format-VMHostDiskPartition

SYNOPSIS This cmdlet formats a new VMFS (Virtual Machine File System) on each of the specified host disk partitions.

SYNTAX Format-VMHostDiskPartition [-VolumeName] <String> -VMHostDiskPartition <VMHostDiskPartition[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet formats a new VMFS (Virtual Machine File System) on each of the specified host disk partitions.

PARAMETERS

-VolumeName <String> Specifies a name for the new VMFS.

-VMHostDiskPartition <VMHostDiskPartition[]> Specifies the disk partitions on which you want to format a new VMFS.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost Host | Get-VMHostDisk | Get-VMHostDiskPartition | ? {.Type -eq “Ntfs”} | Format-VMHostDiskPartition -VolumeName “NewStorage”

Formats the NTFS disk partitions of a host.

REMARKS To see the examples, type: “get-help Format-VMHostDiskPartition -examples”. For more information, type: “get-help Format-VMHostDiskPartition -detailed”. For technical information, type: “get-help Format-VMHostDiskPartition -full”. For online help, type: “get-help Format-VMHostDiskPartition -online”

CHAPTER 15

Get Commands

This page contains details on **Get** commands.

Get-AdvancedSetting

NAME Get-AdvancedSetting

SYNOPSIS This cmdlet retrieves the advanced settings for the specified entity.

SYNTAX Get-AdvancedSetting [-Entity] <VIObject[]> [[-Name] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the advanced settings for the specified entity.

PARAMETERS

-Entity <VIObject[]> Specifies the entities for which you want to retrieve the advanced settings. This parameter accepts VIServer, VirtualMachine, VMHost, DatastoreCluster, and Cluster objects.

-Name <String[]> Specifies the names of the advanced settings you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-AdvancedSetting -Entity (Get-Cluster -Name Cluster)

Retrieves the advanced settings of the cluster named Cluster.

———— Example 2 ————

C:PS>Get-AdvancedSetting -Entity (Get-Cluster -Name Cluster) -Name SettingName

Retrieves the advanced setting named SettingName of the Cluster cluster.

————— Example 3 —————

C:PS>Get-AdvancedSetting -Entity Server -Name '*smtp*'

Retrieve all smtp settings for the specified server.

REMARKS To see the examples, type: "get-help Get-AdvancedSetting -examples". For more information, type: "get-help Get-AdvancedSetting -detailed". For technical information, type: "get-help Get-AdvancedSetting -full". For online help, type: "get-help Get-AdvancedSetting -online"

Get-AlarmAction

NAME Get-AlarmAction

SYNOPSIS This cmdlet retrieves the actions of the specified alarm definitions.

SYNTAX Get-AlarmAction [[-AlarmDefinition] <AlarmDefinition[]>] [-ActionType <ActionType[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the actions of the specified alarm definitions.

PARAMETERS

-AlarmDefinition <AlarmDefinition[]> Specifies the alarm definitions for which you want to retrieve the configured actions.

-ActionType <ActionType[]> Specifies the type of the alarm actions you want to retrieve. The valid values are SendEmail, ExecuteScript, and Send SNMP.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AlarmDefinition -Name "Host processor status" | Get-AlarmAction -ActionType "ExecuteScript", "SendSNMP", "SendEmail"

Extract all PowerCLI supported alarm actions for the default alarm "Host processor status".

————— Example 2 —————

C:PS>Get-AlarmAction -AlarmDefinition "Host processor status" -ActionType "SendSNMP" -Server 'server IP'

Extract the alarm actions for the default alarm "Host processor status" by specifying the alarm by name.

REMARKS To see the examples, type: "get-help Get-AlarmAction -examples". For more information, type: "get-help Get-AlarmAction -detailed". For technical information, type: "get-help Get-AlarmAction -full". For online help, type: "get-help Get-AlarmAction -online"

Get-AlarmActionTrigger

NAME Get-AlarmActionTrigger

SYNOPSIS This cmdlet retrieves the actions that trigger the specified alarm actions.

SYNTAX Get-AlarmActionTrigger [[-AlarmAction] <AlarmAction[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the actions that trigger the specified alarm actions.

PARAMETERS

-AlarmAction <AlarmAction[]> Filters the trigger actions by the alarm actions they trigger.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AlarmAction -AlarmDefinition (Get-AlarmDefinition | select -First 1) | Get-AlarmActionTrigger

Retrieves the action triggers for the actions of the first returned alarm.

REMARKS To see the examples, type: “get-help Get-AlarmActionTrigger -examples”. For more information, type: “get-help Get-AlarmActionTrigger -detailed”. For technical information, type: “get-help Get-AlarmActionTrigger -full”. For online help, type: “get-help Get-AlarmActionTrigger -online”

Get-AlarmDefinition

NAME Get-AlarmDefinition

SYNOPSIS This cmdlet retrieves the available alarm definitions.

SYNTAX Get-AlarmDefinition [-Id <String[]>] [[-Name] <String[]>] [[-Entity] <VIObject[]>] [-Enabled <Boolean>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available alarm definitions.

PARAMETERS

-Id <String[]> Specifies the IDs of the alarms you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Name <String[]> Specifies the names of the alarms you want to retrieve.

-Entity <VIObject[]> Filters the alarm definitions by the entities to which they are defined. This parameter accepts InventoryItem, Datastore, and DatastoreCluster objects.

-Enabled [<Boolean>] Indicates that you want to retrieve only the enabled alarm definitions.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AlarmDefinition -Entity (Get-Folder -NoRecursion) -Name “Host connection and power state” -Enabled:$true

Retrieve the enabled alarms named “Host connection and power state” for the available folders.

————— Example 2 —————

C:PS>Get-AlarmDefinition -Server Server1, Server2

Retrieves the alarms for the specified servers.

REMARKS To see the examples, type: "get-help Get-AlarmDefinition -examples". For more information, type: "get-help Get-AlarmDefinition -detailed". For technical information, type: "get-help Get-AlarmDefinition -full". For online help, type: "get-help Get-AlarmDefinition -online"

Get-Annotation

NAME Get-Annotation

SYNOPSIS This cmdlet retrieves annotations.

SYNTAX Get-Annotation [[-CustomAttribute] <CustomAttribute[]>] [-Entity] <InventoryItem> [-Server <VIServer[]>] [<CommonParameters>]

Get-Annotation [-Entity] <InventoryItem> [-Name <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves annotations. An annotation is a user-defined description field of one or more vSphere objects.

PARAMETERS

-CustomAttribute <CustomAttribute[]> Specifies the custom attributes whose annotations you want to retrieve.

-Entity <InventoryItem> Specifies the entities whose annotations you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Name <String[]> Specifies the names of the annotations you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Cluster Cluster | Get-Annotation -CustomAttribute PhysicalLocation

Retrieves the annotation of the PhysicalLocation custom attribute for Cluster.

REMARKS To see the examples, type: "get-help Get-Annotation -examples". For more information, type: "get-help Get-Annotation -detailed". For technical information, type: "get-help Get-Annotation -full". For online help, type: "get-help Get-Annotation -online"

Get-CDDrive

NAME Get-CDDrive

SYNOPSIS This cmdlet retrieves virtual CD drives.

SYNTAX Get-CDDrive [-Id <String[]>] [-Server <VIServer[]>] [[-VM] <VirtualMachine[]>] [[-Template] <Template[]>] [[-Snapshot] <Snapshot[]>] [-Name <String[]>] [<CommonParameters>]

DESCRIPTION This cmdlet returns a set of virtual CD drives that belong to the virtual machines, templates, and snapshots specified by the VirtualMachine, Template, and Snapshot parameters. At least one of these parameters must be provided. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Id <String[]> Specifies the IDs of the CD drives you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Specifies the virtual machines from which you want to retrieve virtual CD drives.

-Template <Template[]> Specifies the virtual machine templates from which you want to retrieve virtual CD drives.

-Snapshot <Snapshot[]> Specifies the snapshots from which you want to retrieve virtual CD drives.

-Name <String[]> Specifies the names of the CD drives you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VM -Name VM | Get-CDDrive

Connects to a vSphere server and retrieves the CD drive of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Get-CDDrive -examples". For more information, type: "get-help Get-CDDrive -detailed". For technical information, type: "get-help Get-CDDrive -full". For online help, type: "get-help Get-CDDrive -online"

Get-Cluster

NAME Get-Cluster

SYNOPSIS This cmdlet retrieves the clusters available on a vCenter Server system.

SYNTAX Get-Cluster [[-Name] <String[]>] [-Location <VIContainer[]>] [-NoRecursion] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-Cluster [-RelatedObject] <ClusterRelatedObjectBase[]> [<CommonParameters>]

Get-Cluster [[-Name] <String[]>] [-VM <VirtualMachine[]>] [-VMHost <VMHost[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-Cluster [-Server <VIServer[]>] -Id <String[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the clusters available on a vCenter Server system. Returns a set of clusters that correspond to the filter criteria defined by the cmdlet parameters. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Name <String[]> Specifies the names of the clusters you want to retrieve.

-Location <VIContainer[]> Specifies vSphere container objects (such as folders, datacenters, and clusters) you want to search for clusters.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-Tag <Tag[]> Returns only the clusters that are associated with any of the specified tags.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RelatedObject <ClusterRelatedObjectBase[]> Specifies objects to retrieve one or more Cluster objects that are related to them. This parameter accepts OMResource objects.

-VM <VirtualMachine[]> Specifies virtual machines to filter the clusters that contain at least one of them.

-VMHost <VMHost[]> Specifies hosts to filter the clusters that contain at least one of them.

-Id <String[]> Specifies the IDs of the clusters you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Cluster -Location Folder

Get a list of the available clusters in the Folder folder.

REMARKS To see the examples, type: “get-help Get-Cluster -examples”. For more information, type: “get-help Get-Cluster -detailed”. For technical information, type: “get-help Get-Cluster -full”. For online help, type: “get-help Get-Cluster -online”

Get-ContentLibraryItem

NAME Get-ContentLibraryItem

SYNOPSIS This cmdlet retrieves catalog items from the content library.

SYNTAX Get-ContentLibraryItem [[-Name] <String[]>] [-ItemType <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-ContentLibraryItem -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves catalog items from the content library. Returns a set of catalog items that correspond to the filter criteria defined by the cmdlet parameters.

PARAMETERS

-Name <String[]> Specifies the names of the catalog items you want to retrieve.

-ItemType <String[]> Filters the catalog items by type.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the catalog items you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-ContentLibraryItem -Type "OVF" -Name "Windows*"

Retrieves content library templates of the specified type that match the specified name.

REMARKS To see the examples, type: "get-help Get-ContentLibraryItem -examples". For more information, type: "get-help Get-ContentLibraryItem -detailed". For technical information, type: "get-help Get-ContentLibraryItem -full". For online help, type: "get-help Get-ContentLibraryItem -online"

Get-CustomAttribute

NAME Get-CustomAttribute

SYNOPSIS This cmdlet retrieves custom attributes.

SYNTAX Get-CustomAttribute [-Id <String[]>] [[-Name] <String[]>] [[-TargetType] <CustomAttributeTargetType[]>] [-Global] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves custom attributes. A custom attribute is a user-defined description field of one or more vSphere objects.

PARAMETERS

-Id <String[]> Specifies the IDs of the custom attributes you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Name <String[]> Specifies the names of the custom attributes you want to retrieve.

-TargetType <CustomAttributeTargetType[]> Specifies a target type to filter the custom attributes by the type of objects to which they can be applied. The valid values are VirtualMachine, ResourcePool, Folder, VMHost, Cluster, Datacenter, and $null. If the value is $null, the custom attribute is global and applies to all target types.

-Global	Indicates that only global custom attributes are retrieved. A global custom attribute can be applied both to hosts and virtual machines.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-CustomAttribute -Global

Retrieves all global custom attributes.

————— Example 2 —————

C:PS>Get-CustomAttribute -TargetType “VirtualMachine”, “VMHost”

Retrieves all custom attributes of type VirtualMachine and VMHost.

————— Example 3 —————

C:PS>Get-CustomAttribute -Name “Creation*” -Global

Retrieves only global custom attributes that match the specified name pattern.

REMARKS To see the examples, type: “get-help Get-CustomAttribute -examples”. For more information, type: “get-help Get-CustomAttribute -detailed”. For technical information, type: “get-help Get-CustomAttribute -full”. For online help, type: “get-help Get-CustomAttribute -online”

Get-Datacenter

NAME Get-Datacenter

SYNOPSIS This cmdlet retrieves the datacenters available on a vCenter Server system.

SYNTAX Get-Datacenter [[-Name] <String[]>] [-Location <Folder[]>] [-NoRecursion] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-Datacenter [[-Name] <String[]>] [-VM <VirtualMachine[]>] [-Cluster <Cluster[]>] [-VMHost <VMHost[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-Datacenter [-RelatedObject] <DatacenterRelatedObjectBase[]> [<CommonParameters>]

Get-Datacenter [-Server <VIServer[]>] -Id <String[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the datacenters available on a vCenter Server system. Returns a set of datacenters that correspond to the filter criteria defined by the cmdlet parameters. By default, the cmdlet searches recursively from any provided starting point. In this case, if the location is not explicitly specified, the search includes the root folder and all other inventory items on the root folder level. If the command runs with the NoRecursion parameter set to $true, and the location is not specified, only the root folder is searched and no datacenters are returned. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Name <String[]> Specifies the names of the datacenters you want to retrieve.

-Location <Folder[]> Specifies vSphere container objects (such as folders) you want to search for datacenters.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-Tag <Tag[]> Returns only the datacenters that are associated with any of the specified tags.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Specifies virtual machines to filter the datacenters that contain at least one of them.

-Cluster <Cluster[]> Specifies clusters to filter the datacenters that contain at least one of them.

-VMHost <VMHost[]> Specifies hosts to filter the datacenters that contain at least one of them.

-RelatedObject <DatacenterRelatedObjectBase[]> Specifies objects to retrieve one or more Datacenter objects that are related to them. This parameter accepts OMResource objects.

-Id <String[]> Specifies the IDs of the datacenters you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Datacenter -Name Datacenter*

Retrieves a list of all datacenters on the server, whose names begin with “Datacenter”.

REMARKS To see the examples, type: “get-help Get-Datacenter -examples”. For more information, type: “get-help Get-Datacenter -detailed”. For technical information, type: “get-help Get-Datacenter -full”. For online help, type: “get-help Get-Datacenter -online”

Get-Datastore

NAME Get-Datastore

SYNOPSIS This cmdlet retrieves the datastores available on a vCenter Server system.

SYNTAX Get-Datastore [-Server <VIServer[]>] [[-Name] <String[]>] [-Location <VIObject[]>] [-RelatedObject <DatastoreRelatedObjectBase[]>] [-Refresh] [-Tag <Tag[]>] [<CommonParameters>]

Get-Datastore [-Server <VIServer[]>] -Id <String[]> [-Refresh] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the datastores available on a vCenter Server system. Returns a set of datastores that correspond to the filter criteria defined by the cmdlet parameters. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Name <String[]> Specifies the names of the datastores you want to retrieve.

-Location <VIObject[]> Specifies vSphere container objects that you want to search for datastores. This parameter accepts Datacenter, Folder, and DatastoreCluster objects.

-RelatedObject <DatastoreRelatedObjectBase[]> Specifies objects to retrieve one or more Datastore objects that are related to them. This parameter accepts vSphere VirtualMachine, VMHost, Datacenter, DatastoreCluster, Cluster, Folder, HardDisk, and OMResource objects, as well as vCloud Datastore objects.

-Refresh Indicates that the cmdlet first refreshes the storage system information and then retrieves the specified datastores.

-Tag <Tag[]> Returns only the datastores that are associated with any of the specified tags.

-Id <String[]> Specifies the IDs of the datastores you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost -Name VMHost1, VMHost2 | Get-Datastore

Retrieves datastores from the VMHost1 and VMHost2 hosts.

———— Example 2 ————

C:PS>Get-Datastore -Name MyDatastore* -Location MyDatacenter

Retrieves the datastores from the MyDatacenter datacenter that have names starting with MyDatastore.

———— Example 3 ————

C:PS>$vm1 = Get-VM -Name myVM1 $vm2 = Get-VM -Name myVM2 Get-Datastore -RelatedObject $vm1, $vm2

Retrieves the datastores for a specified array of virtual machines.

———— Example 4 ————

C:PS>$myVMHost = Get-VMHost -Name MyVMHost Get-Datastore -VMHost $myVMHost -Refresh

Refreshes the host storage system and retrieves its datastores.

REMARKS To see the examples, type: “get-help Get-Datastore -examples”. For more information, type: “get-help Get-Datastore -detailed”. For technical information, type: “get-help Get-Datastore -full”. For online help, type: “get-help Get-Datastore -online”

Get-DatastoreCluster

NAME Get-DatastoreCluster

SYNOPSIS This cmdlet retrieves datastore clusters.

SYNTAX Get-DatastoreCluster [-Id <String[]>] [[-Name] <String[]>] [-Location <VIContainer[]>] [-VM <VirtualMachine[]>] [-Template <Template[]>] [-Datastore <Datastore[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-DatastoreCluster [-RelatedObject] <DatastoreClusterRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves datastore clusters.

PARAMETERS

-Id <String[]> Specifies the IDs of the datastore clusters you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Name <String[]> Specifies the names of the datastore clusters you want to retrieve.

-Location <VIContainer[]> Specifies the datacenters and folders from which you want to retrieve datastore clusters.

-VM <VirtualMachine[]> Filters the datastore clusters by the virtual machines located in them.

-Template <Template[]> Filters the datastore clusters by the virtual machine templates located in them.

-Datastore <Datastore[]> Filters the datastore clusters by the datastores located in them.

-Tag <Tag[]> Returns only the datastore clusters that are associated with any of the specified tags.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RelatedObject <DatastoreClusterRelatedObjectBase[]> Specifies objects to retrieve one or more DatastoreCluster objects that are related to them. This parameter accepts OMResource objects.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-DatastoreCluster

Retrieves all datastore clusters.

————— Example 2 —————

C:PS>Get-DatastoreCluster -Name DatastoreCluster1

Retrieves a datastore cluster by name.

————— Example 3 —————

C:PS>Get-VM -Name WebServerVM | Get-DatastoreCluster

Retrieves datastore clusters through filtering by virtual machine.

REMARKS To see the examples, type: "get-help Get-DatastoreCluster -examples". For more information, type: "get-help Get-DatastoreCluster -detailed". For technical information, type: "get-help Get-DatastoreCluster -full". For online help, type: "get-help Get-DatastoreCluster -online"

Get-DrsRecommendation

NAME Get-DrsRecommendation

SYNOPSIS This cmdlet retrieves the available DRS recommendations from the provided clusters.

SYNTAX Get-DrsRecommendation [[-Cluster] <Cluster[]>] [-Refresh] [-Priority <Int32[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available DRS recommendations from the provided clusters.

PARAMETERS

-Cluster <Cluster[]> Specifies the clusters whose DRS recommendations you want to retrieve.

-Refresh Indicates that you want the cmdlet to refresh the information about the DRS recommendations before retrieving it.

-Priority <Int32[]> Specifies the priority of the DRS recommendations you want to retrieve. The valid values range from 1 to 5.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Cluster Cluster | Get-DrsRecommendation -Priority 4,5

Retrieves the DRS recommendations with priorities 4 and 5 from the Cluster cluster.

————— Example 2 —————

C:PS>Get-DrsRecommendation -Cluster Cluster -Refresh

Refreshes and retrieves information about the DRS recommendations from the Cluster cluster.

REMARKS To see the examples, type: “get-help Get-DrsRecommendation -examples”. For more information, type: “get-help Get-DrsRecommendation -detailed”. For technical information, type: “get-help Get-DrsRecommendation -full”. For online help, type: “get-help Get-DrsRecommendation -online”

Get-DrsRule

NAME Get-DrsRule

SYNOPSIS This cmdlet retrieves the list of DRS rules for the specified clusters.

SYNTAX Get-DrsRule [[-Name] <String[]>] [-Cluster] <Cluster[]> [[-VM] <VirtualMachine[]>] [-Type <ResourceSchedulingRuleType[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-DrsRule [[-Name] <String[]>] [-Cluster] <Cluster[]> [[-VM] <VirtualMachine[]>] [-VMHost <VMHost[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the list of DRS rules for the specified clusters. Each rule defines the virtual machines that can run on the same host (affinity rule) or must run on different hosts (anti-affinity).

Note: To retrieve VMHostAffinity rules, you need to state this explicitly by using the Type or the VMHost parameter. Otherwise, this cmdlet returns VMAffinity and VMAntiAffinity rules.

PARAMETERS

-Name <String[]> Specifies the name of the DRS rule you want to retrieve.

-Cluster <Cluster[]> Specifies the clusters for which you want to retrieve the DRS rules.

-VM <VirtualMachine[]> Specifies virtual machines to filter the DRS rules that reference them. Passing values to this parameter through a pipeline is deprecated and will be removed in a future release.

-Type <ResourceSchedulingRuleType[]> Specifies the type of DRS rules you want to retrieve. This parameter accepts VMAntiAffinity, VMAffinity, and VMHostAffinity values. You cannot set this parameter, when the VMHost parameter is specified.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VMHost <VMHost[]> Specifies VM hosts to filter the DRS rules that reference them. When this parameter is specified, the cmdlet returns only VMHostAffinity rules. You cannot set this parameter, when the Type parameter is specified.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$myCluster = Get-Cluster -Name “MyCluster1” Get-DrsRule -Cluster $myCluster -Name “*Rule1*“

Retrieves the DRS rules for the cluster stored in the $myCluster variable, whose names contain “Rule1”.

———— Example 2 ————

C:PS>Get-Cluster -Name ‘MyCluster1’ | Get-DrsRule

Retrieves the virtual machine affinity and anti-affinity rules for the specified cluster by pipeline.

———— Example 3 ————

C:PS>$myVm1 = Get-VM -Name ‘MyVm1’ $myCluster1 = Get-Cluster ‘MyCluster1’ Get-DrsRule -Cluster $myCluster1 -VM $myVm1

Retrieves the virtual machine affinity and anti-affinity rules for the specified virtual machine in the specified cluster.

———— Example 4 ————

C:PS>Get-Cluster ‘MyCluster1’ | Get-DrsRule -Type VMHostAffinity

Retrieves virtual machine to host affinity rules for the specified cluster by pipeline.

———— Example 5 ————

C:PS>$myVMHost1 = Get-VMHost -Name ‘MyVMHost1’ $myCluster1 = Get-Cluster -Name ‘MyCluster1’ Get-DrsRule -Cluster $myCluster1 -VMHost $myVMHost1

Retrieves virtual machine to host affinity rules for the specified host and cluster.

REMARKS To see the examples, type: “get-help Get-DrsRule -examples”. For more information, type: “get-help Get-DrsRule -detailed”. For technical information, type: “get-help Get-DrsRule -full”. For online help, type: “get-help Get-DrsRule -online”

Get-EsxCli

NAME Get-EsxCli

SYNOPSIS This cmdlet exposes the ESXCLI functionality.

SYNTAX Get-EsxCli -VMHost <VMHost[]> [-V2] [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet exposes the ESXCLI functionality.

Note: This cmdlet provides a new interface to the ESXCLI functionality. Use the -V2 parameter to switch to the new cmdlet interface. For more information, check the parameter help.

Important: Scripts that use the old cmdlet interface might not be compatible across two different versions of ESXi. The old cmdlet interface is deprecated and will be removed in a future version.

PARAMETERS

-VMHost <VMHost[]> Specifies hosts on which you want to expose the ESXCLI functionality.

-V2 If specified, the cmdlet returns an EsxCli object version 2 (V2), otherwise an EsxCli object version 1 (V1) is returned. Interface V2 supports specifying method arguments only by name. This is the recommended PowerCLI interface for interoperability with ESXCLI. Interface V1 supports specifying method arguments only by position. Scripts that use interface V1 are not guaranteed to be compatible across two different versions of ESXi. Interface V1 is deprecated.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vmHost = Get-VMHost “vmHostIp” $esxcli_v1 = Get-EsxCli -VMHost $vmHost

Retrieves a version 1 interface to ESXCLI. This interface version is deprecated and will be removed in a future release. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 2 —————

C:PS>$esxcli_v1 = Get-EsxCli

Retrieves a version 1 interface to ESXCLI using the default connection when connected directly to a single ESXi server. This interface version is deprecated and will be removed in a future release. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 3 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2

Retrieves a version 2 interface to ESXCLI by specifying a version switch parameter. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 4 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $esxcli.storage.nmp

Retrieves a list of all available applications in the specified namespace. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 5 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $esx-cli.storage.nmp.device

Retrieves a list of all available commands of the specified ESXCLI application. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 6 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli_v1 = Get-EsxCli -VMHost $vmHost $esx-cli_v1.storage.nmp.device.list()

Runs a command of an ESXCLI application by using the ESXCLI V1 interface of PowerCLI. This interface version is deprecated and will be removed in a future release. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 7 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $esx-cli.storage.nmp.device.list.Invoke()

Runs a command of an ESXCLI application by using the ESXCLI V2 interface of PowerCLI. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 8 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $arguments = $esx-cli.storage.nmp.device.set.CreateArgs() $arguments

Creates an arguments hash table for a command of an ESXCLI application and prints argument info to the console, similar to the sample output below. This example uses the ESXCLI V2 interface of PowerCLI. This example works on vCenter Server 5.0/ESXi 5.0 and later.

Name Value —- —– default Unset, ([boolean], optional) device Unset, ([string]) psp Unset, ([string], optional)

————— Example 9 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2

$arguments = $esxcli.storage.nmp.device.set.CreateArgs() $arguments.device = "mpx.vmhba1:C0:T2:L0" $ar-guments.psp = "VMW_PSP_MRU"

$esxcli.storage.nmp.device.Set.Invoke($arguments)

Creates an arguments hash table, assigns argument values and invokes a command of an ESXCLI application. This example uses the ESXCLI V2 interface of PowerCLI. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 10 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $esxcli.storage.nmp.device.Set.Invoke(@{default=$true; device="mpx.vmhba1:C0:T2:L0"})

Invokes a command of an ESXCLI application by specifying the arguments hash table in-line. This example uses the ESXCLI V2 interface of PowerCLI. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 11 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli_v1 = Get-EsxCli -VMHost $vmHost $esxcli_v1.storage.nmp.device.set($null, "mpx.vmhba1:C0:T2:L0", "VMW_PSP_MRU")

Runs a command of an ESXCLI application by using the ESXCLI V1 interface of PowerCLI. This interface version is deprecated and will be removed in a future release. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 12 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $esxcli.TypeManager.QueryMoInstances($null)

Retrieves a list of all available managed object instance descriptors. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 13 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $moTypeInfo = $esxcli.TypeManager.QueryTypeInfo("vim.EsxCLI.storage.nmp.device")

$moTypeInfo.managedTypeInfo[0].method

Gets information about the specified managed object type (vim.EsxCLI.storage.nmp.device) and its methods.

————— Example 14 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $moInstance = $esxcli.TypeManager.CreateDynamicManagedObject("ha-cli-handler-storage-nmp-device")

$moInstance.InvokeOperation("list", $null)

Creates a dynamic managed object for the specified managed object instance descriptor and invokes a method without parameters. This example works on vCenter Server 5.0/ESXi 5.0 and later.

————— Example 15 —————

C:PS>$vmHost = Get-VMHost "vmHostIp" $esxcli = Get-EsxCli -VMHost $vmHost -V2 $moInstance = $esxcli.TypeManager.CreateDynamicManagedObject("ha-cli-handler-storage-nmp-device")

$moInstance.InvokeOperation("set", @{"device" = "mpx.vmhba1:C0:T2:L0"; "psp" = "VMW_PSP_MRU"})

Creates a dynamic managed object for the specified managed object instance descriptor and invokes a method using a hash table with argument values. This example works on vCenter Server 5.0/ESXi 5.0 and later.

REMARKS To see the examples, type: "get-help Get-EsxCli -examples". For more information, type: "get-help Get-EsxCli -detailed". For technical information, type: "get-help Get-EsxCli -full". For online help, type: "get-help Get-EsxCli -online"

Get-EsxTop

NAME Get-EsxTop

SYNOPSIS This cmdlet exposes the esxtop functionality.

SYNTAX Get-EsxTop [[-CounterName] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-EsxTop [[-CounterName] <String[]>] -Counter [-Server <VIServer[]>] [<CommonParameters>]

Get-EsxTop -TopologyInfo [[-Topology] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet exposes the esxtop functionality. The default parameter set is CounterValues. The Counter parameter filters the specified statistics. To retrieve all available counters, use the CounterInfo parameter set. The properties of each counter are returned through the Fields property (an array) of the CounterInfo output object. You can also retrieve stats topologies using the TopogyInfo parameter set. This information contains either inventory data that does not change or a counter instance structure describing the relationship between different counter instances.

PARAMETERS

-CounterName <String[]> Specifies the name of the counter for which you want to retrieve information.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Counter	Indicates that you want to retrieve counters information.
-TopologyInfo	Indicates that you want to retrieve topologies of the statistics.

-Topology <String[]> Specifies the topologies for which you want to retrieve information.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-EsxTop -TopologyInfo

Retrieves the available topologies.

———— Example 2 ————

C:PS>Get-EsxTop -Counter

Retrieves the available counters.

———— Example 3 ————

C:PS>$vm = Get-VM VM $group = Get-EsxTop -CounterName SchedGroup | where {$_.VMName -eq $vm.Name} $groupIDs = $group | select -ExpandProperty GroupID $gr = Get-EsxTop -TopologyInfo -Topology SchedGroup | %{$_.Entries} | where {$groupIDs -contains $_.GroupId}

$cpuIds = @() $gr | %{$_.CpuClient} | %{$cpuIds += $_.CPUClientID}

$cpuStats = Get-EsxTop -CounterName 'VCPU' | where {$cpuIds -contains $_.VCPUID}

$cpuStats | fl *

Retrieves statistics for the virtual CPUs of the specified virtual machine.

REMARKS To see the examples, type: "get-help Get-EsxTop -examples". For more information, type: "get-help Get-EsxTop -detailed". For technical information, type: "get-help Get-EsxTop -full". For online help, type: "get-help Get-EsxTop -online"

Get-FloppyDrive

NAME Get-FloppyDrive

SYNOPSIS This cmdlet retrieves the virtual floppy drives available on a vCenter Server system.

SYNTAX Get-FloppyDrive [-Id <String[]>] [-Server <VIServer[]>] [[-VM] <VirtualMachine[]>] [[-Template] <Template[]>] [[-Snapshot] <Snapshot[]>] [-Name <String[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the virtual floppy drives available on a vCenter Server system. Returns a set of virtual floppy drives that belong to the virtual machines, templates, and snapshots specified by the VirtualMachine, Template, and Snapshot parameters. At least one of these parameters must be provided. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Id <String[]> Specifies the IDs of the floppy drives you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Specifies the virtual machines from which you want to retrieve virtual floppy drives.

-Template <Template[]> Specifies the templates from which you want to retrieve virtual CD drives.

-Snapshot <Snapshot[]> Specifies the snapshots from which you want to retrieve virtual CD drives.

-Name <String[]> Specifies the names of the floppy drives you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-FloppyDrive -VM VM

Retrieves the floppy drive of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Get-FloppyDrive -examples". For more information, type: "get-help Get-FloppyDrive -detailed". For technical information, type: "get-help Get-FloppyDrive -full". For online help, type: "get-help Get-FloppyDrive -online"

Get-Folder

NAME Get-Folder

SYNOPSIS This cmdlet retrieves the folders available on a vCenter Server system.

SYNTAX Get-Folder [-Location <VIContainer[]>] [-Type <FolderType[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [[-Name] <String[]>] [-NoRecursion] [<CommonParameters>]

Get-Folder [-RelatedObject] <FolderRelatedObjectBase[]> [<CommonParameters>]

Get-Folder [-Server <VIServer[]>] -Id <String[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the folders available on a vCenter Server system. The cmdlet returns a set of folders that correspond to the filter criteria provided by the cmdlet parameters. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Location <VIContainer[]> Specifies vSphere container objects (folders, datacenters, or clusters) you want to search for folders.

-Type <FolderType[]> Specifies the type of the folders you want to retrieve. The valid values are VM, HostAndCluster, Datastore, Network, and Datacenter.

-Tag <Tag[]> Returns only the folders that are associated with any of the specified tags.

-Server <VIServer[]> Specifies the vSphere servers on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Name <String[]> Specifies the names of the folders you want to retrieve.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-RelatedObject <FolderRelatedObjectBase[]> Specifies objects to retrieve one or more Folder objects that are related to them. This parameter accepts OMResource objects.

-Id <String[]> Specifies the IDs of the folders you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$server = Connect-VIServer -Server 10.23.112.235

Get-Folder -Server $server -Name Folder

Retrieves the folder named Folder on the server with IP address 10.23.112.235.

———— Example 2 ————

C:PS>Get-Folder -NoRecursion

Retrieves the root folder.

———— Example 3 ————

C:PS>Get-Folder -Location $folder

Gets all folders in the specified location.

———— Example 4 ————

C:PS>$folder = Get-Folder | Select -first 1

Get-Folder -ID $folder.ID

Gets a folder by ID.

———— Example 5 ————

C:PS>Get-Folder -Type Network

Gets all network folders.

REMARKS To see the examples, type: “get-help Get-Folder -examples”. For more information, type: “get-help Get-Folder -detailed”. For technical information, type: “get-help Get-Folder -full”. For online help, type: “get-help Get-Folder -online”

Get-HAPrimaryVMHost

NAME Get-HAPrimaryVMHost

SYNOPSIS On vCenter Server 5.0 and later, this cmdlet retrieves the master host of the specified HA cluster. On vCenter Server versions earlier than 5.0, this cmdlet retrieves the primary HA (High-Availability) hosts for the specified clusters.

SYNTAX Get-HAPrimaryVMHost [[-Cluster] <Cluster[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION On vCenter Server 5.0 and later, the cmdlet retrieves the master host of the specified HA cluster. On vCenter Server versions earlier than 5.0, the cmdlet retrieves the primary HA (High-Availability) hosts for the specified clusters.

PARAMETERS

-Cluster <Cluster[]> Specifies the clusters for which you want to retrieve the HA primary hosts.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Cluster | Get-HAPrimaryVMHost

Retrieves the HA primary hosts of the available cluster.

———— Example 2 ————

C:PS>Get-HAPrimaryVMHost Cluster

Retrieves the HA primary host of the cluster named Cluster.

REMARKS To see the examples, type: “get-help Get-HAPrimaryVMHost -examples”. For more information, type: “get-help Get-HAPrimaryVMHost -detailed”. For technical information, type: “get-help Get-HAPrimaryVMHost -full”. For online help, type: “get-help Get-HAPrimaryVMHost -online”

Get-HardDisk

NAME Get-HardDisk

SYNOPSIS This cmdlet retrieves the virtual hard disks available on a vCenter Server system.

SYNTAX Get-HardDisk [-Id <String[]>] [-Path <DatastoreItem[]>] [-DiskType <DiskType[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-HardDisk -RelatedObject <HardDiskRelatedObjectBase[]> [<CommonParameters>]

Get-HardDisk [-Id <String[]>] -Datastore <Datastore[]> [-DatastorePath <String[]>] [-DiskType <DiskType[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-HardDisk [-Id <String[]>] [-DiskType <DiskType[]>] [-Server <VIServer[]>] [[-VM] <VirtualMachine[]>] [[-Template] <Template[]>] [[-Snapshot] <Snapshot[]>] [-Name <String[]>] [<CommonParameters>]

DESCRIPTION This cmdlet returns the virtual hard disks available on a vCenter Server system. You can retrieve a hard disk by specifying the virtual machines, templates, or snapshots to which it belongs. If the hard disk is not attached to any virtual machines, templates, or snapshots, you can search for it in datastores or retrieve it by providing a datastore path to the file where the virtual hard disk is stored. In this case, you might not be able to derive disk type info, and the value of the DiskType property of the hard disk is Unknown.

PARAMETERS

-Id <String[]> Specifies the IDs of the hard disks you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Path <DatastoreItem[]> Specifies the file paths to the virtual hard disks you want to retrieve. The cmdlet searches recursively the specified locations.

-DiskType <DiskType[]> Specifies the type of the hard disks you want to retrieve. The valid values are rawVirtual, rawPhysical, flat, and unknown. If the hard disk is not attached to any virtual machines, templates, or snapshots, you can retrieve it by providing a datastore path to the file where the virtual hard disk is stored. In this case, you might not be able to derive disk type info, and the value of the DiskType property of the hard disk is Unknown.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RelatedObject <HardDiskRelatedObjectBase[]> Specifies objects to retrieve one or more HardDisk objects that are related to them.

-Datastore <Datastore[]> Specifies the datastores or datastore clusters you want to search for hard disks. This parameter is required when retrieving a hard disk that is attached to no virtual machines, templates, or snapshots.

-DatastorePath <String[]> Specifies datastore paths to the hard disks you want to retrieve. The paths must be in the following format: [datastore_name] <file_path>, where [datastore_name] is the name of the datastore in square brackets and <file_path> is a slash-delimited path from the root of the datastore to the virtual hard disk file. The cmdlet searches recursively the specified locations.

To learn more about the Datastore Provider, in the vSphere PowerCLI service console, type:

help about_vimdatastore

-VM <VirtualMachine[]> Specifies the virtual machines from which you want to retrieve the hard disks.

-Template <Template[]> Specifies the virtual machine templates from which you want to retrieve the hard disks.

-Snapshot <Snapshot[]> Specifies the snapshots from which you want to retrieve the hard disks.

-Name <String[]> Specifies the names of the SCSI hard disks you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-HardDisk -VM VM

Retrieves the hard disks of the virtual machine named VM.

————— Example 2 —————

C:PS>Get-HardDisk -VM $vm -DiskType flat

Retrieves the flat hard disks from the specified virtual machines.

————— Example 3 —————

C:PS>Get-HardDisk -Datastore "Storage1" -DatastorePath "[Storage1] myVM/"

Retrieves the hard disks from the specified datastore and from the specified datastore path.

REMARKS To see the examples, type: "get-help Get-HardDisk -examples". For more information, type: "get-help Get-HardDisk -detailed". For technical information, type: "get-help Get-HardDisk -full". For online help, type: "get-help Get-HardDisk -online"

Get-Inventory

NAME Get-Inventory

SYNOPSIS This cmdlet retrieves the inventory items available on a vCenter Server system.

SYNTAX Get-Inventory [-Location <VIContainer[]>] [[-Name] <String[]>] [-NoRecursion] [-Server <VIServer[]>] [<CommonParameters>]

Get-Inventory -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the inventory items available on a vCenter Server system. The cmdlet returns a set of inventory items that correspond to the filter criteria specified by the provided parameters. To specify a server different from the default one, use the -Server parameter.

PARAMETERS

-Location <VIContainer[]> Specifies vSphere container objects (such as folders, datacenters, and clusters) you want to search for inventory items.

-Name <String[]> Specifies the names of the inventory objects you want to retrieve.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the inventory objects you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Inventory -Location Datacenter -Name *Pool

Retrieves all inventory items in the Datacenter datacenter, whose names end with "Pool".

REMARKS To see the examples, type: "get-help Get-Inventory -examples". For more information, type: "get-help Get-Inventory -detailed". For technical information, type: "get-help Get-Inventory -full". For online help, type: "get-help Get-Inventory -online"

Get-IScsiHbaTarget

NAME Get-IScsiHbaTarget

SYNOPSIS This cmdlet retrieves the available iSCSI HBA targets.

SYNTAX Get-IScsiHbaTarget [[-IScsiHba] <IScsiHba[]>] [-Type <IScsiHbaTargetType[]>] [[-IPEndPoint] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available iSCSI HBA targets. The cmdlet retrieves the configured targets (send and static) on the specified iSCSI storage adapters. If IPEndPoint is specified, filters the result by <Address>:<Port>. If no IScsiHba is provided - retrieves all targets from the entire inventory.

PARAMETERS

-IScsiHba <IScsiHba[]> Specifies the iSCSI HBA whose targets you want to retrieve.

-Type <IScsiHbaTargetType[]> Specifies the type of the iSCSI HBA targets you want to retrieve. The valid values are Send and Static.

-IPEndPoint <String[]> Specifies <Address>:<Port> to filter the available iSCSI HBA targets.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-IScsiHbaTarget -Address "10.23.84.73" -Type Send

Retrieves the targets of type Send on the specified address.

REMARKS To see the examples, type: "get-help Get-IScsiHbaTarget -examples". For more information, type: "get-help Get-IScsiHbaTarget -detailed". For technical information, type: "get-help Get-IScsiHbaTarget -full". For online help, type: "get-help Get-IScsiHbaTarget -online"

Get-Log

NAME Get-Log

SYNOPSIS This cmdlet retrieves entries from vSphere logs.

SYNTAX Get-Log [-Key] <String[]> [[-VMHost] <VMHost[]>] [[-StartLineNum] <Int32>] [[-NumLines] <Int32>] [-Server <VIServer[]>] [<CommonParameters>]

Get-Log [[-VMHost] <VMHost[]>] [-Bundle] [-DestinationPath] <String> [-Server <VIServer[]>] [-RunAsync] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves entries from vSphere logs. Returns portions of the log files according to the criteria provided by the cmdlet parameters. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Key <String[]> Specifies the key identifier of the log file you want to retrieve. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve logs. If no value is given to this parameter, the command returns logs only for the default vCenter Server system.

-StartLineNum <Int32> Specifies the start line number for reading from the logs.

-NumLines <Int32> Specifies the number of the lines you want to retrieve from the logs.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Bundle Indicates whether to retrieve a diagnostic bundle of logs from vCenter Server.

-DestinationPath <String> Specifies a local file path where you want to save the log bundle.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$keys = Get-LogType

Get-Log -Key $keys[0]

Obtain the available keys. Obtains the first log file from the currently connected vCenter Server system.

————— Example 2 —————

C:PS>$vmhost = Get-VMHost Host

$keyList = Get-LogType -VMHost $vmhost

$vmhost | Get-Log -Key $keyList[0] -StartLineNum 1 -NumLines 100

Retrieve the first one hundred log lines for the specified host and key.

————— Example 3 —————

C:PS>Get-VMHost Host | Get-Log -Bundle -DestinationPath "D:VMHostBundeLog"

Retrieve a bundle log for the specified host.

REMARKS To see the examples, type: "get-help Get-Log -examples". For more information, type: "get-help Get-Log -detailed". For technical information, type: "get-help Get-Log -full". For online help, type: "get-help Get-Log -online"

Get-LogType

NAME Get-LogType

SYNOPSIS This cmdlet retrieves information about the log types available on a virtual machine host.

SYNTAX Get-LogType [[-VMHost] <VMHost[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about the log types available on a virtual machine host. If no virtual machine host is specified, the cmdlet retrieves the log types for the default vCenter Server system. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to search for log types. If no value is given to this parameter, the command searches for logs only on the default vCenter Server system.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmhost = Get-VMHost -State "Connected"

Get-Logtype -VMHost $vmhost

Gets information about the available logs on the virtual machine hosts whose state is Connected.

REMARKS To see the examples, type: "get-help Get-LogType -examples". For more information, type: "get-help Get-LogType -detailed". For technical information, type: "get-help Get-LogType -full". For online help, type: "get-help Get-LogType -online"

Get-NetworkAdapter

NAME Get-NetworkAdapter

SYNOPSIS This cmdlet retrieves the virtual network adapters available on a vCenter Server system.

SYNTAX Get-NetworkAdapter [-Id <String[]>] [-Server <VIServer[]>] [[-VM] <VirtualMachine[]>] [[-Template] <Template[]>] [[-Snapshot] <Snapshot[]>] [-Name <String[]>] [<CommonParameters>]

Get-NetworkAdapter -RelatedObject <NetworkAdapterRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the virtual network adapters available on a vCenter Server system. The cmdlet returns a set of virtual network adapters assigned to the virtual machines, templates, and snapshots specified by the VirtualMachine, Template, and Snapshot parameters. At least one of these parameters must be provided. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Id <String[]> Specifies the IDs of the network adapters you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Specifies the virtual machines from which you want to retrieve virtual network adapters.

-Template <Template[]> Specifies the templates from which you want to retrieve virtual network adapters.

-Snapshot <Snapshot[]> Specifies the snapshots from which you want to retrieve virtual network adapters.

-Name <String[]> Specifies the names of the network adapters you want to retrieve.

-RelatedObject <NetworkAdapterRelatedObjectBase[]> Specify an object to retrieve one or more network adapters that are related to the object. This parameter accepts standard and distributed port groups.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-NetworkAdapter -VM MyVM

Retrieves the network adapters added to the the MyVM virtual machine.

————— Example 2 —————

C:PS>$myVDPortgroup = Get-VDPortGroup -Name "MyVDPortGroup" $myNetworkAdapters = Get-NetworkAdapter -RelatedObject $myVDPortgroup

Retrieves all network adapters connected to the specified port group and stores them in the myNetworkAdapters variable.

REMARKS To see the examples, type: "get-help Get-NetworkAdapter -examples". For more information, type: "get-help Get-NetworkAdapter -detailed". For technical information, type: "get-help Get-NetworkAdapter -full". For online help, type: "get-help Get-NetworkAdapter -online"

Get-NicTeamingPolicy

NAME Get-NicTeamingPolicy

SYNOPSIS This cmdlet retrieves the NIC teaming policies of the specified virtual switches and virtual port groups.

SYNTAX Get-NicTeamingPolicy [-VirtualSwitch] <VirtualSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-NicTeamingPolicy [-VirtualPortGroup] <VirtualPortGroup[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the NIC teaming policies of the specified virtual switches and virtual port groups. The NIC teaming policy determines how network traffic is distributed between adapters and how traffic is reorganized in case of adapter failure.

PARAMETERS

-VirtualSwitch <VirtualSwitch[]> Specifies the virtual switches whose NIC teaming policy you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VirtualPortGroup <VirtualPortGroup[]> Specifies the port groups whose NIC teaming policy you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VirtualPortGroup -VMHost (Get-VMHost *.128) -Name Virtual* | Get-NicTeamingPolicy | fl is*

For the specified virtual port groups, retrieves the Nic teaming policy settings whose names start with "is".

————— Example 2 —————

C:PS>Get-VirtualSwitch -VMHost (Get-VMHost *.128) -Name vswitch | Get-NicTeamingPolicy

Retrieves the Nic teaming policy of the specified virtual switch.

REMARKS To see the examples, type: "get-help Get-NicTeamingPolicy -examples". For more information, type: "get-help Get-NicTeamingPolicy -detailed". For technical information, type: "get-help Get-NicTeamingPolicy -full". For online help, type: "get-help Get-NicTeamingPolicy -online"

Get-OSCustomizationNicMapping

NAME Get-OSCustomizationNicMapping

SYNOPSIS This cmdlet retrieves the configured NIC setting mappings for the specified OS customization specification.

SYNTAX Get-OSCustomizationNicMapping [-OSCustomizationSpec] <OSCustomizationSpec[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the configured NIC setting mappings for the specified OS customization specification.

PARAMETERS

-OSCustomizationSpec <OSCustomizationSpec[]> Specifies the OS customization specification for which you want to retrieve the NIC settings mapping.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$spec1 = Get-OSCustomizationSpec "test"

$spec2 = Get-OSCustomizationSpec "test_old"

Get-OSCustomizationNicMapping -OSCustomizationSpec $spec1,$spec2

Retrieves the NIC mappings of the "test" and "test_old" OS customization specifications.

REMARKS To see the examples, type: "get-help Get-OSCustomizationNicMapping -examples". For more information, type: "get-help Get-OSCustomizationNicMapping -detailed". For technical information, type: "get-help Get-OSCustomizationNicMapping -full". For online help, type: "get-help Get-OSCustomizationNicMapping -online"

Get-OSCustomizationSpec

NAME Get-OSCustomizationSpec

SYNOPSIS This cmdlet retrieves the OS customization specifications available on a vCenter Server system.

SYNTAX Get-OSCustomizationSpec [[-Server] <VIServer[]>] [[-Name] <String[]>] [-Id <String[]>] [-Type <OSCustomizationSpecType>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the OS customization specifications available on a vCenter Server system. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Name <String[]> Specifies the names of the OS customization specifications you want to retrieve.

-Id <String[]> Specifies the IDs of the OS customization specifications you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Type <OSCustomizationSpecType> Specifis the type of the OS customization specifications you want to retrieve. The valid values are Persistent and NonPersistent.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-OSCustomizationSpec “test”

Retrieves from the server the OS customization specification named ‘test’.

———— Example 2 ————

C:PS>New-VM -Name VM -VMHost Host -Template Template -OSCustomizationSpec $spec

Creates a new virtual machine from a template and configures it using a customization specification.

REMARKS To see the examples, type: “get-help Get-OSCustomizationSpec -examples”. For more information, type: “get-help Get-OSCustomizationSpec -detailed”. For technical information, type: “get-help Get-OSCustomizationSpec -full”. For online help, type: “get-help Get-OSCustomizationSpec -online”

Get-OvfConfiguration

NAME Get-OvfConfiguration

SYNOPSIS This cmdlet retrieves the OVF configuration object for the specified OVF or OVA package.

SYNTAX Get-OvfConfiguration [-Ovf] <String> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the OVF configuration object for the specified OVF or OVA package. Only user-configurable properties are returned.

PARAMETERS

-Ovf <String> Specifies the local path to the OVF or OVA package for which the user-configurable options are returned. URL paths are not supported.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$ovfPath = “myOvfTemplate.ovf” $ovfConfig = Get-OvfConfiguration -Ovf $ovfPath $ovfConfig.vami.VM1.ip0.Value = “10.23.101.2” $ovfConfig.vami.VM2.ip0.Value = “10.23.101.3” Import-VApp $ovfPath -OvfConfiguration $ovfConfig -VMHost $vmHost

Modifies a specific OVF property and passes it to the Import-VApp cmdlet.

REMARKS To see the examples, type: "get-help Get-OvfConfiguration -examples". For more information, type: "get-help Get-OvfConfiguration -detailed". For technical information, type: "get-help Get-OvfConfiguration -full". For online help, type: "get-help Get-OvfConfiguration -online"

Get-PassthroughDevice

NAME Get-PassthroughDevice

SYNOPSIS This cmdlet retrieves the pass-through devices available on the specified hosts, virtual machines, and templates.

SYNTAX Get-PassthroughDevice [-VM <VirtualMachine[]>] [-VMHost <VMHost[]>] [-Template <Template[]>] [[-Type] <PassthroughDeviceType>] [[-Name] <String[]>] [-Id <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the pass-through devices available on the specified hosts, virtual machines, and templates.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines for which you want to retrieve the pass-through devices.

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve the pass-through devices.

-Template <Template[]> Specifies the virtual machine templates for which you want to retrieve the pass-through devices.

-Type <PassthroughDeviceType> Specifies the type of the pass-through devices you want to retrieve. The valid values are SCSI and PCI. PCI is supported only on vCenter Server 4.1 and ESX 4.1 and later.

-Name <String[]> Specifies the names of the pass-through devices you want to retrieve.

-Id <String[]> Specifies the IDs of the pass-through devices you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-PassthroughDevice -VMHost Host -Type Scsi

Retrieves the SCSI passthrough devices of the Host host.

REMARKS To see the examples, type: "get-help Get-PassthroughDevice -examples". For more information, type: "get-help Get-PassthroughDevice -detailed". For technical information, type: "get-help Get-PassthroughDevice -full". For online help, type: "get-help Get-PassthroughDevice -online"

Get-PowerCLIConfiguration

NAME Get-PowerCLIConfiguration

SYNOPSIS This cmdlet retrieves the vSphere PowerCLI proxy configuration and default servers policy.

SYNTAX Get-PowerCLIConfiguration [-Scope <ConfigurationScope>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the vSphere PowerCLI proxy configuration and default servers policy.

PARAMETERS

-Scope <ConfigurationScope> Specifies a scope to filter vSphere PowerCLI settings by. The parameter accepts Session, User, and All Users values.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-PowerCLIConfiguration

Retrieves information about the vSphere PowerCLI configuration for every scope.

————— Example 2 —————

C:PS>Get-PowerCLIConfiguration -Scope User

Retrieves information about the vSphere PowerCLI configuration for the User scope.

————— Example 3 —————

C:PS>Get-PowerCLIConfiguration -Scope ([VMware.VimAutomation.ViCore.Types.V1.ConfigurationScope]::Session -bor [VMware.VimAutomation.ViCore.Types.V1.ConfigurationScope]::User)

Retrieves information about the vSphere PowerCLI configuration for the User and Session scopes.

REMARKS To see the examples, type: "get-help Get-PowerCLIConfiguration -examples". For more information, type: "get-help Get-PowerCLIConfiguration -detailed". For technical information, type: "get-help Get-PowerCLIConfiguration -full". For online help, type: "get-help Get-PowerCLIConfiguration -online"

Get-PowerCLIVersion

NAME Get-PowerCLIVersion

SYNOPSIS This cmdlet retrieves the versions of the installed PowerCLI snapins.

SYNTAX Get-PowerCLIVersion [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the versions of the installed PowerCLI snapins.

PARAMETERS

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-PowerCLIVersion

Retrieves the version of vSphere PowerCLI.

————— Example 2 —————

C:PS>Get-PowerCLIVersion | select -expand SnapinVersions

Lists the versions of additional PowerCLI snapins.

REMARKS To see the examples, type: "get-help Get-PowerCLIVersion -examples". For more information, type: "get-help Get-PowerCLIVersion -detailed". For technical information, type: "get-help Get-PowerCLIVersion -full". For online help, type: "get-help Get-PowerCLIVersion -online"

Get-ResourcePool

NAME Get-ResourcePool

SYNOPSIS This cmdlet retrieves the resource pools available on a vCenter Server system.

SYNTAX Get-ResourcePool [[-Name] <String[]>] [-Location <VIContainer[]>] [-Server <VIServer[]>] [-Tag <Tag[]>] [-NoRecursion] [<CommonParameters>]

Get-ResourcePool [[-Name] <String[]>] -VM <VirtualMachine[]> [-Server <VIServer[]>] [-Tag <Tag[]>] [<CommonParameters>]

Get-ResourcePool -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-ResourcePool -RelatedObject <ResourcePoolRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION Retrieves the resource pools available on a vCenter Server system. The cmdlet returns a set of resource pools that correspond to the filter criteria provided by the cmdlet parameters. Virtual machine hosts have a hidden resource pool named Resources, which is a parent of all resource pools of the host. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Name <String[]> Specifies the names of the resource pools you want to retrieve.

-Location <VIContainer[]> Specifies vSphere container objects (such as folders, datacenters, and clusters) you want to search for resource pools.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Tag <Tag[]> Returns only the resource pools that are associated with any of the specified tags.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-VM <VirtualMachine[]> Specifies virtual machines to filter the resource pools that contain at least one of them.

-Id <String[]> Specifies the IDs of the resource pools you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <ResourcePoolRelatedObjectBase[]> Specifies objects to retrieve one or more ResourcePool objects that are related to them. This parameter accepts ProviderVdc and OMResource objects.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$server = Connect-VIServer -Server 10.23.112.235

Get-ResourcePool -Server $server -VM VM

Retrieves information of the resource pool to which the virtual machine MS Win belongs.

REMARKS To see the examples, type: "get-help Get-ResourcePool -examples". For more information, type: "get-help Get-ResourcePool -detailed". For technical information, type: "get-help Get-ResourcePool -full". For online help, type: "get-help Get-ResourcePool -online"

Get-ScsiController

NAME Get-ScsiController

SYNOPSIS This cmdlet retrieves the virtual SCSI controllers assigned to the specified HardDisk, VirtualMachine, Template, and Snapshot objects.

SYNTAX Get-ScsiController [-Id <String[]>] [-HardDisk <HardDisk[]>] [-Server <VIServer[]>] [[-VM] <VirtualMachine[]>] [[-Template] <Template[]>] [[-Snapshot] <Snapshot[]>] [-Name <String[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the virtual SCSI controllers assigned to the specified HardDisk, VirtualMachine, Template, and Snapshot objects.

PARAMETERS

-Id <String[]> Specifies the IDs of the SCSI controllers you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-HardDisk <HardDisk[]> Filters the SCSI controllers by the hard disks they belong to.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Filters the SCSI controllers by the virtual machines they belong to.

-Template <Template[]> Filters the SCSI controllers by the virtual machine templates they belong to.

-Snapshot <Snapshot[]> Filters the SCSI controllers by the snapshots they belong to.

-Name <String[]> Specifies the names of the SCSI controllers you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VM VM1, VM2 | Get-ScsiController

Retrieves the SCSI controllers of the VM1 and VM2 virtual machines.

———— Example 2 ————

C:PS>Get-VM VM | Get-Snapshot Snapshot | Get-ScsiController

Retrieves the SCSI controllers of a virtual machine snapshot.

———— Example 3 ————

C:PS>$disk = Get-VM VM | Get-HardDisk | Select -First 2

Get-ScsiController -HardDisk $disk

Retrieves the SCSI controllers of the first two hard disks of a virtual machine.

REMARKS To see the examples, type: "get-help Get-ScsiController -examples". For more information, type: "get-help Get-ScsiController -detailed". For technical information, type: "get-help Get-ScsiController -full". For online help, type: "get-help Get-ScsiController -online"

Get-ScsiLun

NAME Get-ScsiLun

SYNOPSIS This cmdlet retrieves the SCSI devices available on the vCenter Server system.

SYNTAX Get-ScsiLun [[-CanonicalName] <String[]>] [-VmHost] <VMHost[]> [-Key <String[]>] [-LunType <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-ScsiLun -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-ScsiLun [[-CanonicalName] <String[]>] [-Hba] <Hba[]> [-Key <String[]>] [-LunType <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-ScsiLun [[-CanonicalName] <String[]>] [-Datastore] <Datastore[]> [-Key <String[]>] [-LunType <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the SCSI devices available on the vCenter Server system. Examples of SCSI logical unit objects include disks which may contain file system volumes or parts of volumes for the host or might serve as raw disks to a virtual machine. Other examples include SCSI passthrough devices that can be used by virtual machines. When retrieving ScsiLun objects by Datastore, the cmdlet returns a ScsiLun object for each host connected to the specified datastore. ScsiLun objects can be differed by their VMHost property.

PARAMETERS

-CanonicalName <String[]> Specifies the canonical name of the SCSI devices you want to retrieve. An example of a SCSI canonical name for Windows is "vmhba0:0:0:0".

-VmHost <VMHost[]> Specifies the hosts from which you want to retrieve the virtual SCSI devices.

-Key <String[]> Specifies the linkable identifiers of the SCSI devices you want to retrieve.

-LunType <String[]> Specifies the type of the SCSI devices you want to retrieve. The following types are valid:

cdrom communications disk enclosure mediaChanger opticalDevice printer processor scanner storageArrayController tape unknown worm

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the SCSI devices that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Hba <Hba[]> Specifies the storage adapters for which you want to retrieve the SCSI devices.

-Datastore <Datastore[]> Specifies the datastores for which you want to retrieve the SCSI devices. This parameter is supported only for VMFS volumes.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-ScsiLun -VMHost 10.23.123.100 -LunType disk

Retrieves the SCSI devices of "disk" type for the virtual machine host with an IP address 10.23.123.100.

————— Example 2 —————

C:PS>Get-VMHost | Get-ScsiLun -CanonicalName "naa.*"

Retrieves the SCSI devices with canonical names that starts with "naa." on the provided host.

————— Example 3 —————

C:PS>$hba = Get-VMHost | Get-VMHostHba -Type ParallelScsi

Get-ScsiLun -Hba $hba -LunType disk

Retrieves the SCSI devices of "disk" type for the specified HBA devices.

————— Example 4 —————

C:PS>Get-ScsiLun -Datastore Datastore -Key "key-vim.host.ScsiDisk-*"

For the Datastore datastore, retrieves the SCSI devices that have the specified linkable identifiers.

REMARKS To see the examples, type: "get-help Get-ScsiLun -examples". For more information, type: "get-help Get-ScsiLun -detailed". For technical information, type: "get-help Get-ScsiLun -full". For online help, type: "get-help Get-ScsiLun -online"

Get-ScsiLunPath

NAME Get-ScsiLunPath

SYNOPSIS This cmdlet retrieves the list of vmhba paths to a specified SCSI device.

SYNTAX Get-ScsiLunPath [[-Name] <String[]>] [-ScsiLun] <ScsiLun[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the list of vmhba paths to a specified SCSI device.

PARAMETERS

-Name <String[]> Specifies the name of the SCSI device whose vmhba paths you want to retrieve.

-ScsiLun <ScsiLun[]> Specifies the SCSI device whose vmhba paths you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$scsilun = Get-ScsiLun -VMHost 10.23.123.100 -LunType disk

Get-ScsiLunPath $scsilun

Retrieves the vmhba path to the specified SCSI device.

REMARKS To see the examples, type: "get-help Get-ScsiLunPath -examples". For more information, type: "get-help Get-ScsiLunPath -detailed". For technical information, type: "get-help Get-ScsiLunPath -full". For online help, type: "get-help Get-ScsiLunPath -online"

Get-SecurityPolicy

NAME Get-SecurityPolicy

SYNOPSIS This cmdlet retrieves the security policy for virtual port groups or the default port security policy for virtual switches.

SYNTAX Get-SecurityPolicy [-VirtualSwitch] <VirtualSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-SecurityPolicy [-VirtualPortGroup] <VirtualPortGroup[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the security policy for virtual port groups or the default port security policy for virtual switches.

PARAMETERS

-VirtualSwitch <VirtualSwitch[]> Specifies a virtual switch for which you want to retrieve the default port security policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VirtualPortGroup <VirtualPortGroup[]> Specifies a virtual port group for which you want to retrieve the security policy.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VirtualSwitch “MyVirtualSwitch” | Get-SecurityPolicy

Retrieves the security policy of a virtual switch named “MyVirtualSwitch”.

————— Example 2 —————

C:PS>Get-VirtualPortGroup “MyPortgroup” | Get-SecurityPolicy

Retrieves the security policy of a virtual switch port group named “MyPortgroup”.

REMARKS To see the examples, type: “get-help Get-SecurityPolicy -examples”. For more information, type: “get-help Get-SecurityPolicy -detailed”. For technical information, type: “get-help Get-SecurityPolicy -full”. For online help, type: “get-help Get-SecurityPolicy -online”

Get-Snapshot

NAME Get-Snapshot

SYNOPSIS This cmdlet retrieves the virtual machine snapshots available on a vCenter Server system.

SYNTAX Get-Snapshot [[-Name] <String[]>] [-Id <String[]>] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet returns information about the snapshots that correspond to the filter criteria provided by the Name and VM parameters. The disk size of the snapshots is retrieved only if you have the “Datastore/Browse datastore” privilege to the datastore where the shapshot is located. Otherwise, the following message is displayed: “Unable to populate snapshot size due to unsufficient permissions.”

PARAMETERS

-Name <String[]> Specifies the names of the snapshots you want to retrieve.

-Id <String[]> Specifies the IDs of the snapshots you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-VM <VirtualMachine[]> Specifies the virtual machines whose snapshots you want to retrieve. The position of this parameter is deprecated and will be changed in a future release. To avoid errors when you run existing scripts on future PowerCLI versions, specify the parameter by name.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Snapshot -VM VM -Name ‘Before ServicePack 2’

Retrieves the snapshot named “Before ServicePack2” of the VM virtual machine.

REMARKS To see the examples, type: “get-help Get-Snapshot -examples”. For more information, type: “get-help Get-Snapshot -detailed”. For technical information, type: “get-help Get-Snapshot -full”. For online help, type: “get-help Get-Snapshot -online”

Get-Stat

NAME Get-Stat

SYNOPSIS This cmdlet retrieves the statistical information available on a vCenter Server system.

SYNTAX Get-Stat [-Entity] <VIObject[]> [-Common] [-Memory] [-Cpu] [-Disk] [-Network] [-Stat <String[]>] [-Start <DateTime>] [-Finish <DateTime>] [-MaxSamples <Int32>] [-IntervalMins <Int32[]>] [-IntervalSecs <Int32[]>] [-Instance <String[]>] [-Realtime] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the statistical information available on a vCenter Server system for each provided entity. For example, if the CPU parameter is set to $true, collects the average CPU usage and the average CPU usagemhz counters as appropriate for each entity. If the Stat parameter is specified, collects all provided named stats counters. Counters are provided using a dotted notation of the form “counter group”.”counter name”.”rollup type”. For example: “cpu.usage.min”. The cmdlet uses the Start time, if provided, and the Finish time, if provided, along with the MaxSamples, if provided, to bound the data collection. If intervalSecs is provided, the closest matching available interval is used. For each statistics sample on the server, the cmdlet returns a Sample object. The Instance property of the Sample object shows the serial number of the device for which a statistics value is taken. If the Instance property is empty (”), this indicates that the statistics sample contains an average statistic value for all specified devices. If you are connected to a vCenter Server and Get-Stat is run for a host entity, the cmdlet returns only the statistics available on the vCenter Server.

PARAMETERS

-Entity <VIObject[]> Specifies the objects (such as virtual machine, virtual machine host, resource pool, and so on) whose statistics you want to retrieve.

-Common	Indicates whether the command collects common CPU, disk, memory and network statistics.

-Memory	Indicates whether the command collects common memory statistics, such as the mem usage, mem vmmemctl, mem active and mem granted counters as appropriate for each entity.
-Cpu	Indicates whether the command collects common CPU statistics, such as the average CPU usage and average CPU usagemhz counters as appropriate for each entity.
-Disk	Indicates whether the command collects common disk statistics, such as the average disk usage, average disk read and average disk write counters as appropriate for each entity.
-Network	Indicates whether the command collects common network statistics, such as the average network usage, average network transmitted and average network received counters as appropriate for each entity.

-Stat <String[]> Specifies the identifiers of the statistics you want to retrieve. Counters are provided using a dotted notation of the form "counter group"."counter name"."rollup type". For example, "cpu.usage.min".

-Start <DateTime> Specifies the beginning of the time range for which you want to collect statistics. The valid format is dd/mm/yyyy.

-Finish <DateTime> Specifies the end of the time range for which you want to collect statistics. The valid format is dd/mm/yyyy.

-MaxSamples <Int32> Specifies the maximum number of samples for each statistic.

-IntervalMins <Int32[]> Specifies one or more intervals in minutes of the statistics samples you want to retrieve. The closest available statistics interval is taken. To retrieve statistics samples for all available intervals, pass *. If the IntervalMins parameter is not specified, the samples with the best sample rate are retrieved. A best sample rate is the highest sample rate, whose relevant period contains the relevant periods for all other sample rates. A relevant period is the period that starts no earlier than the oldest sample still retained, and is a subset of a query period specified by the user.

-IntervalSecs <Int32[]> Specifies one or more intervals in seconds of the statistics samples you want to retrieve. The closest available statistics interval is taken. To retrieve statistics samples for all available intervals, pass *.

-Instance <String[]> Specifies the Instance property of the statistics you want to retrieve.

-Realtime	Indicates whether the command collects real time statistics.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Stat -Entity $VM -Start 5/5/2013 -Finish 7/10/2013 -Disk -IntervalSecs 300

Prints the disk statistics for the specified time interval for the first virtual machine, retrieved by the Get-VM cmdlet.

———— Example 2 ————

C:PS>Get-Stat -Entity $MyVMHost -Cpu -Instance 0

Retrieves the CPU statistics for the first processor of a multiprocessor host.

Note: This command can only work with a direct ESX connection.

———— Example 3 ————

C:PS>Get-VMHost -Name “MyVMHost” | Get-Stat -Network -IntervalSecs 20

Retrieves the network usage statistics for the specified host for the specified time interval.

———— Example 4 ————

C:PS>Get-VM -Name “MyVM” | Get-Stat -Stat “mem.usage.average” -Start $MyStartDateTime -Finish $MyFinishDateTime -MaxSamples 10

Retrieves the average memory usage statistics for the specified virtual machine between the specified start and finish date and time. The maximum number of retrieved samples is limited to 10.

———— Example 5 ————

C:PS>Get-VM -Name “MyVM” | Get-Stat -CPU -Memory -Realtime

Retrieves the real-time CPU and memory usage statistics for the specified virtual machine.

———— Example 6 ————

C:PS>Get-VMHost -Name “MyVMHost” | Get-Stat -Common

Retrieves the common statistics for the specified host.

———— Example 7 ————

C:PS>Get-Stat -Entity “MyVMHost” -Disk

Retrieves the disk usage statistics for the specified host.

REMARKS To see the examples, type: “get-help Get-Stat -examples”. For more information, type: “get-help Get-Stat -detailed”. For technical information, type: “get-help Get-Stat -full”. For online help, type: “get-help Get-Stat -online”

Get-StatInterval

NAME Get-StatInterval

SYNOPSIS This cmdlet retrieves the available statistics intervals and filters them using the provided parameters.

SYNTAX Get-StatInterval [[-Name] <String[]>] [[-SamplingPeriodSecs] <Int32[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available statistics intervals and filters them using the provided parameters.

PARAMETERS

-Name <String[]> Specifies the names of the statistics intervals you want to retrieve.

-SamplingPeriodSecs <Int32[]> Specifies the sampling period of the statistics intervals you want to retrieve. The sampling period is an integer that defines (in seconds) the interval of the statistics sample.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-StatInterval

Retrieves the available statistics intervals.

REMARKS To see the examples, type: "get-help Get-StatInterval -examples". For more information, type: "get-help Get-StatInterval -detailed". For technical information, type: "get-help Get-StatInterval -full". For online help, type: "get-help Get-StatInterval -online"

Get-StatType

NAME Get-StatType

SYNOPSIS This cmdlet retrieves the available statistics types for a inventory object.

SYNTAX Get-StatType [[-Name] <String[]>] [-Entity] <VIObject[]> [-Start <DateTime>] [-Finish <DateTime>] [-Interval <StatInterval[]>] [-Realtime] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available statistics types for a virtual machine, virtual machine host, cluster, or resource pool. Performance statistics types can be filtered by their names, start and finish times, and collection intervals. If the Realtime parameter is set, the Start and Finish parameters are ignored.

PARAMETERS

-Name <String[]> Specifies the names of the statistics types you want to retrieve.

-Entity <VIObject[]> Specifies clusters, virtual machine hosts, resource pools, or virtual machines, for which you want to retrieve the available statistics types.

-Start <DateTime> Specifies the beginning of the time range for which the statistics types you want to retrieve are collected. The valid format is dd/mm/yyyy. This value corresponds to the server time. When the start time is omitted, the returned statistics types start from the first available statistics type in the system.

-Finish <DateTime> Specifies the end of the time range for which the statistics types you want to retrieve are collected. The valid format is dd/mm/yyyy. This value corresponds to the server time. When the finish time is omitted, the returned result includes up to the most recent statistics type.

-Interval <StatInterval[]> Specifies the interval at which the statistics types you want to retrieve are gathered. The interval can be specified by its name or by its sampling period in seconds.

-Realtime Indicates that you want to retrieve realtime statistics type as well. If this parameter is set, the Start and Finish parameters are ignored.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-StatType -Entity VM

Retrieves the statistics types collected for the VM virtual machine.

REMARKS To see the examples, type: "get-help Get-StatType -examples". For more information, type: "get-help Get-StatType -detailed". For technical information, type: "get-help Get-StatType -full". For online help, type: "get-help Get-StatType -online"

Get-Tag

NAME Get-Tag

SYNOPSIS This cmdlet retrieves the tags available on a vCenter Server system.

SYNTAX Get-Tag [[-Name] <String[]>] [-Category <TagCategory[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-Tag -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the tags available on a vCenter Server system. This cmdlet filters tags by name and category to which tags belong.

PARAMETERS

-Name <String[]> Filters the tags by name.

-Category <TagCategory[]> Filters the tags by category.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Filters the tags by ID.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

——————— Example 1 ———————

C:PS>Get-Tag -Name “MyTag”

Returns all tags named “MyTag”.

——————— Example 2 ———————

C:PS>Get-Tag -Category “MyCategory1”, “MyCategory2” -Name “MyTag”

Returns all tags from the “MyCategory1” and “MyCategory2” categories, named “MyTag”.

REMARKS To see the examples, type: “get-help Get-Tag -examples”. For more information, type: “get-help Get-Tag -detailed”. For technical information, type: “get-help Get-Tag -full”. For online help, type: “get-help Get-Tag -online”

Get-TagAssignment

NAME Get-TagAssignment

SYNOPSIS This cmdlet retrieves the tag assignments of objects.

SYNTAX Get-TagAssignment [[-Entity] <VIObjectCore[]>] [-Category <TagCategory[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the tag assignments of objects. If the Entity parameter is specified, the cmdlet returns only the tag assignments for the corresponding items. If the Category parameter is specified, the cmdlet returns only the tag assignments of tags that belong to the specified category.

PARAMETERS

-Entity <VIObjectCore[]> Retrieves the tags associated with the specified items.

-Category <TagCategory[]> Returns the tags that belong to the specified categories.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$datastore = Get-DataStore MyDatastore Get-TagAssignment -Entity $datastore -Category MyCategory

Retrieves all tag assignments for the $datastore entity that have tags from the "MyCategory" category.

REMARKS To see the examples, type: "get-help Get-TagAssignment -examples". For more information, type: "get-help Get-TagAssignment -detailed". For technical information, type: "get-help Get-TagAssignment -full". For online help, type: "get-help Get-TagAssignment -online"

Get-TagCategory

NAME Get-TagCategory

SYNOPSIS This cmdlet retrieves the tag categories available on a vCenter Server system and filters them using the specified cmdlet parameters.

SYNTAX Get-TagCategory [[-Name] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-TagCategory -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the tag categories available on a vCenter Server system and filters them using the specified cmdlet parameters.

PARAMETERS

-Name <String[]> Filters the tag categories by name.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Filters the tag categories by ID.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-TagCategory -Name "MyTagCategory" -Server "MyServer"

Retrieves a tag category named "MyTagCategory" from a vCenter Server system.

REMARKS To see the examples, type: "get-help Get-TagCategory -examples". For more information, type: "get-help Get-TagCategory -detailed". For technical information, type: "get-help Get-TagCategory -full". For online help, type: "get-help Get-TagCategory -online"

Get-Task

NAME Get-Task

SYNOPSIS This cmdlet retrieves the tasks on a vCenter Server system.

SYNTAX Get-Task [[-Status] <TaskState>] [-Server <VIConnection[]>] [<CommonParameters>]

Get-Task -Id <String[]> [-Server <VIConnection[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the tasks on a vCenter Server system. The cmdlet retrieves information about the current or recent tasks. Use the Status parameter to filter tasks by their current status. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Status <TaskState> Specifies the status of the tasks you want to retrieve. The valid values are Error, Queued, Running, and Success. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-Server <VIConnection[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the tasks that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$serv = Connect-VIServer -Server 10.23.112.235

$task = Get-Task -Server $serv -Status Error

$task

Retrieves information on all tasks on the server with IP address 10.23.112.235, whose state is “Error”.

REMARKS To see the examples, type: “get-help Get-Task -examples”. For more information, type: “get-help Get-Task -detailed”. For technical information, type: “get-help Get-Task -full”. For online help, type: “get-help Get-Task -online”

Get-Template

NAME Get-Template

SYNOPSIS This cmdlet retrieves the virtual machine templates available on a vCenter Server system.

SYNTAX Get-Template [-Location <VIContainer[]>] [-Datastore <StorageResource[]>] [[-Name] <String[]>] [-NoRecursion] [-Server <VIServer[]>] [<CommonParameters>]

Get-Template -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the virtual machine templates available on a vCenter Server system. The cmdlet returns a set of templates that correspond to the filter criteria defined by the cmdlet parameters. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Location <VIContainer[]> Specifies the vSphere container objects (such as folders, datacenters, and clusters) you want to search for templates.

-Datastore <StorageResource[]> Filters templates by the datastores or datastore clusters that they are stored on.

-Name <String[]> Specifies the names of the virtual machine templates you want to retrieve.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the virtual machine templates you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Template -Name Web* -Location Datacenter

Retrieves all virtual machine templates in the Datacenter datacenter, whose names start with “Web”.

REMARKS To see the examples, type: “get-help Get-Template -examples”. For more information, type: “get-help Get-Template -detailed”. For technical information, type: “get-help Get-Template -full”. For online help, type: “get-help Get-Template -online”

Get-UsbDevice

NAME Get-UsbDevice

SYNOPSIS This cmdlet retrieves the USB devices available on a vCenter Server system.

SYNTAX Get-UsbDevice [-Id <String[]>] [-Server <VIServer[]>] [[-VM] <VirtualMachine[]>] [[-Template] <Template[]>] [[-Snapshot] <Snapshot[]>] [-Name <String[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the USB devices available on a vCenter Server system. The cmdlet returns a set of virtual USB devices assigned to the virtual machines, templates, and snapshots specified by the VirtualMachine, Template, and Snapshot parameters. At least one of these parameters must be provided. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Id <String[]> Specifies the IDs of the USB devices you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Specifies the virtual machines whose virtual USB drives you want to retrieve.

-Template <Template[]> Specifies the virtual machine templates whose virtual USB drives you want to retrieve.

-Snapshot <Snapshot[]> Specifies the virtual machine snapshots whose virtual USB you want to retrieve. Supported only on vCenter Server 4.1 and ESX 4.1 and later.

-Name <String[]> Specifies the names of the USB devices you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-UsbDevice -VM (Get-VM -Location Host)

Retrieves the USB devices attached to the virtual machines on the Host host.

REMARKS To see the examples, type: "get-help Get-UsbDevice -examples". For more information, type: "get-help Get-UsbDevice -detailed". For technical information, type: "get-help Get-UsbDevice -full". For online help, type: "get-help Get-UsbDevice -online"

Get-VApp

NAME Get-VApp

SYNOPSIS This cmdlet retrieves vApps.

SYNTAX Get-VApp [-Location <VIContainer[]>] [-Tag <Tag[]>] [[-Name] <String[]>] [-NoRecursion] [-Server <VIServer[]>] [<CommonParameters>]

Get-VApp -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves vApps.

PARAMETERS

-Location <VIContainer[]> Specifies Folder, Cluster, Datacenter, VMHost, and ResourcePool objects you want to search for vApps.

-Tag <Tag[]> Returns only the vApps that are associated with any of the specified tags.

-Name <String[]> Specifies the names of the vApps that you want to retrieve.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the vApps that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-ResourcePool MyResourcePool1 | Get-VApp -NoRecursion

Retrieves all the vApps in the ResourcePool resource pool with no recursion.

REMARKS To see the examples, type: “get-help Get-VApp -examples”. For more information, type: “get-help Get-VApp -detailed”. For technical information, type: “get-help Get-VApp -full”. For online help, type: “get-help Get-VApp -online”

Get-VDBlockedPolicy

NAME Get-VDBlockedPolicy

SYNOPSIS This cmdlet retrieves the blocking policy for distributed ports.

SYNTAX Get-VDBlockedPolicy -VDPortgroup <VDPortgroup[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDBlockedPolicy -VDSwitch <VDSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDBlockedPolicy -VDPort <VDPort[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the blocking policy for distributed ports. For distributed port group and vSphere distributed switch parameter sets, the default port policy at the distributed port group or switch level is retrieved.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies a distributed port group for which you want to retrieve the default blocking policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VDSwitch <VDSwitch[]> Specifies a vSphere distributed switch for which you want to retrieve the default blocking policy.

-VDPort <VDPort[]> Specifies the distributed ports for which you want to retrieve the blocking policy.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch “MyVDSwitch” | Get-VDBlockedPolicy

Retrieves the blocking policy of a vSphere distributed switch named “MyVDSwitch”.

———— Example 2 ————

C:PS>Get-VDPortgroup “MyVDPortgroup” | Get-VDPort | Get-VDBlockedPolicy

Retrieves the blocking policies of all ports inside a distributed port group named “MyVDPortgroup”.

REMARKS To see the examples, type: “get-help Get-VDBlockedPolicy -examples”. For more information, type: “get-help Get-VDBlockedPolicy -detailed”. For technical information, type: “get-help Get-VDBlockedPolicy -full”. For online help, type: “get-help Get-VDBlockedPolicy -online”

Get-VDPort

NAME Get-VDPort

SYNOPSIS This cmdlet retrieves virtual distributed ports.

SYNTAX Get-VDPort [-VDPortgroup <VDPortgroup[]>] [-VDSwitch <VDSwitch[]>] [[-Key] <String[]>] [-ActiveOnly] [-ConnectedOnly] [-Uplink] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves virtual distributed ports. At least one of the VDSwitch or VDPortgroup parameters must be specified.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies the distributed virtual port group whose ports you want to retrieve.

-VDSwitch <VDSwitch[]> Specifies the vSphere distributed switch whose ports you want to retrieve.

-Key <String[]> Specifies the key of the port which you want to retrieve.

-ActiveOnly	If set, only the active ports are returned.
-ConnectedOnly	If set, only the connected ports are returned.
-Uplink	If set, only uplink ports are returned. If not set, both uplink and non-uplink ports are returned. This parameter, like every SwitchParameter, can also be set to false (-Uplink:$false), in which case only non-uplink ports are returned.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDPortGroup "MyVDPortgroup" | Get-VDPort -Key "MyPortgroupKey"

Retrieves a virtual distributed port assigned with a key named "MyPortgroupKey" from a virtual distributed port group named "MyVDPortgroup".

———— Example 2 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDPort -Uplink

Retrieves all uplink virtual distributed ports of a vSphere distributed switch named "MyVDSwitch".

———— Example 3 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDPort -ConnectedOnly

Retrieves all connected virtual distributed ports of a vSphere distributed switch named "MyVDSwitch".

REMARKS To see the examples, type: "get-help Get-VDPort -examples". For more information, type: "get-help Get-VDPort -detailed". For technical information, type: "get-help Get-VDPort -full". For online help, type: "get-help Get-VDPort -online"

Get-VDPortgroup

NAME Get-VDPortgroup

SYNOPSIS This cmdlet retrieves distributed port groups.

SYNTAX Get-VDPortgroup [[-Name] <String[]>] [-NetworkAdapter <NetworkAdapter[]>] [-VDSwitch <VDSwitch[]>] [-VMHostNetworkAdapter <HostVirtualNic[]>] [-Server <VIServer[]>] [-Tag <Tag[]>] [<CommonParameters>]

Get-VDPortgroup -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDPortgroup -RelatedObject <VDPortgroupRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves distributed port groups.

PARAMETERS

-Name <String[]> Specifies the names of the distributed port groups that you want to retrieve.

-NetworkAdapter <NetworkAdapter[]> Specifies a virtual machine network adapter to retrieve the distributed port group to which the network adapter is connected.

-VDSwitch <VDSwitch[]> Specifies a vSphere distributed switch to retrieve the distributed port groups that belong to the switch.

-VMHostNetworkAdapter <HostVirtualNic[]> Specifies a host virtual network adapter to retrieve the distributed port groups to which the network adapter is connected.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Tag <Tag[]> Returns only the distributed port groups that are associated with any of the specified tags.

-Id <String[]> Specifies the IDs of the distributed port groups that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <VDPortgroupRelatedObjectBase[]> Specifies an object to retrieve one or more distributed port groups that are related to the object. This parameter accepts ExternalNetwork, OrgNetwork, NetworkPool, and OMResource objects.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDPortGroup -Name “MyVDPortGroup” -VDSwitch “MyVDSwitch”

Retrieves the distributed port group named “MyVDPortGroup” on the specified vSphere distributed switch.

————— Example 2 —————

C:PS>Get-OrgNetwork -Name “MyOrgNetwork” | Get-VDPortGroup

Retrieves the distributed port groups that are related to the specified organization network in the cloud.

————— Example 3 —————

C:PS>Get-NetworkAdapter -Name “MyVMNetworkAdapter” | Get-VDPortGroup

Retrieves the distributed port group to which the specified virtual machine network adapter is connected.

REMARKS To see the examples, type: “get-help Get-VDPortgroup -examples”. For more information, type: “get-help Get-VDPortgroup -detailed”. For technical information, type: “get-help Get-VDPortgroup -full”. For online help, type: “get-help Get-VDPortgroup -online”

Get-VDPortgroupOverridePolicy

NAME Get-VDPortgroupOverridePolicy

SYNOPSIS This cmdlet retrieves the policy for overriding port group settings at port level.

SYNTAX Get-VDPortgroupOverridePolicy [-VDPortgroup] <VDPortgroup[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the policy for overriding port group settings at port level.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies a distributed port group for which you want to retrieve the default port group overriding policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDPortgroup "MyVDPortgroup" | Get-VDPortgroupOverridePolicy

Retrieves the overriding policy settings of a distributed port group named "MyVDPortgroup".

REMARKS To see the examples, type: "get-help Get-VDPortgroupOverridePolicy -examples". For more information, type: "get-help Get-VDPortgroupOverridePolicy -detailed". For technical information, type: "get-help Get-VDPortgroupOverridePolicy -full". For online help, type: "get-help Get-VDPortgroupOverridePolicy -online"

Get-VDSecurityPolicy

NAME Get-VDSecurityPolicy

SYNOPSIS This cmdlet retrieves the security policy for distributed ports.

SYNTAX Get-VDSecurityPolicy -VDPortgroup <VDPortgroup[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDSecurityPolicy -VDSwitch <VDSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDSecurityPolicy -VDPort <VDPort[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the security policy for distributed ports. For distributed port group and vSphere distributed switch parameter sets, the default port policy at the distributed port group or switch level is retrieved.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies a distributed port group for which you want to retrieve the default security policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VDSwitch <VDSwitch[]> Specifies a vSphere distributed switch for which you want to retrieve the default security policy.

-VDPort <VDPort[]> Specifies the distributed ports for which you want to retrieve the security policy.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch “MyVDSwitch” | Get-VDSecurityPolicy

Retrieves the security policy of a vSphere distributed switch named “MyVDSwitch”.

————— Example 2 —————

C:PS>Get-VDPortgroup “MyVDPortgroup” | Get-VDPort –Key 4 | Get-VDSecurityPolicy

Retrieves the security policies of a specific port inside a distributed port group named “MyVDPortgroup”.

REMARKS To see the examples, type: “get-help Get-VDSecurityPolicy -examples”. For more information, type: “get-help Get-VDSecurityPolicy -detailed”. For technical information, type: “get-help Get-VDSecurityPolicy -full”. For online help, type: “get-help Get-VDSecurityPolicy -online”

Get-VDSwitch

NAME Get-VDSwitch

SYNOPSIS This cmdlet retrieves vSphere distributed switches.

SYNTAX Get-VDSwitch [[-Name] <String[]>] [-Location <FolderContainer[]>] [-VMHost <VMHost[]>] [-VM <VirtualMachine[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VDSwitch -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDSwitch -RelatedObject <VDSwitchRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves vSphere distributed switches.

PARAMETERS

-Name <String[]> Specifies the names of the vSphere distributed switches that you want to retrieve.

-Location <FolderContainer[]> Specifies vCenter Server container objects that you want to search for vSphere distributed switches. This parameter accepts Datacenter and Folder objects.

-VMHost <VMHost[]> Specifies hosts to retrieve vSphere distributed switches to which the hosts are added.

-VM <VirtualMachine[]> Specifies virtual machines to retrieve vSphere distributed switches they are connected to.

-Tag <Tag[]> Returns only the vSphere distributed switches that are associated with any of the specified tags.

-Server <VIServer[]> Specify the cloud servers on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-CIServer.

-Id <String[]> Specifies the IDs of the vSphere distributed switches that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <VDSwitchRelatedObjectBase[]> Specifies an object to retrieve one or more vSphere distributed switches that are related to the object. This parameter accepts NetworkPool and OMResource objects.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Datacenter -Name MyDatacenter | Get-VDSwitch

Retrieves all vSphere distributed switches in the specified datacenter.

———— Example 2 ————

C:PS>Get-VMHost -Name MyVMHost | Get-VDSwitch

Retrieves all vSphere distributed switches to which the specified host is added.

———— Example 3 ————

C:PS>Get-VM -Name MyVM | Get-VDSwitch

Retrieves all vSphere distributed switches to which the specified virtual machine is connected.

REMARKS To see the examples, type: "get-help Get-VDSwitch -examples". For more information, type: "get-help Get-VDSwitch -detailed". For technical information, type: "get-help Get-VDSwitch -full". For online help, type: "get-help Get-VDSwitch -online"

Get-VDSwitchPrivateVlan

NAME Get-VDSwitchPrivateVlan

SYNOPSIS This cmdlet retrieves the private VLAN configuration entries of a vSphere distributed switch.

SYNTAX Get-VDSwitchPrivateVlan [-VDSwitch] <VDSwitch[]> [-PrimaryVlanId <Int32[]>] [-SecondaryVlanId <Int32[]>] [-PrivateVlanType <PrivateVlanType[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the private VLAN configuration entries of a vSphere distributed switch.

PARAMETERS

-VDSwitch <VDSwitch[]> Specifies the vSphere distributed switch whose private VLAN configuration entries to retrieve.

-PrimaryVlanId <Int32[]> Specifies the primary VLAN ID of the private VLAN configuration entries that you want to retrieve.

-SecondaryVlanId <Int32[]> Specifies the secondary VLAN ID of the private VLAN configuration entries that you want to retrieve.

-PrivateVlanType <PrivateVlanType[]> Specifies the private VLAN type of the VLAN configuration entries that you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDSwitchPrivateVlan -PrivateVlanType Isolated

Retrieves all private VLAN entries of a vSphere distributed switch named "MyVDSwitch" with specified 'isolated' VLAN port type.

———— Example 2 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDSwitchPrivateVlan -PrimaryVlanId 1,3

Retrieves the private VLAN entries of a vSphere distributed switch named "MyVDSwitch" with primary VLAN identifiers 1 and 3.

REMARKS To see the examples, type: "get-help Get-VDSwitchPrivateVlan -examples". For more information, type: "get-help Get-VDSwitchPrivateVlan -detailed". For technical information, type: "get-help Get-VDSwitchPrivateVlan -full". For online help, type: "get-help Get-VDSwitchPrivateVlan -online"

Get-VDTrafficShapingPolicy

NAME Get-VDTrafficShapingPolicy

SYNOPSIS This cmdlet retrieves the traffic shaping policy for distributed ports.

SYNTAX Get-VDTrafficShapingPolicy -Direction <TrafficDirection> -VDPortgroup <VDPortgroup[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDTrafficShapingPolicy -Direction <TrafficDirection> -VDSwitch <VDSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDTrafficShapingPolicy -Direction <TrafficDirection> -VDPort <VDPort[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the traffic shaping policy for distributed ports. For distributed port group and vSphere distributed switch parameter sets, the default port policy at the distributed port group or switch level is retrieved.

PARAMETERS

-Direction <TrafficDirection> Specifies the direction of the traffic shaping policy.

-VDPortgroup <VDPortgroup[]> Specifies a distributed port group for which you want to retrieve the default traffic shaping policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VDSwitch <VDSwitch[]> Specifies a vSphere distributed switch for which you want to retrieve the default traffic shaping policy.

-VDPort <VDPort[]> Specifies the distributed ports for which you want to retrieve the traffic shaping policy.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDTrafficShapingPolicy -Direction In

Retrieves the ingress traffic shaping policy of a vSphere distributed switch named "MyVDSwitch".

———— Example 2 ————

C:PS>Get-VDPortgroup "MyVDPortgroup" | Get-VDPort | Get-VDTrafficShapingPolicy -Direction Out

Retrieves the engress traffic shaping policies of all ports inside a distributed port group named "MyVDPortgroup".

REMARKS To see the examples, type: "get-help Get-VDTrafficShapingPolicy -examples". For more information, type: "get-help Get-VDTrafficShapingPolicy -detailed". For technical information, type: "get-help Get-VDTrafficShapingPolicy -full". For online help, type: "get-help Get-VDTrafficShapingPolicy -online"

Get-VDUplinkLacpPolicy

NAME Get-VDUplinkLacpPolicy

SYNOPSIS This cmdlet retrieves the Link Aggregation Control Protocol policy for uplink ports.

SYNTAX Get-VDUplinkLacpPolicy -VDPortgroup <VDPortgroup[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDUplinkLacpPolicy -VDSwitch <VDSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDUplinkLacpPolicy -VDPort <VDPort[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the Link Aggregation Control Protocol policy for uplink ports. For uplink port group and vSphere distributed switch parameter sets, the default port policy at the uplink port group or switch level is retrieved.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies an uplink port group for which you want to retrieve the default Link Aggregation Control Protocol policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VDSwitch <VDSwitch[]> Specifies a vSphere distributed switch for which you want to retrieve the default Link Aggregation Control Protocol policy.

-VDPort <VDPort[]> Specifies the uplink port for which you want to retrieve the Link Aggregation Control Protocol policy.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDUplinkLacpPolicy

Retrieves the Link Aggregation Control Protocol policy of a vSphere distributed switch named "MyVDSwitch".

———— Example 2 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-UplinkPortgroup "MyVDPortgroup" | Get-VDUplinkLacpPolicy

Retrieves the Link Aggregation Control Protocol policy of an uplink port group named "MyUplinkPortgroup" inside a vSphere distributed switch named "MyVDSwitch".

REMARKS To see the examples, type: "get-help Get-VDUplinkLacpPolicy -examples". For more information, type: "get-help Get-VDUplinkLacpPolicy -detailed". For technical information, type: "get-help Get-VDUplinkLacpPolicy -full". For online help, type: "get-help Get-VDUplinkLacpPolicy -online"

Get-VDUplinkTeamingPolicy

NAME Get-VDUplinkTeamingPolicy

SYNOPSIS This cmdlet retrieves the uplink teaming policy for distributed ports.

SYNTAX Get-VDUplinkTeamingPolicy -VDPortgroup <VDPortgroup[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDUplinkTeamingPolicy -VDSwitch <VDSwitch[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VDUplinkTeamingPolicy -VDPort <VDPort[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the uplink teaming policy for distributed ports. For distributed port group and vSphere distributed switch parameter sets, the default port policy at the distributed port group or switch level is retrieved.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies a distributed port group for which you want to retrieve the default uplink teaming policy.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VDSwitch <VDSwitch[]> Specifies a vSphere distributed switch for which you want to retrieve the default uplink teaming policy.

-VDPort <VDPort[]> Specifies the distributed port for which you want to retrieve the uplink teaming policy.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch “MyVDSwitch” | Get-VDUplinkTeamingPolicy

Retrieves the uplink teaming policy of a vSphere distributed switch named “MyVDSwitch”.

————— Example 2 —————

C:PS>Get-VDPortgroup “MyVDPortgroup” | Get-VDPort | Get-VDUplinkTeamingPolicy

Retrieves the uplink teaming policy of all ports inside a distributed port group named “MyVDPortgroup”.

REMARKS To see the examples, type: “get-help Get-VDUplinkTeamingPolicy -examples”. For more information, type: “get-help Get-VDUplinkTeamingPolicy -detailed”. For technical information, type: “get-help Get-VDUplinkTeamingPolicy -full”. For online help, type: “get-help Get-VDUplinkTeamingPolicy -online”

Get-VIAccount

NAME Get-VIAccount

SYNOPSIS This cmdlet retrieves the accounts from the ESX/ESXi or vCenter Server.

SYNTAX Get-VIAccount [-Group] [-User] [[-Name] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VIAccount [-Group] [-User] [[-Id] <String[]>] [-Domain <String>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the accounts from the ESX/ESXi or vCenter Server. The Group and User switch parameters let you retrieve group and user accounts. By default, the cmdlet lists only user accounts. If the Domain parameter is specified, the cmdlet retrieves only the accounts on the specified AD domain. Otherwise, only local accounts are listed.

PARAMETERS

-Group	Specifies that you want to retrieve only group accounts.
-User	Specifies that you want to retrieve only user accounts.

-Name <String[]> Specifies the names of the accounts you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the accounts you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Domain <String> Specifies AD domains to search for accounts.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIAccount -Id Administrator

Retrieve accounts by Id.

————— Example 2 —————

C:PS>Get-VIAccount -Group

Retrieve all group accounts.

————— Example 3 —————

C:PS>Get-VIAccount -Id Administrator -Domain "MSDomain"

Get all VIAccounts for specified ID and Domain.

————— Example 4 —————

C:PS>Get-VIAccount -Domain "MSDomain"

Retrieve all accounts for the specified domain.

REMARKS To see the examples, type: "get-help Get-VIAccount -examples". For more information, type: "get-help Get-VIAccount -detailed". For technical information, type: "get-help Get-VIAccount -full". For online help, type: "get-help Get-VIAccount -online"

Get-VIEvent

NAME Get-VIEvent

SYNOPSIS This cmdlet retrieves information about the events on a vCenter Server system.

SYNTAX Get-VIEvent [[-Entity] <VIObject[]>] [-Start <DateTime>] [-Finish <DateTime>] [-Username <String>] [-MaxSamples <Int32>] [-Types <EventCategory[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about the events on a vCenter Server system. An event is any action in the vCenter Server system or ESX/ESXi host. You can filter retrieved events by specifying arguments for the cmdlet parameters. Filters are additive. For example, when you specify the Entity, Start, and Finish parameters, Get-VIEvent filters events both by the entity and the timestamp properties. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Entity <VIObject[]> Specifies objects (such as virtual machine, virtual machine host, resource pool, and so on) for which you want to collect events.

-Start <DateTime> Specifies the start date of the events you want to retrieve. The valid formats are dd/mm/yyyy and mm/dd/yyyy, depending on the local machine regional settings.

-Finish <DateTime> Specifies the end date of the events you want to retrieve. The valid formats are dd/mm/yyyy and mm/dd/yyyy, depending on the local machine regional settings.

-Username <String> Specifies the user that has initiated the events you want to retrieve.

-MaxSamples <Int32> Specifies the maximum number of retrieved events. When you do not filter events by time period, the maximum number of retrieved events is set to 100 by default.

Note: This parameter is ignored when the Start and Finish parameters are specified and all events from the specified period are retrieved.

-Types <EventCategory[]> Specifies the type of the events you want to collect. The valid values are Error, Info, and Warning.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIEvent -Entity MyVM1 -Username admin -Types error -MaxSamples 15

Retrieves a list of the last fifteen error events on the MyVM1 virtual machine for the user admin.

————— Example 2 —————

C:PS>Connect-VIServer -Server 10.23.113.41

$events = Get-VIEvent -MaxSamples 100

foreach ($event in $events) {if ($event.fullFormattedMessage -match "User (.*)@bd{1,3}.d{1,3}.d{1,3}.d{1,3}b logged in") {Write-Host ("User " + $matches[1] + " logged in at:" + $event.createdTime)} }

Gathers information for the users that have logged in.

REMARKS To see the examples, type: "get-help Get-VIEvent -examples". For more information, type: "get-help Get-VIEvent -detailed". For technical information, type: "get-help Get-VIEvent -full". For online help, type: "get-help Get-VIEvent -online"

Get-View

NAME Get-View

SYNOPSIS This cmdlet returns the vSphere View objects that correspond to the specified search criteria.

SYNTAX Get-View [-VIObject] <VIObject[]> [-Property <String[]>] [<CommonParameters>]

Get-View [-Server <VIServer[]>] [-SearchRoot <ManagedObjectReference>] -ViewType <Type> [-Filter <Hashtable>] [-Property <String[]>] [<CommonParameters>]

Get-View [-Server <VIServer[]>] [-Id] <ManagedObjectReference[]> [-Property <String[]>] [<CommonParameters>]

Get-View [-Property <String[]>] -RelatedObject <ViewBaseMirroredObject[]> [<CommonParameters>]

DESCRIPTION This cmdlet returns the vSphere View objects that correspond to the specified search criteria. The cmdlet retrieves the vSphere View objects specified by their IDs or by their corresponding vSphere inventory objects (VIObject). A View object ID is a <type>-<value> string. For objects with constant names such as AlarmManager and ServiceInstance, the ID format is <type> (see the examples).

PARAMETERS

-VIObject <VIObject[]> Specifies the vSphere managed object that corresponds to the View object you want to retrieve.

When you pass VIServer, Get-View returns ServiceInstance. When the retrieved View object is a ServiceInstance, you cannot convert it to a VIObject with Get-VIObjectByVIView.

-Property <String[]> Specifies the properties of the view object you want to retrieve. If no value is given, all properties are shown.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-SearchRoot <ManagedObjectReference> Specifies a starting point for the search (in the context of the inventory).

-ViewType <Type> Specifies the type of the View objects you want to retrieve.

-Filter <Hashtable> Specifies a hash of <name>-<value> pairs, where <name> represents the property value to test, and <value> represents a regex pattern the property must match. If more than one pair is present, all the patterns must match.

-Id <ManagedObjectReference[]> Specifies the IDs of the View objects you want to retrieve. A view object ID is a <type>-<value> string. For objects with constant names such as AlarmManager and ServiceInstance, the ID format is <type> (see the examples).

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <ViewBaseMirroredObject[]> Specifies view-related objects to retrieve their views.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vm = Get-View -ViewType VirtualMachine -Filter @{"Name" = "VM"}

$vmhostView = Get-View -ID $vm.Runtime.Host

$vmhostView.Summary.Runtime

Gets the VM virtual machine using a filter by name, populates the view object and retrieves the runtime information.

————— Example 2 —————

C:PS>$folder = Get-Folder Folder | Get-View

Get-View -SearchRoot $folder.MoRef -ViewType "VirtualMachine"

Gets the view objects of virtual machines by specifying the root folder - MoRef.

————— Example 3 —————

C:PS>$folder = Get-Folder VM

$folderView = Get-View $folder -Property "[VirtualMachine]ChildEntity.Network.*"

$folderView.LinkedView.ChildEntity[0].LinkedView.Network

Gets the view of a folder by specifying for the Property parameter a property path, which leads to the networks of the virtual machines in the specified folder. Retrieves the first of the returned networks.

———— Example 4 ————

C:PS>Connect-CIServer CloudServer1

Connect-VIServer VIServer1

$cloudExternalNetworkView = Get-ExternalNetwork ExternalNetwork1 | Get-CIView

Get-View -RelatedObject $cloudExternalNetworkView

Gets the view of a vSphere object related to the specified Cloud object. In this case, gets the vSphere port group for the cloud external network.

REMARKS To see the examples, type: “get-help Get-View -examples”. For more information, type: “get-help Get-View -detailed”. For technical information, type: “get-help Get-View -full”. For online help, type: “get-help Get-View -online”

Get-VIObjectByVIView

NAME Get-VIObjectByVIView

SYNOPSIS This cmdlet converts a vSphere View object to a VIObject.

SYNTAX Get-VIObjectByVIView [-VIView] <ViewBase[]> [<CommonParameters>]

Get-VIObjectByVIView [-Server <VIServer[]>] [-MORef] <ManagedObjectReference[]> [<CommonParameters>]

DESCRIPTION This cmdlet converts a vSphere View object to a VIObject using the object ID provided by the MoRef parameter. If the View object is a ServiceInstance, you cannot convert it to a VIObject.

PARAMETERS

-VIView <ViewBase[]> Specifies the vSphere .NET View object you want to convert to a vSphere PowerCLI object.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-MORef <ManagedObjectReference[]> Specifies the managed object ID, obtained from a property of another managed object or a view.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$view = Get-VM VM | Stop-VM | Get-View

$vm = Get-VIObjectByVIView $view | Start-VM

Gets the VM virtual machine, stops it, and gets its view object. Then, the command gets the virtual machine object using the Get-VIObjectByVIView cmdlet and starts the VM virtual machine.

REMARKS To see the examples, type: "get-help Get-VIObjectByVIView -examples". For more information, type: "get-help Get-VIObjectByVIView -detailed". For technical information, type: "get-help Get-VIObjectByVIView -full". For online help, type: "get-help Get-VIObjectByVIView -online"

Get-VIPermission

NAME Get-VIPermission

SYNOPSIS This cmdlet retrieves the permissions defined on the specified inventory objects.

SYNTAX Get-VIPermission [[-Entity] <VIObject[]>] [-Principal <VIAccount[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the permissions defined on the specified inventory objects. If no inventory objects are specified, the cmdlet retrieves all permissions available on the server.

PARAMETERS

-Entity <VIObject[]> Specifies the inventory items for which you want to retrieve permissions.

-Principal <VIAccount[]> Specifies the users and groups for which you want to retrieve permissions.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIPermission -Entity (Get-Datacenter) -Principal Administrator

Retrieves the permissions of the Administrator user on the provided datacenters.

REMARKS To see the examples, type: "get-help Get-VIPermission -examples". For more information, type: "get-help Get-VIPermission -detailed". For technical information, type: "get-help Get-VIPermission -full". For online help, type: "get-help Get-VIPermission -online"

Get-VIPrivilege

NAME Get-VIPrivilege

SYNOPSIS This cmdlet retrieves the privilege groups and items for the provided servers.

SYNTAX Get-VIPrivilege [-PrivilegeGroup] [-PrivilegeItem] [[-Name] <String[]>] [-Id <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VIPrivilege [[-Name] <String[]>] [-Role] <Role[]> [-Id <String[]>] [<CommonParameters>]

Get-VIPrivilege [[-Name] <String[]>] [-Group] <PrivilegeGroup[]> [-Id <String[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the privilege groups and items for the provided servers.

PARAMETERS

-PrivilegeGroup	Indicates that you want to retrieve only the privilege groups and not the privilege items in them.

-PrivilegeItem Indicates that you want to retrieve only the privilege items and not the privilege groups.

-Name <String[]> Specifies the names of the privileges you want to retrieve.

-Id <String[]> Specifies the IDs of the privileges you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Role <Role[]> Specifies the roles whose privileges you want to retrieve.

-Group <PrivilegeGroup[]> Specifies the privilege group whose items you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIPrivilege -Name "Host*"

Retrieves all privileges with names that start with "Host".

————— Example 2 —————

C:PS>Get-VIPrivilege -PrivilegeGroup

Retrieves all privilege groups.

REMARKS To see the examples, type: "get-help Get-VIPrivilege -examples". For more information, type: "get-help Get-VIPrivilege -detailed". For technical information, type: "get-help Get-VIPrivilege -full". For online help, type: "get-help Get-VIPrivilege -online"

Get-VIProperty

NAME Get-VIProperty

SYNOPSIS This cmdlet retrieves extended object properties.

SYNTAX Get-VIProperty [[-Name] <String[]>] [[-ObjectType] <String[]>] [-DeclaredOnly] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the extended properties and filters them by using the provided cmdlet parameters.

PARAMETERS

-Name <String[]> Specifies the names of the extended properties you want to retrieve.

-ObjectType <String[]> Specifies the object types for which you want to retrieve extended properties.

-DeclaredOnly Indicates that you want to retrieve only the extended properties that have been directly defined for the specified object types.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIProperty -Name “property*”

Retrieve all custom properties that match the specified name pattern.

————— Example 2 —————

C:PS>Get-VIProperty -ObjectType ‘VirtualMachine’

Retrieve all custom properties for the specified object type.

————— Example 3 —————

C:PS>Get-VIProperty -ObjectType ‘VirtualMachine’ -DeclaredOnly

Retrieve all custom properties for the specified object type that are not inherited.

REMARKS To see the examples, type: “get-help Get-VIProperty -examples”. For more information, type: “get-help Get-VIProperty -detailed”. For technical information, type: “get-help Get-VIProperty -full”. For online help, type: “get-help Get-VIProperty -online”

Get-VIRole

NAME Get-VIRole

SYNOPSIS This cmdlet retrieves all roles defined on the provided servers.

SYNTAX Get-VIRole [[-Name] <String[]>] [-Id <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves all roles defined on the provided servers.

PARAMETERS

-Name <String[]> Specifies the names of the roles you want to retrieve.

-Id <String[]> Specifies the IDs of the roles you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIRole -Server $server -Name “Admin*”

Retrieves all roles on the specified server with names that start with “Admin”.

REMARKS To see the examples, type: “get-help Get-VIRole -examples”. For more information, type: “get-help Get-VIRole -detailed”. For technical information, type: “get-help Get-VIRole -full”. For online help, type: “get-help Get-VIRole -online”

Get-VirtualPortGroup

NAME Get-VirtualPortGroup

SYNOPSIS This cmdlet retrieves the available port groups of hosts, virtual machines, and virtual switches.

SYNTAX Get-VirtualPortGroup [[-VMHost] <VMHost[]>] [-VM <VirtualMachine[]>] [-VirtualSwitch <VirtualSwitchBase[]>] [-Name <String[]>] [-Datacenter <Datacenter[]>] [-Standard] [-Distributed] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VirtualPortGroup -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VirtualPortGroup -RelatedObject <VirtualPortGroupRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available port groups of hosts, virtual machines, and virtual switches. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts whose port groups you want to retrieve. The position of this parameter is deprecated and will be changed in a future release. To avoid errors when you run existing scripts on future PowerCLI versions, specify the parameter by name.

-VM <VirtualMachine[]> Specifies the virtual machines whose port groups you want to retrieve.

-VirtualSwitch <VirtualSwitchBase[]> Specifies the virtual switches for which you want to retrieve their port groups.

-Name <String[]> Specifies the names of the port groups you want to retrieve.

-Datacenter <Datacenter[]> Filters the port groups of the virtual switches connected to hosts in the specified datacenters.

-Standard	Indicates that you want to retrieve the port groups for VirtualSwitch objects.
-Distributed	Indicates that you want to retrieve the port groups for DistributedSwitch objects. This parameter is obsolete. To retrieve distributed port groups, use the Get-VDPortgroup cmdlet instead.

-Tag <Tag[]> Returns only the virtual port groups that are associated with any of the specified tags.

Note: This parameter is compatible only with standard virtual port groups. For distributed port groups, you should use the Get-VDPortgroup cmdlet.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the port groups you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <VirtualPortGroupRelatedObjectBase[]> Specifies objects to retrieve one or more VirtualPortGroup objects that are related to them. This parameter accepts vCloud NetworkPool, vCloud ExternalNetwork, and vCloud OrgNetwork objects.

Note: In vCloud Director 5.1 environments, you cannot retrieve a distributed port group from an organization network backed by the distributed port group.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VirtualPortgroup -Name “VM Network”

Retrieves all port groups named “VM Network”.

———— Example 2 ————

C:PS>$myVMHost = Get-VMHost -Name “MyVMHost” Get-VirtualPortGroup -Name “VM Network” -VMHost $myVmHost

Retrieves the port group named “VM Network” on the specified host.

———— Example 3 ————

C:PS>$myVM = Get-VM -Name “MyVM” Get-VirtualPortGroup -VM $myVM

Retrieves all port groups to which the specified virtual machine is connected.

———— Example 4 ————

C:PS>$virtualSwitch = Get-VirtualSwitch -VMHost MyVMHost -Name vSwitch0 Get-VirtualPortGroup -VirtualSwitch $virtualSwitch

Retrieves all port groups which belong to the specified virtual switch.

REMARKS To see the examples, type: “get-help Get-VirtualPortGroup -examples”. For more information, type: “get-help Get-VirtualPortGroup -detailed”. For technical information, type: “get-help Get-VirtualPortGroup -full”. For online help, type: “get-help Get-VirtualPortGroup -online”

Get-VirtualSwitch

NAME Get-VirtualSwitch

SYNOPSIS This cmdlet retrieves the virtual switches associated with a virtual machine host or used by a virtual machine.

SYNTAX Get-VirtualSwitch [[-VMHost] <VMHost[]>] [[-VM] <VirtualMachine[]>] [-Datacenter <Datacenter[]>] [-Name <String[]>] [-Standard] [-Distributed] [-Server <VIServer[]>] [<CommonParameters>]

Get-VirtualSwitch -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VirtualSwitch -RelatedObject <VirtualSwitchRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the virtual switches associated with a virtual machine host or used by a virtual machine. At least one of the VMHost and VM parameters must be provided. The VM, VMHost, Name parameters do not accept string values through a pipeline because of collision. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts whose virtual switches you want to retrieve. The position of this parameter is deprecated and will be changed in a future release. To avoid errors when you run existing scripts on future PowerCLI versions, specify the parameter by name.

-VM <VirtualMachine[]> Specifies the virtual machines whose virtual switches you want to retrieve.

-Datacenter <Datacenter[]> Filters the virtual switches connected to hosts in the specified datacenters.

-Name <String[]> Specifies the names of the virtual switches you want to retrieve. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. The position of this parameter is deprecated and will be changed in a future release. To avoid errors when you run existing scripts on future PowerCLI versions, specify the parameter by name.

-Standard Indicates that you want to retrieve only VirtualSwitch objects.

-Distributed Indicates that you want to retrieve only DistributedSwitch objects. This parameter is obsolete. To retrieve distributed switches, use the Get-VDSwitch cmdlet instead.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the virtual switches you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <VirtualSwitchRelatedObjectBase[]> Specifies objects to retrieve one or more VirtualSwitch objects that are related to them. This parameter accepts vCloud NetworkPool objects.

Note: In vCloud Director 5.1 environments, you cannot retrieve a distributed switch from a network pool backed by the distributed switch.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VirtualSwitch -VM VM

Retrieves the virtual switch used by the virtual machine named VM.

———— Example 2 ————

C:PS>Get-Datacenter -Name “MyDatacenter” | Get-VirtualSwitch

Retrieves all virtual switches in the specified datacenter.

———— Example 3 ————

C:PS>Get-VMHost -Name “MyVMHost” | Get-VirtualSwitch

Retrieves all virtual switches on the specified host.

———— Example 4 ————

C:PS>Get-VirtualSwitch -Name “vSwitch0”

Retrieves all virtual switches named “vSwitch0”.

REMARKS To see the examples, type: “get-help Get-VirtualSwitch -examples”. For more information, type: “get-help Get-VirtualSwitch -detailed”. For technical information, type: “get-help Get-VirtualSwitch -full”. For online help, type: “get-help Get-VirtualSwitch -online”

Get-VM

NAME Get-VM

SYNOPSIS This cmdlet retrieves the virtual machines on a vCenter Server system.

SYNTAX Get-VM [[-Name] <String[]>] [-Server <VIServer[]>] [-Datastore <StorageResource[]>] [-Location <VIContainer[]>] [-Tag <Tag[]>] [-NoRecursion] [<CommonParameters>]

Get-VM [[-Name] <String[]>] [-Server <VIServer[]>] [-VirtualSwitch <VirtualSwitchBase[]>] [-Tag <Tag[]>] [<CommonParameters>]

Get-VM [-Server <VIServer[]>] -Id <String[]> [<CommonParameters>]

Get-VM -RelatedObject <VmRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the virtual machines on a vCenter Server system. Returns a set of virtual machines that correspond to the filter criteria provided by the cmdlet parameters. For virtual machines with multiple NICs and multiple IP addresses, the IPAddress property of the VMGuest object contains all IP addresses of the virtual machine. The IP at position 0 is the primary IP address.

PARAMETERS

-Name <String[]> Specifies the names of the virtual machines you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Datastore <StorageResource[]> Specifies datastores or datastore clusters to filter the virtual machines associated with them. Passing values to this parameter through a pipeline is deprecated and will be removed in a future release.

-Location <VIContainer[]> Specifies vSphere container objects you want to search for virtual machines. Supported container object types are: ResourcePool, VApp, VMHost, Folder, Cluster, Datacenter.

-Tag <Tag[]> Returns only the virtual machines that are associated with any of the specified tags.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-VirtualSwitch <VirtualSwitchBase[]> When specified, the cmdlet returns only the virtual machines that have network adapters attached to the specified switches.

-Id <String[]> Specifies the IDs of the virtual machines you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <VmRelatedObjectBase[]> Specifies objects to retrieve one or more vSphere VirtualMachine objects that are related to them. This parameter accepts vCloud CIVM and OMResource objects.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VM -Name MyVM*

Retrieves all virtual machines whose names starting with “MyVM”.

———— Example 2 ————

C:PS>$myDatastore = Get-Datastore -Name “MyDatastore” Get-VM -Datastore $myDatastore

Retrieves all virtual machines that reside on the specified datastore.

———— Example 3 ————

C:PS>$myDatacenter = Get-Datacenter -Name “MyDatacenter” Get-VM -Location $myDatacenter

Retrieves all virtual machines in the specified datacenter.

———— Example 4 ————

C:PS>$myVDSwitch = Get-VDSwitch -Name “MyVDSwitch” Get-VM -DistributedSwitch $myVDSwitch

Retrieves all virtual machines connected to the specified distributed switch.

REMARKS To see the examples, type: "get-help Get-VM -examples". For more information, type: "get-help Get-VM -detailed". For technical information, type: "get-help Get-VM -full". For online help, type: "get-help Get-VM -online"

Get-VMGuest

NAME Get-VMGuest

SYNOPSIS This cmdlet retrieves the guest operating systems of the specified virtual machines.

SYNTAX Get-VMGuest [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the guest operating systems of the specified virtual machines. To specify a server different from the default one, use the Server parameter. When Get-VMGuest is run against a virtual machine that is just starting, the properties of the returned VMGuest object are not filled at one time.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines whose guest operating systems you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMGuest -VM VM

Retrieves the guest OS of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Get-VMGuest -examples". For more information, type: "get-help Get-VMGuest -detailed". For technical information, type: "get-help Get-VMGuest -full". For online help, type: "get-help Get-VMGuest -online"

Get-VMGuestNetworkInterface

NAME Get-VMGuestNetworkInterface

SYNOPSIS This cmdlet etrieves information about the network configuration of the specified virtual machines or guests.

SYNTAX Get-VMGuestNetworkInterface [-Name <String[]>] [[-VM] <VirtualMachine[]>] [-VMGuest <VMGuest[]>] [-Server <VIServer[]>] [-ToolsWaitSecs <Int32>] [-GuestPassword <SecureString>] [-GuestUser <String>] [-GuestCredential <PSCredential>] [-HostPassword <SecureString>] [-HostUser <String>] [-HostCredential <PSCredential>] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use Invoke-VMScript instead.

This cmdlet retrieves information about the network configuration of the specified virtual machines or guests. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Execute and VirtualMachine.GuestOperations.Modify privileges.

PARAMETERS

-Name <String[]> Specifies the names of the guest network interfaces you want to retrieve.

-VM <VirtualMachine[]> Specifies the virtual machines for which you want to retrieve the network configuration.

-VMGuest <VMGuest[]> Specifies the guest OS for which you want to retrieve the network configuration.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from the VMware Tools. If a non-positive value is provided, the system waits infinitely long time.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest OS.

-GuestUser <String> Specifies the user name you want to use for authenticating with the guest OS.

-GuestCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the guest OS. Do not use this parameter if the GuestUser and GuestPassword parameters are used.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host. Do not use this parameter if the HostUser and HostPassword parameters are used. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMGuestNetworkInterface -VM $vm -GuestUser User -GuestPassword pass2

Retrieves the guest network interface of the $vm virtual machine.

REMARKS To see the examples, type: "get-help Get-VMGuestNetworkInterface -examples". For more information, type: "get-help Get-VMGuestNetworkInterface -detailed". For technical information, type: "get-help Get-VMGuestNetworkInterface -full". For online help, type: "get-help Get-VMGuestNetworkInterface -online"

Get-VMGuestRoute

NAME Get-VMGuestRoute

SYNOPSIS This cmdlet retrieves the routing configuration of the specified virtual machines or guests.

SYNTAX Get-VMGuestRoute [[-VM] <VirtualMachine[]>] [-VMGuest <VMGuest[]>] [-Server <VIServer[]>] [-ToolsWaitSecs <Int32>] [-GuestPassword <SecureString>] [-GuestUser <String>] [-GuestCredential <PSCredential>] [-HostPassword <SecureString>] [-HostUser <String>] [-HostCredential <PSCredential>] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use Invoke-VMGuestScript instead.

This cmdlet retrieves the routing configuration of the specified virtual machines or guests. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Execute and VirtualMachine.GuestOperations.Modify privileges.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines for which you want to retrieve the routing configuration.

-VMGuest <VMGuest[]> Specifies the guest operating systems for which you want to retrieve the routing configuration.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from the VMware Tools. If a non-positive value is provided, the system waits infinitely long time.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest OS.

-GuestUser <String> Specifies the user name you want to use for authenticating with the guest OS.

-GuestCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the guest OS. Do not use this parameter if the GuestUser and GuestPassword parameters are used.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host. Do not use this parameter if the HostUser and HostPassword parameters are used. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMGuestRoute -VM $vm -GuestUser User -GuestPassword pass2

Retrieves the guest route of the $vm virtual machine.

REMARKS To see the examples, type: "get-help Get-VMGuestRoute -examples". For more information, type: "get-help Get-VMGuestRoute -detailed". For technical information, type: "get-help Get-VMGuestRoute -full". For online help, type: "get-help Get-VMGuestRoute -online"

Get-VMHost

NAME Get-VMHost

SYNOPSIS This cmdlet retrieves the hosts on a vCenter Server system.

SYNTAX Get-VMHost [[-Name] <String[]>] [-NoRecursion] [-Datastore <StorageResource[]>] [-State <VMHost-State[]>] [-Location <VIContainer[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHost [[-Name] <String[]>] [-DistributedSwitch <DistributedSwitch[]>] [-Tag <Tag[]>] [-Server <VIS-erver[]>] [<CommonParameters>]

Get-VMHost [[-Name] <String[]>] [-NoRecursion] [-VM <VirtualMachine[]>] [-ResourcePool <Resource-Pool[]>] [-Datastore <StorageResource[]>] [-Location <VIContainer[]>] [-Tag <Tag[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHost [-Server <VIServer[]>] -Id <String[]> [<CommonParameters>]

Get-VMHost [-RelatedObject] <VMHostRelatedObjectBase[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the hosts on a vCenter Server system. Returns a set of hosts that correspond to the filter criteria provided by the cmdlet parameters. To specify a server different from the default one, use the Server parameter. When working directly on an ESX host, the Name property of the returned VMHost object contains either the DNS name or the IP of the ESX host, depending on which of them was specified when connecting with Connect-VIServer.

PARAMETERS

-Name <String[]> Specifies the names of the hosts you want to retrieve.

-NoRecursion Indicates that you want to disable the recursive behavior of the command.

-Datastore <StorageResource[]> Specifies the datastores or datastore clusters to which the hosts that you want to retrieve are associated. Passing values to this parameter through a pipeline is deprecated and will be removed in a future release.

-State <VMHostState[]> Specifies the state of the hosts you want to retrieve. The valid values are Connected, Disconnected, NotResponding, and Maintenance.

-Location <VIContainer[]> Specifies the vSphere container objects (such as folders, datacenters, and clusters) you want to search for hosts.

-Tag <Tag[]> Returns only the virtual machine hosts that are associated with any of the specified tags.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-DistributedSwitch <DistributedSwitch[]> Filters the available hosts by the virtual switches they are connected to.

-VM <VirtualMachine[]> Specifies virtual machines whose hosts you want to retrieve.

-ResourcePool <ResourcePool[]> Specifies resource pools associated with the hosts you want to retrieve.

-Id <String[]> Specifies the IDs of the hosts you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-RelatedObject <VMHostRelatedObjectBase[]> Specifies objects to retrieve one or more VMHost objects that are related to them. This parameter accepts OMResource objects.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost -Location MyDatacenter

Retrieves all hosts in the specified datacenter.

———— Example 2 ————

C:PS>$MyVM = Get-VM -Name MyVM Get-VMHost -VM $MyVM

Retrieves the host on which the specified virtual machine runs.

———— Example 3 ————

C:PS>$myVDSwitch = Get-VDSwitch -Name “MyVDSwitch” Get-VMHost -DistributedSwitch $myVDSwitch

Retrieves all hosts associated with the specified vSphere distributed switch.

REMARKS To see the examples, type: “get-help Get-VMHost -examples”. For more information, type: “get-help Get-VMHost -detailed”. For technical information, type: “get-help Get-VMHost -full”. For online help, type: “get-help Get-VMHost -online”

Get-VMHostAccount

NAME Get-VMHostAccount

SYNOPSIS This cmdlet retrieves the host accounts available on a vCenter Server system.

SYNTAX Get-VMHostAccount [-Group] [-User] [[-Id] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the host accounts available on a vCenter Server system. If both User and Group parameters are set to $true, in the list returned by the command, group accounts come out on top. If none of the User and Group switch parameters are provided, the cmdlet retrieves only the user accounts. If the ID parameter is set, the cmdlet filters the host accounts by their IDs. To specify a server different from the default one, use the Server parameter. Note: The specified server must be an ESX/ESXi host.

PARAMETERS

-Group	Indicates that you want to retrieve only group host accounts. Starting with ESXi 5.1, this parameter is not supported.
-User	Indicates that you want to retrieve only user host accounts.

-Id <String[]> Specifies the IDs of the host accounts you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostAccount -Group

Retrieves the group accounts for the default ESX/ESXi host. Starting with ESXi 5.1, you cannot retrieve group accounts.

———— Example 2 ————

C:PS>$myServer1 = Connect-VIServer -Server 10.23.112.235 $myUserAccount1 = New-VMHostAccount -ID MyUser1 -Password MyPass1 -UserAccount Get-VMHostAccount -Server $myServer1 -ID $myUserAccount1.Id -User

Retrieves a host account specified by an ID and an ESX/ESXi host.

———— Example 3 ————

C:PS>$myServer1 = Connect-VIServer -Server 10.23.112.235 $myGroupAccount1 = New-VMHostAccount -ID MyGroup1 -Password MyPassword1 -GroupAccount Get-VMHostAccount -Server $myServer1 -Id $myGroupAccount.Id -Group

Retrieves a group host account specified by an ID and an ESX/ESXi host. Starting with ESXi 5.1, you cannot retrieve group host accounts.

———— Example 4 ————

C:PS>$myServer1 = Connect-VIServer 10.23.112.235 Get-VMHostAccount -Server $myServer1 -User -Group

Retrieves all user and group accounts on a specified ESX/ESXi host. Starting with ESXi 5.1, you cannot retrieve group host accounts.

REMARKS To see the examples, type: “get-help Get-VMHostAccount -examples”. For more information, type: “get-help Get-VMHostAccount -detailed”. For technical information, type: “get-help Get-VMHostAccount -full”. For online help, type: “get-help Get-VMHostAccount -online”

Get-VMHostAdvancedConfiguration

NAME Get-VMHostAdvancedConfiguration

SYNOPSIS This cmdlet retrieves the advanced configuration of the hosts.

SYNTAX Get-VMHostAdvancedConfiguration [[-Name] <String[]>] [-VMHost] <VMHost[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use Get-AdvancedSetting instead.

This cmdlet retrieves the advanced configuration of the hosts. For each of the specified hosts, the cmdlet returns a hash table mapping the names of the settings to the corresponding values.

PARAMETERS

-Name <String[]> Specifies the names of the host configuration settings you want to retrieve.

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve the configuration settings.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostAdvancedConfiguration -VMHost 10.23.123.100 -Name net*tcp*

Retrieves the virtual machine host advanced configuration settings, whose names contain "net" and "tcp".

REMARKS To see the examples, type: "get-help Get-VMHostAdvancedConfiguration -examples". For more information, type: "get-help Get-VMHostAdvancedConfiguration -detailed". For technical information, type: "get-help Get-VMHostAdvancedConfiguration -full". For online help, type: "get-help Get-VMHostAdvancedConfiguration -online"

Get-VMHostAuthentication

NAME Get-VMHostAuthentication

SYNOPSIS This cmdlet retrieves authentication information for the specified hosts.

SYNTAX Get-VMHostAuthentication [[-VMHost] <VMHost[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHostAuthentication -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves authentication information for the specified hosts.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve authentication information.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the host authentication information that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmhost | Get-VMHostAuthentication

Retrieve authentication information for the specified host.

REMARKS To see the examples, type: "get-help Get-VMHostAuthentication -examples". For more information, type: "get-help Get-VMHostAuthentication -detailed". For technical information, type: "get-help Get-VMHostAuthentication -full". For online help, type: "get-help Get-VMHostAuthentication -online"

Get-VMHostAvailableTimeZone

NAME Get-VMHostAvailableTimeZone

SYNOPSIS This cmdlet retrieves the time zones available on the specified host.

SYNTAX Get-VMHostAvailableTimeZone [-VMHost] <VMHost[]> [[-Name] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the time zones available on the specified host.

PARAMETERS

-VMHost <VMHost[]> Specifies the host for which you want to retrieve the available time zones.

-Name <String[]> Specifies the names of the available time zones you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostAvailableTimeZone -Name Pacific* -VMHost 10.23.112.19

Retrieves the Pacific time zones available on the specified host.

REMARKS To see the examples, type: "get-help Get-VMHostAvailableTimeZone -examples". For more information, type: "get-help Get-VMHostAvailableTimeZone -detailed". For technical information, type: "get-help Get-VMHostAvailableTimeZone -full". For online help, type: "get-help Get-VMHostAvailableTimeZone -online"

Get-VMHostDiagnosticPartition

NAME Get-VMHostDiagnosticPartition

SYNOPSIS This cmdlet retrieves a list of the diagnostic partitions on the specified hosts.

SYNTAX Get-VMHostDiagnosticPartition [-VMHost] <VMHost[]> [-All] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves a list of the diagnostic partitions on the specified hosts. The list is ordered by the partitions preference. Local diagnostic partitions are more preferable than shared diagnostic partitions because multiple servers cannot share the same partition. The most preferred diagnostic partition is first in the list.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve diagnostic partitions.

-All Indicates that you want to retrieve all diagnostic partitions on the specified hosts. By default, only the active partitions are retrieved.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost 192.168.1.10 | Get-VMHostDiagnosticPartition -All

Retrieves all diagnostic partitions for the specified host.

REMARKS To see the examples, type: "get-help Get-VMHostDiagnosticPartition -examples". For more information, type: "get-help Get-VMHostDiagnosticPartition -detailed". For technical information, type: "get-help Get-VMHostDiagnosticPartition -full". For online help, type: "get-help Get-VMHostDiagnosticPartition -online"

Get-VMHostDisk

NAME Get-VMHostDisk

SYNOPSIS This cmdlet retrieves information about the specified SCSI LUN disk.

SYNTAX Get-VMHostDisk [[-VMHost] <VMHost[]>] [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHostDisk -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHostDisk [[-ScsiLun] <ScsiLun[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about the specified SCSI LUN disk.

PARAMETERS

-VMHost <VMHost[]> Specifies hosts to retrieve the hard disks attached to them.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the SCSI LUN disks that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-ScsiLun <ScsiLun[]> Specifies the SCSI LUN for which you want to retrieve information.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostDisk -VMHost $vmhost

Retrieves the disks of the specified host.

———— Example 2 ————

C:PS>$vmhost | Get-VMHostHba -Type FibreChannel | Get-ScsiLun | Get-VMHostDisk

Retrieves the host disks only for LUNs that are of FibreChannel type.

REMARKS To see the examples, type: "get-help Get-VMHostDisk -examples". For more information, type: "get-help Get-VMHostDisk -detailed". For technical information, type: "get-help Get-VMHostDisk -full". For online help, type: "get-help Get-VMHostDisk -online"

Get-VMHostDiskPartition

NAME Get-VMHostDiskPartition

SYNOPSIS This cmdlet retrieves the partitions of a host disk (LUN).

SYNTAX Get-VMHostDiskPartition [[-VMHostDisk] <VMHostDisk[]>] [<CommonParameters>]

Get-VMHostDiskPartition -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the partitions of a host disk (LUN).

PARAMETERS

-VMHostDisk <VMHostDisk[]> Specifies the host disk for which you want to retrieve the partitions.

-Id <String[]> Specifies the IDs of the host disk partitions that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHost Host | Get-VMHostDisk | Get-VMHostDiskPartition | ? {.Type -eq “Ntfs”} | Format-VMHostDiskPartition -VolumeName “NewStorage”

Formats the NTFS disk partitions of a host.

REMARKS To see the examples, type: “get-help Get-VMHostDiskPartition -examples”. For more information, type: “get-help Get-VMHostDiskPartition -detailed”. For technical information, type: “get-help Get-VMHostDiskPartition -full”. For online help, type: “get-help Get-VMHostDiskPartition -online”

Get-VMHostFirewallDefaultPolicy

NAME Get-VMHostFirewallDefaultPolicy

SYNOPSIS This cmdlet retrieves the firewall default policy of the specified hosts.

SYNTAX Get-VMHostFirewallDefaultPolicy [-VMHost] <VMHost[]> [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the firewall default policy of the specified hosts. The firewall policy determines whether the outgoing and incoming connections are allowed.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts whose firewall default policy you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostFirewallDefaultPolicy -VMHost 10.23.123.100

Retrieves the default firewall policy of the virtual machine host with an IP address 10.23.123.100.

REMARKS To see the examples, type: “get-help Get-VMHostFirewallDefaultPolicy -examples”. For more information, type: “get-help Get-VMHostFirewallDefaultPolicy -detailed”. For technical information, type: “get-help Get-VMHostFirewallDefaultPolicy -full”. For online help, type: “get-help Get-VMHostFirewallDefaultPolicy -online”

Get-VMHostFirewallException

NAME Get-VMHostFirewallException

SYNOPSIS This cmdlet retrieves the exceptions from the firewall policy on the specified hosts.

SYNTAX Get-VMHostFirewallException [[-Name] <String[]>] [-VMHost] <VMHost[]> [-Port <Int32[]>] [-Enabled <Boolean>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the exceptions from the firewall policy on the specified hosts. The exceptions can be filtered using the VMHost, Name, Port, and Enabled parameters.

PARAMETERS

-Name <String[]> Specifies the names of the firewall exceptions you want to retrieve.

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve firewall exceptions.

-Port <Int32[]> Specifies the number of the port for which you want to retrieve the firewall exceptions.

-Enabled [<Boolean>] Indicates whether you want to retrieve only the enabled hosts firewall exceptions.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost -Name VMHost1 | Get-VMHostFirewallException -Enabled:$true

Retrieves the enabled firewall exceptions of a host named VMHost1.

———— Example 2 ————

C:PS>Get-VMHostFirewallException “SSH Server”, “SSH Client” -VMHost $vmhost

Retrieves the firewall exceptions for the SSH server and client for the virtual machine host $vmhost.

———— Example 3 ————

C:PS>Get-VMHostFirewallException -VMHost Host -Port 21

Retrieves the firewall exceptions of a host named Host. Only firewall exceptions which have port 21 in their incoming port or outgoing ports are returned.

REMARKS To see the examples, type: “get-help Get-VMHostFirewallException -examples”. For more information, type: “get-help Get-VMHostFirewallException -detailed”. For technical information, type: “get-help Get-VMHostFirewallException -full”. For online help, type: “get-help Get-VMHostFirewallException -online”

Get-VMHostFirmware

NAME Get-VMHostFirmware

SYNOPSIS This cmdlet retrieves hosts firmware information.

SYNTAX Get-VMHostFirmware [-VMHost] <VMHost[]> [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHostFirmware [-VMHost] <VMHost[]> [-BackupConfiguration] -DestinationPath <String> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves firmware information for the hosts specified by the VMHost parameter. To specify a server different from the default one, use the Server parameter. To run this cmdlet, you need to have the "Host.Config.Firmware" permission to the ESX.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve firmware information.

-Server <VIServer[]> This parameter is required when you specify the host by name. In this case, the host with the specified name is searched on the specified servers and firmware information is retrieved from it. If a VMHost object is passed to the VMHost parameter, the Server parameter is not used.

-BackupConfiguration Indicates that you want to backup the host firmware configuration and download the bundle to the specified DestinationPath.

-DestinationPath <String> Specifies a destination path where to download the host configuration backup bundle if the BackupConfiguration parameter is set.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostFirmware -VMHost $vmhost

Retrieves the firmware information for the specified host.

————— Example 2 —————

C:PS>Get-VMHostFirmware -VMHost $vmhost -BackupConfiguration -DestinationPath C:Downloads

Backups a server configuration for the virtual machine host stored in the $vmhost variable, and downloads the configuration files into the specified folder.

REMARKS To see the examples, type: "get-help Get-VMHostFirmware -examples". For more information, type: "get-help Get-VMHostFirmware -detailed". For technical information, type: "get-help Get-VMHostFirmware -full". For online help, type: "get-help Get-VMHostFirmware -online"

Get-VMHostHardware

NAME Get-VMHostHardware

SYNOPSIS This cmdlet retrieves ESXi host hardware and firmware information.

SYNTAX Get-VMHostHardware [-VMHost <VMHost[]>] [-WaitForAllData] [-SkipCACheck] [-SkipCNCheck] [-SkipRevocationCheck] [-SkipAllSslCertificateChecks] [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHostHardware -Id <String[]> [-WaitForAllData] [-SkipCACheck] [-SkipCNCheck] [-SkipRevocationCheck] [-SkipAllSslCertificateChecks] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves hardware and firmware information for the hosts specified by the VMHost parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve hardware information. If not specified, the cmdlet retrieves hardware information for all hosts on all default connections.

-WaitForAllData If specified, forces all data for each result object to be retrieved before that object is returned. If this parameter is not specified, retrieval of some of the data in the output objects might be postponed to an arbitrary point in time between the cmdlet call and the first time the data is accessed through the corresponding property. As a result, not specifying this parameter makes the cmdlet return data faster, but different portions of the data in result objects might come from different points in time.

-SkipCACheck Indicates that when connecting through HTTPS, the client does not validate that the server certificate is signed by a trusted certification authority (CA).

Note: You should use this parameter only when the remote server can be trusted by using another mechanism, such as when the remote computer is part of a network that is physically secure and isolated.

-SkipCNCheck Indicates that the certificate common name (CN) of the server does not need to match the hostname of the server.

Note: You should use this parameter only for trusted computers.

-SkipRevocationCheck Indicates that the revocation check for server certificates is skipped.

Note: You should use this parameter only for trusted computers.

-SkipAllSslCertificateChecks Indicates that all checks for SSL server certificates are skipped.

Note: You should use this parameter only for trusted computers.

-Server <VIServer[]> This parameter is required when you specify the host by name. In this case, the host with the specified name is searched on the specified servers and hardware information is retrieved from it. If a VMHost object is passed to the VMHost parameter, the Server parameter is not used.

-Id <String[]> Filters the ESXi hosts by ID.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHost “MyVMHost” | Get-VMHostHardware

Retrieves hardware information about the “MyVMHost” host.

————— Example 2 —————

C:PS>Get-VMHostHardware -VMHost “MyVMHost” -SkipAllSslCertificateChecks

Retrieves hardware information about the “MyVMHost” host, skipping all verifications of SSL server certificates.

REMARKS To see the examples, type: “get-help Get-VMHostHardware -examples”. For more information, type: “get-help Get-VMHostHardware -detailed”. For technical information, type: “get-help Get-VMHostHardware -full”. For online help, type: “get-help Get-VMHostHardware -online”

Get-VMHostHba

NAME Get-VMHostHba

SYNOPSIS This cmdlet retrieves information about the available HBAs (Host Bus Adapter).

SYNTAX Get-VMHostHba [[-VMHost] <VMHost[]>] [[-Device] <String[]>] [-Type <HbaType[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about the available HBAs (Host Bus Adapter).

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve HBAs.

-Device <String[]> Specifies the devices of the HBA you want to retrieve.

-Type <HbaType[]> Specifies the type of the HBAs you want to retrieve. The valid values are Block, FibreChannel, iSCSI, and ParallelSCSI.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostHba -Device *hba0* | fl

Retrieves the available HBA devices that contain "hba0" in their names.

REMARKS To see the examples, type: "get-help Get-VMHostHba -examples". For more information, type: "get-help Get-VMHostHba -detailed". For technical information, type: "get-help Get-VMHostHba -full". For online help, type: "get-help Get-VMHostHba -online"

Get-VMHostModule

NAME Get-VMHostModule

SYNOPSIS This cmdlet retrieves the option strings of the specified host modules.

SYNTAX Get-VMHostModule [-VMHost] <VMHost[]> [[-Name] <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the option strings of the specified host modules. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies hosts to retrieve their modules.

-Name <String[]> Specifies the names of the host modules you want to retrieve. To find a list of the valid module names for different servers, see the vSphere documentation or in the ESX console, run "esxcfg-module -l". This parameter is mandatory only if you are connected to ESX/vCenter Server with version earlier than 4.0.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostModule -Name Shaper

Retrieves the option string for the specified host module.

REMARKS To see the examples, type: "get-help Get-VMHostModule -examples". For more information, type: "get-help Get-VMHostModule -detailed". For technical information, type: "get-help Get-VMHostModule -full". For online help, type: "get-help Get-VMHostModule -online"

Get-VMHostNetwork

NAME Get-VMHostNetwork

SYNOPSIS THis cmdlet retrieves the host networks on a vCenter Server system.

SYNTAX Get-VMHostNetwork [-Server <VIServer[]>] [-VMHost] <VMHost[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the host networks on a vCenter Server system. This command retrieves the networking configuration of the hosts specified by the VMHost parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VMHost <VMHost[]> Specifies the hosts whose networking configuration you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostNetwork -VMHost 10.23.113.212 | fl

Retrieves the networking configuration for the virtual machine host on IP address 10.23.113.212.

REMARKS To see the examples, type: "get-help Get-VMHostNetwork -examples". For more information, type: "get-help Get-VMHostNetwork -detailed". For technical information, type: "get-help Get-VMHostNetwork -full". For online help, type: "get-help Get-VMHostNetwork -online"

Get-VMHostNetworkAdapter

NAME Get-VMHostNetworkAdapter

SYNOPSIS This cmdlet retrieves the host network adapters on a vCenter Server system.

SYNTAX Get-VMHostNetworkAdapter [-VMHost <VMHost[]>] [[-VirtualSwitch] <VirtualSwitchBase[]>] [-PortGroup <VirtualPortGroupBase[]>] [-Physical] [-VMKernel] [-Console] [[-Name] <String[]>] [-Id <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the host network adapters on a vCenter Server system.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts whose network adapters you want to retrieve. The position of this parameter is deprecated and might change in a following release.

-VirtualSwitch <VirtualSwitchBase[]> Specifies the virtual switches to which network adapters that you want to retrieve are connected. The position of this parameter is deprecated and will be changed in a future release. To avoid errors when you run existing scripts on future PowerCLI versions, specify the parameter by name.

-PortGroup <VirtualPortGroupBase[]> Specifies the port groups to which network adapters that you want to retrieve are connected.

-Physical	Indicates that you want to retrieve only physical network adapters.
-VMKernel	Indicates that you want to retrieve only VMKernel virtual network adapters.
-Console	Indicates that you want to retrieve only service console virtual network adapters.

-Name <String[]> Specifies the names of the host network adapters you want to retrieve. The position of this parameter is deprecated and will be changed in a future release. To avoid errors when you run existing scripts on future PowerCLI versions, specify the parameter by name.

-Id <String[]> Specifies the IDs of the host network adapters you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostNetworkAdapter -VMKernel

Retrieves information about about all VMKernel network adapters on servers that you are connected to.

———— Example 2 ————

C:PS>$myVMHost = Get-VMHost -Name MyVMHost Get-VMHostNetworkAdapter -VMHost $myVMHost -Physical

Retrieves all physical network adapters on the specified host.

———— Example 3 ————

C:PS>$myVDSwitch = Get-VDSwitch -Name MyVDSwitch Get-VMHostNetworkAdapter -VirtualSwitch $myVDSwitch -VMKernel

Retrieves all VMKernel network adapters connected to the specified virtual switch.

———— Example 4 ————

C:PS>Get-VDPortGroup -Name MyVDPortGroup | Get-VMHostNetworkAdapter

Retrieves VMHost network adapters by a specified distributed port group.

———— Example 5 ————

C:PS>$myVirtualSwitch = Get-VirtualSwitch -Name MyVirtualSwitch Get-VMHostNetworkAdapter -VirtualSwitch $myVirtualSwitch

Retrieves physical VMHost network adapters by a specified standard virtual switch.

REMARKS To see the examples, type: "get-help Get-VMHostNetworkAdapter -examples". For more information, type: "get-help Get-VMHostNetworkAdapter -detailed". For technical information, type: "get-help Get-VMHostNetworkAdapter -full". For online help, type: "get-help Get-VMHostNetworkAdapter -online"

Get-VMHostNtpServer

NAME Get-VMHostNtpServer

SYNOPSIS This cmdlet retrieves the NTP servers on the specified hosts.

SYNTAX Get-VMHostNtpServer [-VMHost] <VMHost[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the NTP servers on the specified hosts.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts, whose NTP servers you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostNtpServer -VMHost 10.23.123.100

Retrieves the NTP server of the virtual machine host with an IP address 10.23.123.100.

127.127.1.0

REMARKS To see the examples, type: "get-help Get-VMHostNtpServer -examples". For more information, type: "get-help Get-VMHostNtpServer -detailed". For technical information, type: "get-help Get-VMHostNtpServer -full". For online help, type: "get-help Get-VMHostNtpServer -online"

Get-VMHostPatch

NAME Get-VMHostPatch

SYNOPSIS This cmdlet retrieves information about the host patches installed on the specified hosts. This cmdlet is deprecated and will not return any results for ESX hosts version 5.0 and later. Use (Get-ESXCli).software.vib.list() as an alternative.

SYNTAX Get-VMHostPatch [[-VMHost] <VMHost[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about the host patches installed on the specified hosts. This cmdlet is deprecated and will not return any results for ESX hosts version 5.0 and later. Use (Get-ESXCli).software.vib.list() as an alternative.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve the available patches.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostPatch Host

Retrieves information of the host patches installed on the specified host.

REMARKS To see the examples, type: "get-help Get-VMHostPatch -examples". For more information, type: "get-help Get-VMHostPatch -detailed". For technical information, type: "get-help Get-VMHostPatch -full". For online help, type: "get-help Get-VMHostPatch -online"

Get-VMHostPciDevice

NAME Get-VMHostPciDevice

SYNOPSIS This cmdlet retrieves the PCI devices on the specified hosts.

SYNTAX Get-VMHostPciDevice [-Name <String[]>] [-VMHost <VMHost[]>] [-DeviceClass <PciDeviceClass[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the PCI devices on the specified hosts.

PARAMETERS

-Name <String[]> Filters the PCI devices by name.

Note: This parameter is not case-sensitive.

-VMHost <VMHost[]> Specifies the hosts whose PCI devices you want to retrieve. If not specified, the cmdlet retrieves PCI devices for all hosts on all default connections.

-DeviceClass <PciDeviceClass[]> Limits results to devices of the specified classes.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost "MyHost" | Get-VMHostPciDevice

Retrieves the PCI devices on the "MyVMHost" host.

REMARKS To see the examples, type: "get-help Get-VMHostPciDevice -examples". For more information, type: "get-help Get-VMHostPciDevice -detailed". For technical information, type: "get-help Get-VMHostPciDevice -full". For online help, type: "get-help Get-VMHostPciDevice -online"

Get-VMHostProfile

NAME Get-VMHostProfile

SYNOPSIS This cmdlet retrieves the available host profiles.

SYNTAX Get-VMHostProfile [[-Name] <String[]>] [-Description <String[]>] [-Entity <InventoryItem[]>] [-ReferenceHost <VMHost[]>] [-Id <String[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the available host profiles. A host profile encapsulates the host settings and helps to manage the host configuration.

PARAMETERS

-Name <String[]> Specifies the names of the host profiles you want to retrieve.

-Description <String[]> Specifies a phrase from the description of the host profiles you want to retrieve.

-Entity <InventoryItem[]> Specifies inventory objects associated with the host profiles you want to retrieve.

-ReferenceHost <VMHost[]> Specifies the reference hosts of the profiles you want to retrieve.

-Id <String[]> Specifies the IDs of the host profiles you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostProfile -Name Profile

Retrieves the virtual machine host profile named Profile.

REMARKS To see the examples, type: "get-help Get-VMHostProfile -examples". For more information, type: "get-help Get-VMHostProfile -detailed". For technical information, type: "get-help Get-VMHostProfile -full". For online help, type: "get-help Get-VMHostProfile -online"

Get-VMHostProfileRequiredInput

NAME Get-VMHostProfileRequiredInput

SYNOPSIS This cmdlet performs a check whether the available information is sufficient to apply a host profile.

SYNTAX Get-VMHostProfileRequiredInput [-VMHost] <VMHost> [[-Variable] <Hashtable>] [-Profile <VMHostProfile>] [-Inapplicable] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet performs a check whether the available information is sufficient to apply a host profile, and returns missing values. If the cmdlet returns no output, the information in the hashtable passed to the Variable parameter is sufficient to apply the host profile to the host by using the Apply-VMHostProfile cmdlet.

PARAMETERS

-VMHost <VMHost> Specifies a host to check if the provided information is sufficient for applying the specified host profile.

-Variable <Hashtable> Provides a hash table that contains the available values required for applying the specified profile to the the specified host.

-Profile <VMHostProfile> Specifies a host profile to check if the provided information is sufficient for applying it to the specified host.

-Inapplicable Indicates that you want to view also the elements that are inapplicable to the operation.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHost | Get-VMHostProfileRequiredInput

Retrieves required input for a list of hosts.

————— Example 2 —————

C:PS>Get-VMHostProfileRequiredInput -VMHost $vmhost -Profile $vmhostProfile

Retrieves a required input by specifying a host and a profile.

————— Example 3 —————

C:PS>$result = Get-VMHostProfileRequiredInput -VMHost $vmhost -Variable $requiredInputHashtable;

if (-not $result) { Apply-VMHostProfile -Entity $vmhost -Variable $requiredInputHashtable}

Check whether the specified hashtable that contains values for each required input is exhaustive. If the result is null, then the hashtable can be used by Apply-VMHostProfile.

————— Example 4 —————

C:PS>$result = Get-VMHostProfileRequiredInput -VMHost $vmhost -Variable $requiredInputHashtable -Inapplicable;

$result | where { $_.Type -eq 'Inapplicable' } | foreach { Write-Host "Key $($_.Key) is not applicable for host $vmhost" }

Retrieve the required and inapplicable input and display all inapplicable keys.

REMARKS To see the examples, type: "get-help Get-VMHostProfileRequiredInput -examples". For more information, type: "get-help Get-VMHostProfileRequiredInput -detailed". For technical information, type: "get-help Get-VMHostProfileRequiredInput -full". For online help, type: "get-help Get-VMHostProfileRequiredInput -online"

Get-VMHostRoute

NAME Get-VMHostRoute

SYNOPSIS This cmdlet retrieves the routes from the routing table of the specified hosts.

SYNTAX Get-VMHostRoute [[-VMHost] <VMHost[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the routes from the routing table of the specified hosts.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve routes.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostRoute -VMHost $vmhost

Retrieves the routes for the specified host.

———— Example 2 ————

C:PS>Get-VMHostRoute -VMHost $vmhost1, $vmhost2 -Server $server1, $server2

Retrieves the routes for the specified hosts and servers.

REMARKS To see the examples, type: "get-help Get-VMHostRoute -examples". For more information, type: "get-help Get-VMHostRoute -detailed". For technical information, type: "get-help Get-VMHostRoute -full". For online help, type: "get-help Get-VMHostRoute -online"

Get-VMHostService

NAME Get-VMHostService

SYNOPSIS This cmdlet retrieves information about a host service.

SYNTAX Get-VMHostService [-VMHost] <VMHost[]> [-Refresh] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about a host service. If the Refresh parameter is set to $true, the cmdlet refreshes the host services information before retrieving it. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies hosts for which you want to retrieve the available services.

-Refresh Indicates whether the cmdlet refreshes the service information before retrieving it.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostService -Refresh

Refreshes and retrieves the host services on the default server.

———— Example 2 ————

C:PS>$vmhost = Get-VMHost -Name Host

$vmHostService = Get-VMHostService -VMHost $vmhost

Gets the services of the specified host.

REMARKS To see the examples, type: "get-help Get-VMHostService -examples". For more information, type: "get-help Get-VMHostService -detailed". For technical information, type: "get-help Get-VMHostService -full". For online help, type: "get-help Get-VMHostService -online"

Get-VMHostSnmp

NAME Get-VMHostSnmp

SYNOPSIS This cmdlet retrieves hosts SNMP configuration.

SYNTAX Get-VMHostSnmp [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves hosts SNMP configuration. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostSNMP

Retrieves the host SNMP configuration. Operates on the default server which must be an ESX host.

REMARKS To see the examples, type: "get-help Get-VMHostSnmp -examples". For more information, type: "get-help Get-VMHostSnmp -detailed". For technical information, type: "get-help Get-VMHostSnmp -full". For online help, type: "get-help Get-VMHostSnmp -online"

Get-VMHostStartPolicy

NAME Get-VMHostStartPolicy

SYNOPSIS This cmdlet retrieves the start policy of hosts.

SYNTAX Get-VMHostStartPolicy [-VMHost] <VMHost[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the start policy of the hosts specified by the VMHost parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts whose start policy you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostStartPolicy -VMHost 10.23.113.212

Retrieves the start policy of the virtual machine host on IP address 10.23.113.212.

REMARKS To see the examples, type: "get-help Get-VMHostStartPolicy -examples". For more information, type: "get-help Get-VMHostStartPolicy -detailed". For technical information, type: "get-help Get-VMHostStartPolicy -full". For online help, type: "get-help Get-VMHostStartPolicy -online"

Get-VMHostStorage

NAME Get-VMHostStorage

SYNOPSIS This cmdlet retrieves the host storages on a vCenter Server system.

SYNTAX Get-VMHostStorage [-VMHost] <VMHost[]> [-Refresh] [-RescanAllHba] [-RescanVmfs] [-Server <VIServer[]>] [<CommonParameters>]

Get-VMHostStorage -Id <String[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the host storages on a vCenter Server system. The cmdlet returns a list of the storages on the hosts specified by the VMHost parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts for which you want to retrieve storage information.

-Refresh	Indicates whether the cmdlet refreshes the storage system information before retrieving the specified host storages.
-RescanAllHba	Indicates whether to issue a request to rescan all virtual machine hosts bus adapters for new storage devices prior to retrieving the storage information.
-RescanVmfs	Indicates whether to perform a re-scan for new virtual machine file systems that might have been added, prior to retrieving the storage information.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Id <String[]> Specifies the IDs of the host storages that you want to retrieve.

Note: When a list of values is specified for the Id parameter, the returned objects would have an ID that matches exactly one of the string values in that list.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VmHostStorage -VMHost 10.23.112.234 -Refresh

Retrieves storage information from the virtual machine host with IP address 10.23.112.234.

———— Example 2 ————

C:PS>Get-VMHostStorage | select -expandproperty scsilun | fl *

Retrieves information about the vendor, model, and serial number of the virtual machine host storage SCSI adapter.

Note that for devices, that are not SCSI-3 compliant, the SerialNumber property is not defined. And not all SCSI-3 compliant devices provide the serial number information.

REMARKS To see the examples, type: “get-help Get-VMHostStorage -examples”. For more information, type: “get-help Get-VMHostStorage -detailed”. For technical information, type: “get-help Get-VMHostStorage -full”. For online help, type: “get-help Get-VMHostStorage -online”

Get-VMHostSysLogServer

NAME Get-VMHostSysLogServer

SYNOPSIS This cmdlet displays the remote syslog servers of the specified hosts.

SYNTAX Get-VMHostSysLogServer [-VMHost] <VMHost[]> [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet displays the remote syslog servers of the specified hosts. The returned object contains the server address and the port if configured.

PARAMETERS

-VMHost <VMHost[]> Specifies the host whose remote syslog server you want to display.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$sysLogServer = Get-VMHostSysLogServer -VMHost 10.23.123.234

Retrieves the SysLog server of the virtual machine host with IP address 10.23.123.234.

REMARKS To see the examples, type: “get-help Get-VMHostSysLogServer -examples”. For more information, type: “get-help Get-VMHostSysLogServer -detailed”. For technical information, type: “get-help Get-VMHostSysLogServer -full”. For online help, type: “get-help Get-VMHostSysLogServer -online”

Get-VMQuestion

NAME Get-VMQuestion

SYNOPSIS This cmdlet retrieves the pending questions for the specified virtual machines.

SYNTAX Get-VMQuestion [-VM <VirtualMachine[]>] [[-QuestionText] <String>] [-QuestionId <String>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the pending questions for the specified virtual machines. A question is a task that requires a response from you. If the VM parameter is not specified or its value is $null, the cmdlet returns all questions for all virtual machines on the specified servers.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines whose pending questions you want to retrieve.

-QuestionText <String> Specifies a phrase from the text that describes the questions you want to retrieve.

-QuestionId <String> Specifies the IDs of the questions you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMQuestion -VM VM

Retrieves the questions of the VM virtual machine.

————— Example 2 —————

C:PS>$vm = Get-VM VM

$vm | Get-VMQuestion -QuestionText "*have been moved or copied*"

Retrieves the VM virtual machine questions that contain the phrase "have been moved or copied".

REMARKS To see the examples, type: "get-help Get-VMQuestion -examples". For more information, type: "get-help Get-VMQuestion -detailed". For technical information, type: "get-help Get-VMQuestion -full". For online help, type: "get-help Get-VMQuestion -online"

Get-VMResourceConfiguration

NAME Get-VMResourceConfiguration

SYNOPSIS This cmdlet retrieves information about the resource allocation between the selected virtual machines.

SYNTAX Get-VMResourceConfiguration [-Server <VIServer[]>] [-VM] <VirtualMachine[]> [<CommonParameters>]

DESCRIPTION This cmdlet retrieves information about the resource allocation between the selected virtual machines.

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VM <VirtualMachine[]> Specifies the virtual machines whose resource configuration you want to retrieve.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMResourceConfiguration -VM VM | Format-Custom -Property DiskResourceConfiguration

Displays the disk share for the VM virtual machine.

REMARKS To see the examples, type: "get-help Get-VMResourceConfiguration -examples". For more information, type: "get-help Get-VMResourceConfiguration -detailed". For technical information, type: "get-help Get-VMResourceConfiguration -full". For online help, type: "get-help Get-VMResourceConfiguration -online"

Get-VMStartPolicy

NAME Get-VMStartPolicy

SYNOPSIS This cmdlet retrieves the start policy of the virtual machines on a vCenter Server system.

SYNTAX Get-VMStartPolicy [-VMHost <VMHost[]>] [[-VM] <VirtualMachine[]>] [-Server <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet retrieves the start policy of the virtual machines on a vCenter Server system. The virtual machines are specified by the VM parameter or retrieved from the host passed through the VMHost parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts of the virtual machines whose start policy you want to retrieve.

-VM <VirtualMachine[]> Specifies the virtual machines whose start policy you want to retrieve.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMStartPolicy -VM VM

Retrieves the start policy of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Get-VMStartPolicy -examples". For more information, type: "get-help Get-VMStartPolicy -detailed". For technical information, type: "get-help Get-VMStartPolicy -full". For online help, type: "get-help Get-VMStartPolicy -online"

CHAPTER 16

Import Commands

This page contains details on **Import** commands.

Import-VApp

NAME Import-VApp

SYNOPSIS This cmdlet imports OVF (Open Virtualization Format) and OVA packages. The package can contain a virtual appliance or a virtual machine.

SYNTAX Import-VApp [-Source] <String> [-OvfConfiguration <Hashtable>] [[-Name] <String>] [-Location <VIContainer>] [-VMHost] <VMHost> [-Datastore <StorageResource>] [-Force] [-DiskStorageFormat <VirtualDiskStorageFormat>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet imports OVF (Open Virtualization Format) and OVA packages. The package can contain a vApp or a virtual machine. The cmdlet returns a VApp object when the OVF contains a vApp and a VirtualMachine object when the OVF contains a single virtual machine.

PARAMETERS

-Source <String> Specifies the path to the OVF or OVA package that you want to import.

-OvfConfiguration <Hashtable> Specifies values for a set of user-configurable OVF properties.

-Name <String> Specifies a name for the imported vApp or virtual machine.

-Location <VIContainer> Specifies a vSphere inventory container where you want to import the vApp or virtual machine. It must be a vApp, a resource pool, or a cluster.

-VMHost <VMHost> Specifies a host where you want to run the vApp or virtual machine.

-Datastore <StorageResource> Specifies a datastore or a datastore cluster where you want to store the vApp or virtual machine.

-Force Indicates that you want to import an OVF or OVA package even if the package signature cannot be verified.

-DiskStorageFormat <VirtualDiskStorageFormat> Specifies the storage format for the disks of the imported VMs. By default, the storage format is thick. When you set this parameter, you set the storage format for all virtual machine disks in the OVF package. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

——————— Example 1 ———————

C:PS>$vmHost = Get-VMHost -Name "MyVMHost1" Import-vApp -Source "c:vAppsWebServerWebServer.ovf" -VMHost $vmHost

Imports an OVF package by specifying the target host and name.

——————— Example 2 ———————

C:PS>$myCluster = Get-Cluster -Name "MyCluster1" $vmHost = Get-VMHost -Name "MyVMHost1" Import-vApp -Source "c:vAppsWebServerWebServer.ovf" -VMHost $vmHost -Location $myCluster -Name "MyWebServerProduction1"

Imports an OVF package within a cluster.

——————— Example 3 ———————

C:PS>$vmHost = Get-VMHost -Name "MyVMHost1" $myDatastore = Get-Datastore -Name "MyDatastore1" $vmHost | Import-vApp -Source "c:vAppsWebServerWebServer.ovf" -Datastore $myDatastore

Imports an OVF package by specifying a datastore where to store the virtual machines.

——————— Example 4 ———————

C:PS>$myDatastore = Get-Datastore -Name "MyDatastore1" $vmHost = Get-VMHost -Name "MyVMHost1" $vmHost | Import-vApp -Source "c:vAppsWebServerWebServer.ova" -Datastore $myDatastore -Force

Imports an OVA package even if the package signature cannot be verified.

——————— Example 5 ———————

C:PS>$ovfConfig = Get-OvfConfiguration "myOvfTemplate.ovf" $ovfConfig.NetworkMapping.Network.Value = "Network 2" $ovfConfig.vami.VM_1.ip0.Value = "10.23.101.2" $ovfConfig.vami.VM_2.ip0.Value = "10.23.101.3" Import-VApp $ovfPath -OvfConfiguration $ovfConfig -VMHost $vmHost

Imports an OVF package with specified network mapping and two standard OVF properties.

——————— Example 6 ———————

C:PS>$ovfConfig = Get-OvfConfiguration "myOvfTemplate.ovf" $portGroup = Get-VirtualPortGroup -Name "Network 2" -Standard $ovfConfig.NetworkMapping.Network.Value = $portGroup Import-VApp $ovfPath -OvfConfiguration $ovfConfig -VMHost $vmHost

Imports an OVF package by specifying network mapping with a standard port group object.

———— Example 7 ————

C:PS>$ovfConfig = Get-OvfConfiguration "myOvfTemplate.ovf" $vdPortGroup = Get-VDPortgroup "my-DistributedPortGroup" $ovfConfig.NetworkMapping.Network.Value = $vdPortGroup Import-VApp $ovfPath -OvfConfiguration $ovfConfig -VMHost $vmHost

Imports an OVF package by specifying network mapping with a distributed port group object.

———— Example 8 ————

C:PS>$ovfConfig.ToHashTable() $ovfConfig = @{

"NetworkMapping.VM Test Network"="Network 2"; "vami.ip0.VM_1"="10.23.101.2"; "vami.ip0.VM_2"="10.23.101.3"

} Import-VApp $ovfPath -OvfConfiguration $ovfConfig -VMHost $vmHost

Imports an OVF package by specifying a hash table with populated OVF properties to the OvfConfiguration parameter.

REMARKS To see the examples, type: "get-help Import-VApp -examples". For more information, type: "get-help Import-VApp -detailed". For technical information, type: "get-help Import-VApp -full". For online help, type: "get-help Import-VApp -online"

Import-VMHostProfile

NAME Import-VMHostProfile

SYNOPSIS This cmdlet imports a host profile from a file. The file path must be accessible from the vSphere PowerCLI client side.

SYNTAX Import-VMHostProfile [-FilePath] <String> [-Name] <String> [[-ReferenceHost] <VMHost>] [-Description <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet imports a host profile from a file. The file path must be accessible from the vSphere PowerCLI client side.

PARAMETERS

-FilePath <String> Specifies the path to the file, from which you want to import a host profile.

-Name <String> Specifies a name of the imported host profile.

-ReferenceHost <VMHost> Specifies a reference host for the imported host profile.

-Description <String> Specifies a description for the imported host profile.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Import-VMHostProfile -FilePath export.prf -Name Profile

Imports a virtual machine host profile from the export.prf file and names it Profile.

REMARKS To see the examples, type: "get-help Import-VMHostProfile -examples". For more information, type: "get-help Import-VMHostProfile -detailed". For technical information, type: "get-help Import-VMHostProfile -full". For online help, type: "get-help Import-VMHostProfile -online"

CHAPTER 17

Install Commands

This page contains details on **Install** commands.

Install-VMHostPatch

NAME Install-VMHostPatch

SYNOPSIS This cmdlet updates the specified hosts.

SYNTAX Install-VMHostPatch [-VMHost] <VMHost[]> -HostPath <String[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

Install-VMHostPatch [-VMHost] <VMHost[]> -WebPath <String[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

Install-VMHostPatch [-VMHost] <VMHost[]> -LocalPath <String[]> [-HostUsername <String>] [-HostPassword <SecureString>] [-HostCredential <PSCredential>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet updates the specified hosts. The cmdlet installs patches on the host. The patches that can be located locally, on a Web location, or in a host file system. When using the LocalPath or WebPath parameters, the ESX/ESXi host attempts to store the patch contents in its local temporary directory. Because ESXi hosts might not have enough free space on their local drives, this cannot apply to large size patches. The best practice for upgrading an ESXi host is to upload the patch contents on the host's datastore and to run Install-VMHostPatch with the HostPath parameter. If you want to install patches packaged in a ZIP archive, you must extract them and use one of the HostPatch, LocalPath, or WebPath parameters. If you use the HostPath parameter, you must extract each patch to a temporary folder that is named after the patch ID (for example, c:tempESX400-200906001), and copy the folder in the root folder of a datastore. Note that the datastore path is case-sensitive. If you use the LocalPath parameter, you must extract each patch to a folder. The name of the folder must contain the patch ID (for example, "ESX400-200906001"). If you use the WebPath parameter, you must extract each patch to a folder that is published on a Web server. The patch URL address must contain the patch ID (for example, http://myInternalWebServer/esx40/ESX400-200906001/). Depending on the component to be upgraded, you might have to set the host into a maintenance mode and to restart the host or the hostd management service after applying the patch. This cmdlet is experimental and might be changed or removed in a future release.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to update.

-HostPath <String[]> Specifies a file path on the ESX/ESXi host to the patches you want to install.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-WebPath <String[]> Specifies the Web location of the patches you want to install.

-LocalPath <String[]> Specifies the local file system path to the patches you want to install. Providing credentials when installing a patch from a local path is mandatory.

-HostUsername <String> Specifies the username you want to use to authenticate with the host.

-HostPassword <SecureString> Specifies the password you want to use to authenticate with the host.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Install-VMHostPatch -VMHost $vmhost1,$vmhost2 -LocalPath c:esx40patchesESX400-200906001metadata.zip -HostUsername admin -HostPassword pass

Updates ESX servers using a local file. Before running the cmdlet, you must download the patch file locally and extract to a folder. The name of the folder must contain the patch ID (for example, "ESX400-200906001"). Providing credentials when installing a patch from a local path is mandatory.

————— Example 2 —————

C:PS>$vmhost | Install-VMHostPatch -WebPath http://myInternalWebServer/esx40/ESX400-200906001/metadata.zip

Upgrades an ESX server using a Web location. Before running the cmdlet, you must download the patch file and extract it to a folder that is published on a Web server. The patch URL address must contain the patch ID (for example, http://myInternalWebServer/esx40/ESX400-200906001/).

————— Example 3 —————

C:PS>$datastore = Get-Datastore -Name Datastore

Copy-DatastoreItem c:tempESX400-200906001

$datastore.DatastoreBrowserPath -Recurse

$vmhost1,$vmhost2 | Install-VMHostPatch -HostPath /vmfs/volumes/datastore/ESX400-200906001/metadata.zip

Upgrades ESX servers using the -HostPath parameter. First, you must download the patch file and extract its contents to a temporary folder that is named after the patch ID (for example, c:tempESX400-200906001). Copy the folder in the root folder of the Datastore datastore and run Install-VMHostPatch providing the datastore path to the patch. Note that the datastore path is case-sensitive.

REMARKS To see the examples, type: "get-help Install-VMHostPatch -examples". For more information, type: "get-help Install-VMHostPatch -detailed". For technical information, type: "get-help Install-VMHostPatch -full". For online help, type: "get-help Install-VMHostPatch -online"

CHAPTER 18

Invoke Commands

This page contains details on **Invoke** commands.

Invoke-DrsRecommendation

NAME Invoke-DrsRecommendation

SYNOPSIS This cmdlet applies the specified DRS recommendations.

SYNTAX Invoke-DrsRecommendation [-DrsRecommendation] <DrsRecommendation[]> [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet applies the specified DRS recommendations.

PARAMETERS

-DrsRecommendation <DrsRecommendation[]> Specifies the DRS recommendations you want to apply.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-DrsRecommendation -Priority 1,2 | Invoke-DrsRecommendation

Retrieves and applies DRS recommendations with priorities 1 and 2.

————— Example 2 —————

C:PS>$drs = Get-DrsRecommendation -Cluster Cluster Invoke-DrsRecommendation -DrsRecommendation $drs -RunAsync

Retrieves the DRS recommendations from the Cluster cluster and applies them. The command returns without waiting for the task to complete.

REMARKS To see the examples, type: "get-help Invoke-DrsRecommendation -examples". For more information, type: "get-help Invoke-DrsRecommendation -detailed". For technical information, type: "get-help Invoke-DrsRecommendation -full". For online help, type: "get-help Invoke-DrsRecommendation -online"

Invoke-VMHostProfile

NAME Invoke-VMHostProfile

SYNOPSIS This cmdlet applies a host profile to the specified host or cluster.

SYNTAX Invoke-VMHostProfile [-Entity] <InventoryItem[]> [-Profile <VMHostProfile>] [-Variable <Hashtable>] [-AssociateOnly] [-ApplyOnly] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet applies a host profile to the specified host or cluster. The host or cluster must be in a maintenance mode. If no value is provided to the Profile parameter, the profile currently associated with the host or cluster is applied.

PARAMETERS

-Entity <InventoryItem[]> Specifies hosts or clusters to which you want to apply the virtual machine host profile.

-Profile <VMHostProfile> Specifies the host profile you want to apply.

-Variable <Hashtable> Specifies a hash table object that provides values for the host profile required variables.

-AssociateOnly	Indicates whether to associate the host profile to the specified host or cluster without applying it.
-ApplyOnly	Indicates whether to apply the host profile to the specified virtual machine host without associating it.
-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Invoke-VMHostProfile -AssociateOnly -Entity $cluster -Profile $profile

Associates the specified profile to all hosts in the specified cluster.

————— Example 2 —————

C:PS>Invoke-VMHostProfile -Entity $vmhost -Profile $profile

Associates and applies the specified profile to the specified host.

————— Example 3 —————

C:PS>Get-VMHost | Invoke-VMHostProfile -ApplyOnly -Profile $profile

Applies the specified profile to all specified hosts.

————— Example 4 —————

C:PS>Get-VMHost | Invoke-VMHostProfile -AssociateOnly -profile $profile

Associates the specified profile to all specified hosts.

————— Example 5 —————

C:PS>Invoke-VMHostProfile $vmhost

Applies the associated host's profile to the host.

————— Example 6 —————

C:PS>$requireInput = Invoke-VMHostProfile $vmhost -Profile $profile;

$requireInput['network.hostPortGroup["key-vim-profile-host-HostPortgroupProfile-VMkernel"].ipConfig.IpAddressPolicy.address'] = '192.168.0.1';

$requireInput['network.hostPortGroup["key-vim-profile-host-HostPortgroupProfile-VMkernel"].ipConfig.IpAddressPolicy.subnetmask'] = '255.255.255.0';

Invoke-VMHostProfile $vmhost -Profile $profile -Variable $requireInput;

Applies a profile to host but first assigns values to all required values.

REMARKS To see the examples, type: "get-help Invoke-VMHostProfile -examples". For more information, type: "get-help Invoke-VMHostProfile -detailed". For technical information, type: "get-help Invoke-VMHostProfile -full". For online help, type: "get-help Invoke-VMHostProfile -online"

Invoke-VMScript

NAME Invoke-VMScript

SYNOPSIS This cmdlet runs a script in the guest OS of each of the specified virtual machines.

SYNTAX Invoke-VMScript [-ScriptText] <String> [-VM] <VirtualMachine[]> [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-GuestCredential <PSCredential>] [-GuestUser <String>] [-GuestPassword <SecureString>] [-ToolsWaitSecs <Int32>] [-ScriptType <ScriptType>] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet runs a script in the guest OS of each of the specified virtual machines. To run Invoke-VMScript, the user must have read access to the folder containing the virtual machine and a Virtual Machine.Interaction.Console Interaction privilege. The virtual machines must be powered on and have VMware Tools installed. Network connectivity to the ESX system hosting the virtual machine on port 902 must be present. To authenticate with the host or the guest OS, one of the HostUser/HostPassword (GuestUser/GuestPassword) pair and HostCredential (GuestCredential) parameters must be provided. The guest account you use to authenticate with the guest operating system must have administrator's privileges. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Modify and VirtualMachine.GuestOperations.Execute privileges.

PARAMETERS

-ScriptText <String> Provides the text of the script you want to run. You can also pass to this parameter a string variable containing the path to the script.

-VM <VirtualMachine[]> Specifies the virtual machines on whose guest operating systems you want to run the script.

-HostCredential <PSCredential> Specifies a PSCredential object containing the credentials you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-GuestCredential <PSCredential> Specifies a PSCredential object containing the credentials you want to use for authenticating with the virtual machine guest OS.

-GuestUser <String> Specifies the user name you want to use for authenticating with the virtual machine guest OS.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the virtual machine guest OS.

-ToolsWaitSecs <Int32> Specifies how long in seconds the system waits for connecting to the VMware Tools. The default value is 20.

-ScriptType <ScriptType> Specifies the type of the script. The valid values are PowerShell, Bat, and Bash. If the virtual machine OS is Windows, the default value is PowerShell. If the virtual machine OS is Linux, the default value is Bash.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Invoke-VMScript -VM VM -ScriptText "dir" -GuestUser administrator -GuestPassword pass2

Lists the directory entries on the guest OS.

————— Example 2 —————

C:PS>$script = '&"$env:ProgramFilesCommon FilesMicrosoft SharedMSInfomsinfo32.exe" /report "$env:Tmpinforeport"'

Invoke-VMScript -ScriptText $script -VM VM -GuestCredential $guestCredential

Runs a PowerShell script. In PowerShell, to access environment variables, you must use the following syntax: $env:<environment variable> (for example, $env:ProgramFiles). Also, to run the program, you must specify an ampersand (&) in front of the program path. The outer quotes ($script = '...') are required because this is how you define a string variable in PowerShell. The inner double quotes are required because there are spaces in the path.

————— Example 3 —————

C:PS>$script = '"%programfiles%Common FilesMicrosoft SharedMSInfomsinfo32.exe" /report "%tmp%inforeport"'

Invoke-VMScript -ScriptText $script -VM VM -GuestCredential $guestCredential -ScriptType Bat

Runs a BAT script. In BAT scripts, to access environment variables, you must use the following syntax: %<environment variable>% (for example, %programfiles%).

The outer quotes ($script = '...') are required because this is how you define a string variable in PowerShell. The inner double quotes are required because there are spaces in the path.

REMARKS To see the examples, type: "get-help Invoke-VMScript -examples". For more information, type: "get-help Invoke-VMScript -detailed". For technical information, type: "get-help Invoke-VMScript -full". For online help, type: "get-help Invoke-VMScript -online"

CHAPTER 19

Mount Commands

This page contains details on **Mount** commands.

Mount-Tools

NAME Mount-Tools

SYNOPSIS This cmdlet mounts the VMware Tools CD installer as a CD-ROM on the guest operating system.

SYNTAX Mount-Tools [[-Guest] <VMGuest[]>] [<CommonParameters>]

Mount-Tools [[-VM] <VirtualMachine[]>] [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet mounts the VMware Tools CD installer as a CD-ROM on the guest operating system that is specified by the Guest or VM parameters. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Guest <VMGuest[]> Specifies the guest operating systems on which you want to mount VMware Tools.

-VM <VirtualMachine[]> Specifies the virtual machines on which you want to mount VMware Tools.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Mount-Tools VM

Mounts the VMware Tools on the specified virtual machine. The virtual machine must be powered on.

————— Example 2 —————

C:PS>Get-VMGuest VM | Mount-Tools

Mounts the VMware Tools on the virtual machine specified by its guest operating system. The virtual machine must be powered on.

REMARKS To see the examples, type: “get-help Mount-Tools -examples”. For more information, type: “get-help Mount-Tools -detailed”. For technical information, type: “get-help Mount-Tools -full”. For online help, type: “get-help Mount-Tools -online”

CHAPTER 20

Move Commands

This page contains details on **Move** commands.

Move-Cluster

NAME Move-Cluster

SYNOPSIS This cmdlet moves a vCenter Server cluster from one location to another.

SYNTAX Move-Cluster [-Cluster] <Cluster[]> [-Destination] <VIContainer> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a vCenter Server cluster to the location that is specified by the Destination parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Cluster <Cluster[]> Specifies the clusters you want to move to another location.

-Destination <VIContainer> Specifies the folder or datacenter where you want to move the clusters. If a datacenter is specified for the Destination parameter, the cluster is moved to its "hostFolder" folder. The "hostFolder" is a system folder and is guaranteed to exist.

Note: You cannot move clusters from one datacenter to another. You can only move clusters between folders and to datacenter level.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Move-Cluster -Cluster Cluster -Destination Folder

Moves the cluster named Cluster into the folder called Folder on the same datacenter

REMARKS To see the examples, type: “get-help Move-Cluster -examples”. For more information, type: “get-help Move-Cluster -detailed”. For technical information, type: “get-help Move-Cluster -full”. For online help, type: “get-help Move-Cluster -online”

Move-Datacenter

NAME Move-Datacenter

SYNOPSIS This cmdlet moves a vCenter Server datacenter from one location to another.

SYNTAX Move-Datacenter [-Datacenter] <Datacenter[]> [-Destination] <VIContainer> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a vCenter Server datacenter to the location that is specified by the Destination parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Datacenter <Datacenter[]> Specifies the datacenters you want to move to another location.

-Destination <VIContainer> Specifies the folder where you want to move the datacenters.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Move-Datacenter Datacenter -Destination Folder

Moves the Datacenter datacenter into the folder named Folder.

REMARKS To see the examples, type: “get-help Move-Datacenter -examples”. For more information, type: “get-help Move-Datacenter -detailed”. For technical information, type: “get-help Move-Datacenter -full”. For online help, type: “get-help Move-Datacenter -online”

Move-Datastore

NAME Move-Datastore

SYNOPSIS This cmdlet moves datastores from one location to another.

SYNTAX Move-Datastore [-Datastore] <Datastore[]> [-Destination] <VIObject> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves datastores from one location to another.

PARAMETERS

-Datastore <Datastore[]> Specifies the datastore that you want to move.

-Destination <VIObject> Specifies a datastore cluster, folder, or datacenter where you want to place the datastore.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Move-Datastore “MyDatastore” -Destination “MyDatastoreFolder”

Moves the MyDatastore datastore to the specified folder.

———— Example 2 ————

C:PS>$myDatastoreCluster = Get-DatastoreCluster -Name “MyDatatoreCluster” Get-Datastore “MyDatastore1”, “MyDatastore2” | Move-Datstore -Destination $myDatastoreCluster

Moves two datastores to a specified datastore cluster.

REMARKS To see the examples, type: “get-help Move-Datastore -examples”. For more information, type: “get-help Move-Datastore -detailed”. For technical information, type: “get-help Move-Datastore -full”. For online help, type: “get-help Move-Datastore -online”

Move-Folder

NAME Move-Folder

SYNOPSIS This cmdlet moves a vCenter Server folder from one location to another.

SYNTAX Move-Folder [-Folder] <Folder[]> [-Destination] <VIContainer> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a vCenter Server folder to the location that is specified by the Destination parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-Folder <Folder[]> Specifies the folders you want to move to another location.

-Destination <VIContainer> Specifies the datacenter or folder where you want to move the folders. If a datacenter is specified for the Destination parameter, the folder is moved to the datacenter's hostFolder or vmFolder folder, depending on the folder's child item type. The hostFolder and vmFolder are system folders and are guaranteed to exist.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Folder -Name "vmFolder" | Move-Folder -Destination "destinationVmFolder"

Moves the "vmFolder" folder into another folder of the same type named "destinationVmFolder".

REMARKS To see the examples, type: "get-help Move-Folder -examples". For more information, type: "get-help Move-Folder -detailed". For technical information, type: "get-help Move-Folder -full". For online help, type: "get-help Move-Folder -online"

Move-HardDisk

NAME Move-HardDisk

SYNOPSIS This cmdlet moves a hard disk from one location to another.

SYNTAX Move-HardDisk [-HardDisk] <HardDisk[]> [-Datastore] <Datastore> [-StorageFormat <VirtualDiskStorageFormat>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a hard disk from one location to another.

PARAMETERS

-HardDisk <HardDisk[]> Specifies the hard disk that you want to move to another location.

-Datastore <Datastore> Specifies a datastore where you want to place the hard disk.

-StorageFormat <VirtualDiskStorageFormat> Specifies the storage format of the relocated hard disk. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myDatastore1 = Get-Datastore -Name 'MyDatastore1' $myDisk = Get-VM -Name MyVm1 | Get-HardDisk Move-HardDisk -HardDisk $myDisk -Datastore $myDatastore1

Moves the hard disk of a specified virtual machine to another datastore.

————— Example 2 —————

C:PS>$myDisk = Get-VM -Name 'MyVM1' | Get-HardDisk $myDatastore1 = Get-Datastore -Name 'MyDatastore1' Move-hardDisk - HardDisk $myDisk -Datastore $myDatastore1 - StorageFormat 'EagerZeroedThick'

Moves the hard disk of a specified virtual machine to another datastore and changes the storage format of the hard disk to EagerZeroedThick.

REMARKS To see the examples, type: "get-help Move-HardDisk -examples". For more information, type: "get-help Move-HardDisk -detailed". For technical information, type: "get-help Move-HardDisk -full". For online help, type: "get-help Move-HardDisk -online"

Move-Inventory

NAME Move-Inventory

SYNOPSIS This cmdlet moves a vCenter Server inventory item from one location to another.

SYNTAX Move-Inventory [-Item] <InventoryItem[]> [-Destination] <VIContainer> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a vCenter Server inventory object or template to the location that is specified by the Destination parameter.

PARAMETERS

-Item <InventoryItem[]> Specifies the Folder, ResourcePool, Datacenter, VirtualMachine, VMHost, Template, or Cluster objects you want to move to another location.

-Destination <VIContainer> Specifies the location where you want to move the inventory items.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vm = Get-VM -Name VM*

Move-Inventory -Item $vm -Destination Folder

Moves the virtual machines whose names start with VM to the Folder folder.

————— Example 2 —————

C:PS>Get-Folder Folder1 | Get-Inventory -NoRecursion | Move-Inventory -Destination Folder2

Moves all objects from the Folder1 folder to the Folder2 folder.

REMARKS To see the examples, type: "get-help Move-Inventory -examples". For more information, type: "get-help Move-Inventory -detailed". For technical information, type: "get-help Move-Inventory -full". For online help, type: "get-help Move-Inventory -online"

Move-ResourcePool

NAME Move-ResourcePool

SYNOPSIS This cmdlet moves a resource pool from one location to another.

SYNTAX Move-ResourcePool [-ResourcePool] <ResourcePool[]> [-Destination] <VIContainer> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a resource pool to the location that is specified by the Destination parameter. To specify a server different from the default one, use the Server parameter. Moving a resource pool is only supported when the objects assigned to the ResourcePool and Destination parameters are passed through connections to one and the same server with the same credentials.

PARAMETERS

-ResourcePool <ResourcePool[]> Specifies the resource pools you want to move to another location.

-Destination <VIContainer> Specifies the location where you want to move the resource pools. If a host or a cluster is specified for the Destination parameter, the resource pool is moved into a resource pool named Resources. The Resources resource pool is a system resource pool and is guaranteed to exist.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Move-ResourcePool -ResourcePool ResourcePool -Destination Host

Moves the resource pool named ResourcePool to the virtual machine host Host.

REMARKS To see the examples, type: "get-help Move-ResourcePool -examples". For more information, type: "get-help Move-ResourcePool -detailed". For technical information, type: "get-help Move-ResourcePool -full". For online help, type: "get-help Move-ResourcePool -online"

Move-Template

NAME Move-Template

SYNOPSIS This cmdlet moves virtual machine templates to another location.

SYNTAX Move-Template [-Template] <Template[]> [-Destination] <VIContainer> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves virtual machine templates to a location that is specified by the Destination parameter.

PARAMETERS

-Template <Template[]> Specifies the virtual machine templates you want to move to another location.

-Destination <VIContainer> Specifies a container object where you want to move the templates.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Move-Template -Template $template -Destination $dest

Moves the $template object into the $dest container object.

REMARKS To see the examples, type: "get-help Move-Template -examples". For more information, type: "get-help Move-Template -detailed". For technical information, type: "get-help Move-Template -full". For online help, type: "get-help Move-Template -online"

Move-VApp

NAME Move-VApp

SYNOPSIS This cmdlet moves the specified virtual appliances to a new location.

SYNTAX Move-VApp [-Destination] <VIContainer> [-VApp] <VApp[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves the specified vApps to a new location. If the destination is a host or a cluster, the vApps are moved to the system "Resources" resource pool.

PARAMETERS

-Destination <VIContainer> Specifies where you want to move the specified vApps. Supported types are Folder, VMHost, Cluster, ResourcePool, VApp, and Datacenter.

-VApp <VApp[]> Specifies the vApps you want to move.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmHost = Get-VMHost -Name "MyVMHost1" $myDestinationRP = New-ResourcePool -Name "vApp ResourcePool" -Location $vmHost Move-VApp -VApp (Get-Vapp -Name "MyVApp1" -Location $vmHost) -Destination $myDestinationRP

Moves a vApp from a host to a resource pool from the same host.

————— Example 2 —————

C:PS>$vmHost = Get-VMHost -Name “MyVMHost1” $myDestinationVApp = New-VApp -Name “MyvApp1” -Location $vmHost (Get-Vapp -Name “MyvApp2” -Location (Get-ResourcePool -Name “MyResourcePool1”) | Move-VApp -Destination $myDestinationVApp

Moves a vApp from a resource pool to another vApp.

———— Example 3 ————

C:PS>Move-VApp -Name “MyvApp1” (Get-VMHost -Name “MyVMHost1”)

Moves a vApp from a resource pool to a host.

———— Example 4 ————

C:PS>$myVmFolder1 = Get-Folder -Name “MyVMFolder1” -Location (Get-Datacenter -Name “MyDatacenter1”) -NoRecursion $myVMFolder2 = New-Folder -Name “MyVMFolder2” -Location $myVmFolder1 Get-VApp -Name “MyVApp” | Move-VApp -Destination $myVMFolder2 -RunAsync

Moves a virtual appliance to a folder asynchronously.

REMARKS To see the examples, type: “get-help Move-VApp -examples”. For more information, type: “get-help Move-VApp -detailed”. For technical information, type: “get-help Move-VApp -full”. For online help, type: “get-help Move-VApp -online”

Move-VM

NAME Move-VM

SYNOPSIS This cmdlet moves virtual machines to another location.

SYNTAX Move-VM [-AdvancedOption <AdvancedOption[]>] [[-Destination] <VIContainer>] [-Datastore <StorageResource>] [-DiskStorageFormat <VirtualDiskStorageFormat>] [-VMotionPriority <VMotionPriority>] [-RunAsync] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves a virtual machine to the location that is specified by the Destination or the Datastore parameters. The destination must be a folder, host, cluster, or a resource pool. You can move a virtual machine to a DRS cluster. Moving a virtual machine to the top level of a non-DRS cluster is only possible if the virtual machine is in a resource pool in that cluster. If the virtual machine is outside the non-DRS cluster, you need to specify a virtual machine host in that cluster as destination. When moving virtual machines that are powered on, vMotion is used. To specify a server different from the default one, use the Server parameter. You can move storage and compute resources simultaneously.

PARAMETERS

-AdvancedOption <AdvancedOption[]> This parameter is only applicable when a DatastoreCluster object is passed to the Datastore parameter. Specifies one or more rules for the placement of the virtual machines that you want to relocate.

To indicate that VMs should be stored on different datastores, pass an SdrsVMAntiAffinityRule object to the parameter. You can set more than one Storage DRS (SDRS) VM anti-affinity rules.

To indicate that the VM disks should be stored on different datastores, pass an SdrsVMDiskAntiAffinityRule object to the parameter. You can set only one SDRS VM disk anti-affinity rule.

-Destination <VIContainer> Specifies a folder, host, cluster, or a resource pool where you want to move the virtual machines. If a datacenter is specified for the Destination parameter, the virtual machines are moved to the datacenter’s “vmFolder” folder. The “vmFolder” is a system folder and is guaranteed to exist. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-Datastore <StorageResource> Specifies the datastore or datastore cluster where you want to move the virtual machines. When you pass a datastore cluster to the Datastore parameter, you can also set the AdvancedOption parameter.

-DiskStorageFormat <VirtualDiskStorageFormat> Specifies a new storage format for the hard disk of the virtual machine you want to move. This parameter is applicable only when moving a virtual machine to a different datastore, using the Datastore parameter. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-VMotionPriority <VMotionPriority> Determines the priority that should be used for a vMotion operation.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-VM <VirtualMachine[]> Specifies the virtual machines you want to move to another location.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VM -Name VM | Move-VM -Destination 10.23.112.235

Moves the virtual machine named VM from its current location to the host on IP address 10.23.112.235.

———— Example 2 ————

C:PS>Move-VM -VM VM -Destination Folder

Moves the virtual machine to a folder called Folder. Note that you are able to move virtual machines only to folders containing virtual machines (the 'blue' folders in the vSphere Client).

———— Example 3 ————

C:PS>Move-VM -VM 'MyVM' -Destination 'MyDestination'

Moves a powered-on virtual machine from one existing host to another, by using vMotion, passing parameters by name.

———— Example 4 ————

C:PS>Move-VM -VM 'MyVM' -Datastore 'MyDatastore'

Moves a powered-on virtual machine from one existing datastore to another, by using storage vMotion, passing parameters by name.

———— Example 5 ————

C:PS>$myDatastoreCluster1 = Get-DatastoreCluster -Name 'MyDatastoreCluster1' Move-VM -VM 'MyVM1' -Datastore $myDatastoreCluster1

Moves the MyVM1 virtual machine to any datastore in the specified datastore cluster.

————— Example 6 —————

C:PS>$myDatastoreCluster1 = Get-DatastoreCluster -Name 'MyDatastoreCluster1' $myVm2 = Get-VM -Name 'MyVM2' $vmAntiAffinityRule = New-Object -TypeName VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMAntiAffinityRule -ArgumentList $myVm2 Move-VM -VM 'MyVM1' -Datastore $myDatastoreCluster1 -AdvancedOption $vmAntiAffinityRule

Moves the MyVM1 virtual machine to the specified datastore cluster and sets a VM anti-affinity rule for the placement of the virtual machine. The MyVM1 virtual machine will be placed on any datastore in the specified datastore cluster that does not contain the MyVM2 virtual machine.

————— Example 7 —————

C:PS>$myVm1 = Get-VM -Name 'MyVM1' $vmdks = Get-Harddisk -VM $myVm1 $myDatastoreCluster1 = Get-DatastoreCluster -Name 'MyDatastoreCluster1' $vmdkAntiAffinityRule = New-Object -TypeName VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMDiskAntiAffinityRule -ArgumentList $vmdks Move-VM -VM '$myVm1' -Datastore $myDatastoreCluster1 -AdvancedOption $vmdkAntiAffinityRule

Moves the MyVM1 virtual machine to the specified datastore cluster and sets a VM disk anti-affinity rule for the placement of the virtual machine. The disks of the MyVM1 virtual machine will be stored on different datastores in the specified datastore cluster.

————— Example 8 —————

C:PS>Get-VM -Name 'MyVM' -Server 'MyServer' | Move-VM -Destination 'NewHost' -Datastore 'DatastoreOnNewHost' -DiskStorageFormat 'Thin'

Moves a powered-off virtual machine to another datastore on another host and changes its disk storage format to thin.

REMARKS To see the examples, type: "get-help Move-VM -examples". For more information, type: "get-help Move-VM -detailed". For technical information, type: "get-help Move-VM -full". For online help, type: "get-help Move-VM -online"

Move-VMHost

NAME Move-VMHost

SYNOPSIS This cmdlet moves hosts to another location.

SYNTAX Move-VMHost [-VMHost] <VMHost[]> [-Destination] <VIContainer> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet moves hosts to the location that is specified by the Destination parameter. To specify a server different from the default one, use the Server parameter.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to move to another location.

-Destination <VIContainer> Specifies the location where you want to move the hosts.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Move-VMHost 192.168.112.113 -Destination Datacenter

Moves the host with IP address 10.23.112.113 to a different datacenter named Datacenter.

REMARKS To see the examples, type: "get-help Move-VMHost -examples". For more information, type: "get-help Move-VMHost -detailed". For technical information, type: "get-help Move-VMHost -full". For online help, type: "get-help Move-VMHost -online"

CHAPTER 21

New Commands

This page contains details on **New** commands.

New-AdvancedSetting

NAME New-AdvancedSetting

SYNOPSIS This cmdlet creates a new advanced setting for the specified entity.

SYNTAX New-AdvancedSetting [-Name] <String> [-Value] <Object> [-Entity] <VIObject[]> [-Type <AdvancedSettingType>] [-Force] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new advanced setting for the specified entity.

PARAMETERS

-Name <String> Specifies a name for the advanced setting.

-Value <Object> Specifies a value for the advanced setting.

-Entity <VIObject[]> Specifies the entity for which you want to create a new advanced setting. This parameter accepts VIServer, VirtualMachine, DatastoreCluster, and Cluster objects.

-Type <AdvancedSettingType> Specifies the type of the new advanced setting.

-Force Indicates that you want to create the new advanced setting even if another setting with the same name exists for the specified object type.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-AdvancedSetting -Entity (Get-Cluster -Name Cluster) -Name SettingName -Value SettingValue -Type ClusterHA

Creates a new advanced setting for the Cluster cluster - of type CLusterHA, with name SettingName and value SettingValue.

REMARKS To see the examples, type: "get-help New-AdvancedSetting -examples". For more information, type: "get-help New-AdvancedSetting -detailed". For technical information, type: "get-help New-AdvancedSetting -full". For online help, type: "get-help New-AdvancedSetting -online"

New-AlarmAction

NAME New-AlarmAction

SYNOPSIS This cmdlet creates an alarm action and attaches it to the specified alarm.

SYNTAX New-AlarmAction [-AlarmDefinition] <AlarmDefinition> -Snmp [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-AlarmAction [-AlarmDefinition] <AlarmDefinition> -Email [-Subject <String>] -To <String[]> [-Cc <String[]>] [-Body <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-AlarmAction [-AlarmDefinition] <AlarmDefinition> -Script -ScriptPath <String> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates an alarm action and attaches it to the specified alarm.

PARAMETERS

-AlarmDefinition <AlarmDefinition> Specifies the alarm definition for which you want to configure actions.

-Snmp	Indicates that a SNMP message is sent when the alarm is activated.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.
-Email	Indicates that when the alarm is activated, the system sends an email message to the specified address. Use the Subject, To, CC, and Body parameters to customize the alarm message.

-Subject <String> Specifies a subject for the email message you want to send.

-To <String[]> Specifies the email address to which you want to send a message.

-Cc <String[]> Specifies the email addresses you want to add to the CC field of the email message.

-Body <String> Specifies the text of the email message.

-Script	Indicates that a script is run when the alarm is activated.

-ScriptPath <String> Specifies the path to a batch file, located on a vCenter Server system, that will run when the alarm is activated.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-AlarmDefinition -Name "Alarm1" | New-AlarmAction -Snmp

Creates an alarm action SNMP.

———— Example 2 ————

C:PS>Get-AlarmDefinition -Name "Alarm1" | New-AlarmAction -Script -ScriptPath 'c:test.bat'

Creates an alarm action script.

———— Example 3 ————

C:PS>Get-AlarmDefinition -Name "Alarm1" | New-AlarmAction -Email -To 'test@vmware.com' -CC @('test1@vmware.com', 'test2@vmware.com') -Body 'Test body' -Subject 'Test subject'

Creates an alarm action Email.

REMARKS To see the examples, type: "get-help New-AlarmAction -examples". For more information, type: "get-help New-AlarmAction -detailed". For technical information, type: "get-help New-AlarmAction -full". For online help, type: "get-help New-AlarmAction -online"

New-AlarmActionTrigger

NAME New-AlarmActionTrigger

SYNOPSIS This cmdlet creates a new action trigger for the specified alarm action.

SYNTAX New-AlarmActionTrigger [-StartStatus] <InventoryItemStatus> [-EndStatus] <InventoryItemStatus> -AlarmAction <AlarmAction> [-Repeat] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new action trigger for the specified alarm action.

PARAMETERS

-StartStatus <InventoryItemStatus> Specifies the start status for the alarm action. The valid values are Green, Yellow, and Red.

-EndStatus <InventoryItemStatus> Specifies the end status for the alarm action. The valid values are Green, Yellow, and Red.

-AlarmAction <AlarmAction> Specifies the alarm action for which you want to create an action trigger.

-Repeat	Indicates whether you want the alarm action to repeat until the alarm is acknowledged.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-AlarmDefinition -Name "Alarm1" | Get-AlarmAction | New-AlarmActionTrigger -StartStatus 'Red' -EndStatus 'Yellow' -Repeat

Creates an action trigger for all actions for the specified alarm definition.

REMARKS To see the examples, type: "get-help New-AlarmActionTrigger -examples". For more information, type: "get-help New-AlarmActionTrigger -detailed". For technical information, type: "get-help New-AlarmActionTrigger -full". For online help, type: "get-help New-AlarmActionTrigger -online"

New-CDDrive

NAME New-CDDrive

SYNOPSIS This cmdlet creates a new virtual CD drive.

SYNTAX New-CDDrive [-IsoPath <String>] [-HostDevice <String>] [-StartConnected] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new virtual CD drive for each of the provided virtual machines. If an ISO location is provided, sets the CD to point to the ISO.

PARAMETERS

-IsoPath <String> Specifies the datastore path to the ISO (CD image) file that backs the virtual CD drive. Do not set this parameter if the HostDevice parameter is set.

-HostDevice <String> Specifies the path to the CD drive on the virtual machine host that backs the virtual CD drive. Do not set this parameter if the ISOPath parameter is set.

-StartConnected	Indicates that the virtual CD drive starts connected when the virtual machine associated with it powers on.

-VM <VirtualMachine[]> Specifies the virtual machine to which the new virtual CD drive belongs.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>New-CDDrive -VM $vm -ISOPath "Path_to_ISOtest.iso"

Creates a CD drive on the specified virtual machine and attach an ISO image to it.

REMARKS To see the examples, type: "get-help New-CDDrive -examples". For more information, type: "get-help New-CDDrive -detailed". For technical information, type: "get-help New-CDDrive -full". For online help, type: "get-help New-CDDrive -online"

New-Cluster

NAME New-Cluster

SYNOPSIS This cmdlet creates a new cluster.

SYNTAX New-Cluster [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-VMSwapfilePolicy <VMSwapfilePolicy>] [-Name] <String> -Location <VIContainer> [-HAEnabled] [-HAAdmissionControlEnabled] [-HAFailoverLevel <Int32>] [-DrsEnabled] [-DrsMode <DrsMode>] [-DrsAutomationLevel <DrsAutomationLevel>] [-VsanDiskClaimMode <VsanDiskClaimMode>] [-VsanEnabled] [-EVCMode <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new cluster with the provided inputs, in the location that is specified by the Location parameter. HAEnabled is automatically set to $true if some of the HA settings, HAAdmissionControlEnabled, HAFailoverLevel, HARestartPriority, HAIsolationResponse, are specified . DrsEnabled is automatically set to $true if some of the DRS settings, DrsAutomationLevel, DrsMode, are specified .

PARAMETERS

-HARestartPriority <HARestartPriority> Specifies the cluster HA restart priority. The valid values are Disabled, Low, Medium, and High. VMware HA is a feature that detects failed virtual machines and automatically restarts them on alternative ESX hosts. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-HAIsolationResponse <HAIsolationResponse> Indicates that the virtual machine should be powered off if a host determines that it is isolated from the rest of the compute resource. The valid values are PowerOff and DoNothing. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-VMSwapfilePolicy <VMSwapfilePolicy> Specifies the swapfile placement policy. The following values are valid:

InHostDataStore - Store the swapfile in the datastore that is specified by the VMSwapfileDatastoreID property of the virtual machine host. If the VMSwapfileDatastoreID property is not set or indicates a datastore with unsufficient free space, store the swapfile in the same directory as the virtual machine. This setting might degrade VMotion performance.

WithVM - Store the swapfile in the same directory as the virtual machine.

-Name <String> Specifies the name of the new cluster.

-Location <VIContainer> Specifies the location where you want to place the new cluster. If a datacenter is specified for the Location parameter, the cluster is created in its "hostFolder" folder. The "hostFolder" is a system folder and is guaranteed to exist.

-HAEnabled	Indicates that VMware HA (High Availability) is enabled.
-HAAdmissionControlEnabled	Indicates that virtual machines cannot be powered on if they violate availability constraints.

-HAFailoverLevel <Int32> Specifies a configured failover level. This is the number of physical host failures that can be tolerated without impacting the ability to meet minimum thresholds for all running virtual machines. The valid values range from 1 to 4.

-DrsEnabled	Indicates that VMware DRS (Distributed Resource Scheduler) is enabled.

-DrsMode <DrsMode> This parameter is deprecated and scheduled for removal. Use the DrsAutomationLevel parameter instead.

Specifies a DRS (Distributed Resource Scheduler) mode. The valid values are FullyAutomated, Manual, and PartiallyAutomated.

-DrsAutomationLevel <DrsAutomationLevel> Specifies a DRS (Distributed Resource Scheduler) automation level. The valid values are FullyAutomated, Manual, and PartiallyAutomated.

-VsanDiskClaimMode <VsanDiskClaimMode> Specifies the mode by which disks are claimed by the Virtual SAN. If not specified and VsanEnabled is specified, the assumed value is Manual.

-VsanEnabled	Indicates that the Virtual SAN feature is enabled on this cluster.

-EVCMode <String> Specifies the VMware Enhanced vMotion Compatibility (EVC) mode of the newly created cluster. If not specified or set to $null, EVC is disabled.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-Cluster -Name “MyCluster” -Location “MyDatacenter”

Creates a new cluster named “MyCluster” in the “MyDatacenter” datacenter.

————— Example 2 —————

C:PS>New-Cluster -Name “MyCluster” -Location “MyDatacenter” -HAEnabled -HAAdmissionControlEnabled -HAFailoverLevel 2 -VMSwapfilePolicy “InHostDatastore” -HARestartPriority “Low” -HAIsolationResponse “PowerOff”

Creates a new cluster named “MyCluster” in the “MyDatacenter” datacenter, with specified VMware HA (Hgh Availability) settings.

————— Example 3 —————

C:PS>New-Cluster -Name “MyCluster” -Location “MyDatacenter” -DRSEnabled -DRSAutomationLevel ‘Manual’

Creates a new cluster named “MyCluster” in the “MyDatacenter” datacenter, with specified VMware DRS (Distributed Resource Scheduler) settings.

————— Example 4 —————

C:PS>New-Cluster -Name “MyCluster” -Location “MyDatacenter” -EVCMode ‘intel-nehalem’

Creates a new cluster named “MyCluster” in the “MyDatacenter” datacenter, with specified VMware EVC (Enhanced vMotion Compatibility) settings.

REMARKS To see the examples, type: “get-help New-Cluster -examples”. For more information, type: “get-help New-Cluster -detailed”. For technical information, type: “get-help New-Cluster -full”. For online help, type: “get-help New-Cluster -online”

New-CustomAttribute

NAME New-CustomAttribute

SYNOPSIS This cmdlet creates a new custom attribute.

SYNTAX New-CustomAttribute [-Name] <String> [[-TargetType] <CustomAttributeTargetType[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new custom attribute. A custom attribute is a user-defined description field of one or more vCenter Server objects.

PARAMETERS

-Name <String> Specifies a name for the new custom attribute.

-TargetType <CustomAttributeTargetType[]> Specifies the type of the objects to which the new custom attribute applies. The valid values are VirtualMachine, ResourcePool, Folder, VMHost, Cluster, Datacenter, and $null. If the value is $null the custom attribute is global and applies to all target types.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>New-CustomAttribute -Name “CompanyName” -TargetType VMHost, VirtualMachine

Creates a new custom attribute named CompanyName for the virtual machines and hosts on the server.

REMARKS To see the examples, type: “get-help New-CustomAttribute -examples”. For more information, type: “get-help New-CustomAttribute -detailed”. For technical information, type: “get-help New-CustomAttribute -full”. For online help, type: “get-help New-CustomAttribute -online”

New-Datacenter

NAME New-Datacenter

SYNOPSIS This cmdlet creates a new datacenter.

SYNTAX New-Datacenter [-Location] <VIContainer> [-Name] <String> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new datacenter in the location that is specified by the Location parameter.

PARAMETERS

-Location <VIContainer> Specifies the location where you want to create the new datacenter.

-Name <String> Specifies a name for the new datacenter.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$folder = Get-Folder -NoRecursion | New-Folder -Name Folder

New-Datacenter -Location $folder -Name Datacenter | fl

Gets the inventory root folder and create a new folder called Folder in it. Creates a new datacenter called Datacenter in the Folder folder. The result is pipelined to the fl command to retrieve a flat view of the new datacenter properties.

REMARKS To see the examples, type: “get-help New-Datacenter -examples”. For more information, type: “get-help New-Datacenter -detailed”. For technical information, type: “get-help New-Datacenter -full”. For online help, type: “get-help New-Datacenter -online”

New-Datastore

NAME New-Datastore

SYNOPSIS This cmdlet creates a new datastore.

SYNTAX New-Datastore [-Server <VIServer[]>] [-VMHost] <VMHost[]> [-Name] <String> -Path <String> [-Vmfs] [-BlockSizeMB <Int32>] [-FileSystemVersion <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-Datastore [-Server <VIServer[]>] [-VMHost] <VMHost[]> [-Name] <String> -Path <String> [-Nfs] -NfsHost <String[]> [-ReadOnly] [-Kerberos] [-FileSystemVersion <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new datastore based on the provided parameters. The following characters cannot be used in a datastore name: slash (/), backslash (), and percent (%).

PARAMETERS

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-VMHost <VMHost[]> Specifies a host where you want to create the new datastore.

-Name <String> Specifies a name for the new datastore.

-Path <String> If you want to create an NFS datastore, specify the remote path of the NFS mount point. If you want to create a VMFS datastore, specify the canonical name of the SCSI logical unit that will contain new VMFS datastores.

-Vmfs Indicates that you want to create a VMFS datastore.

-BlockSizeMB <Int32> Specifies the maximum file size of VMFS in megabytes (MB). If no value is given, the maximum file size for the current system platform is used.

-FileSystemVersion <String> Specifies the file system you want to use on the new datastore.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Nfs Indicates that you want to create an NFS datastore.

-NfsHost <String[]> Specifies the NFS host where you want to create the new datastore.

-ReadOnly Indicates that the access mode for the mount point is ReadOnly. The default access mode is ReadWrite.

-Kerberos By default, NFS datastores are created with AUTH_SYS as the authentication protocol. This parameter indicates that the NFS datastore uses Kerberos version 5 for authentication. This parameter is available only for NFS version 4.1 datastores.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>New-Datastore -VMHost $vmhost -Name Datastore -Path $scsiLun.CanonicalName -Vmfs -FileSystemVersion 3

Creates a VMFS datastore by specifying the file system type.

———— Example 2 ————

C:PS>New-Datastore -Nfs -VMHost 10.23.112.60 -Name NFSDatastore -Path /mynfs -NfsHost 10.23.84.73

Creates a NFS datastore.

———— Example 3 ————

C:PS>$vmhost1, $vmhost2 | New-Datastore -Nfs -Name NFS1 -Path "/mnt/nfs1/nfs11/test1" -NfsHost 10.23.113.55 -ReadOnly

Creates a read-only NFS datastore across multiple virtual machine hosts.

REMARKS To see the examples, type: "get-help New-Datastore -examples". For more information, type: "get-help New-Datastore -detailed". For technical information, type: "get-help New-Datastore -full". For online help, type: "get-help New-Datastore -online"

New-DatastoreCluster

NAME New-DatastoreCluster

SYNOPSIS This cmdlet creates a new datastore cluster.

SYNTAX New-DatastoreCluster [-Name] <String> -Location <VIContainer> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new datastore cluster. By default, Storage DRS is disabled. To enable Storage DRS, run Set-DatastoreCluster.

PARAMETERS

-Name <String> Specifies a name for the datastore cluster that you want to create.

-Location <VIContainer> Specifies a container object (Datacenter or Folder) where you want to place the new datastore cluster.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

——————— Example 1 ———————

C:PS>New-DatastoreCluster -Name ‘MyDatastoreCluster’ -Location ‘MyDatacenter’

Creates a new datastore cluster on the specified datacenter.

REMARKS To see the examples, type: “get-help New-DatastoreCluster -examples”. For more information, type: “get-help New-DatastoreCluster -detailed”. For technical information, type: “get-help New-DatastoreCluster -full”. For online help, type: “get-help New-DatastoreCluster -online”

New-DrsRule

NAME New-DrsRule

SYNOPSIS This cmdlet creates a new DRS rule.

SYNTAX New-DrsRule [-Name] <String> [-Cluster] <Cluster[]> [-Enabled <Boolean>] -KeepTogether <Boolean> -VM <VirtualMachine[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new DRS rule. Each rule defines the virtual machines that can run on the same host (affinity rule) or must run on different hosts (anti-affinity rule).

PARAMETERS

-Name <String> Specifies a name for the new DRS rule.

-Cluster <Cluster[]> Specifies the clusters for which the new DRS rule applies.

-Enabled [<Boolean>] If the value of this parameter is $true, the new DRS rule is enabled for the specified clusters. If the value is $false, it is disabled.

-KeepTogether [<Boolean>] If the value of this parameter is $true, the new DRS rule is an affinity rule. If the value is $false, the DRS rule is an anti-affinity rule.

-VM <VirtualMachine[]> Specifies the virtual machines that are referenced by the new DRS rule.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>New-DrsRule -Cluster $cluster -Name antiAffinityRule1 -KeepTogether $false -VM $antiAffinityVMs

Creates a new DRS rule for the cluster saved in the $cluster variable with the specified parameters.

REMARKS To see the examples, type: "get-help New-DrsRule -examples". For more information, type: "get-help New-DrsRule -detailed". For technical information, type: "get-help New-DrsRule -full". For online help, type: "get-help New-DrsRule -online"

New-FloppyDrive

NAME New-FloppyDrive

SYNOPSIS This cmdlet creates a new virtual floppy drive.

SYNTAX New-FloppyDrive [-FloppyImagePath <String>] [-NewFloppyImagePath <String>] [-HostDevice <String>] [-StartConnected] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new virtual floppy drive for each of the provided virtual machines. If a floppy image path is provided, sets the floppy drive to point to the image. If both the FloppyImagePath and HostDevice parameters are specified, an error is generated.

PARAMETERS

-FloppyImagePath <String> Specifies the datastore path to the floppy image file backing the virtual floppy drive. Do not use this parameter together with the HostDevice parameter.

-NewFloppyImagePath <String> Specifies a new datastore path to a floppy image file backing the virtual floppy drive. Do not use this parameter together with the HostDevice parameter.

-HostDevice <String> Specifies the path to the floppy drive on the host which will back this virtual floppy drive. Do not use this parameter together with the FloppyImagePath parameter.

-StartConnected Indicates that the virtual floppy drive starts connected when its associated virtual machine powers on.

-VM <VirtualMachine[]> Specifies the virtual machines to which you want to attach the new virtual floppy drive.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-FloppyDrive -VM VM -HostDevice '/dev/fd0' -StartConnected

Creates a floppy drive backed by the client device /dev/fd0 and sets it to start connected when the virtual machine is powered on.

REMARKS To see the examples, type: "get-help New-FloppyDrive -examples". For more information, type: "get-help New-FloppyDrive -detailed". For technical information, type: "get-help New-FloppyDrive -full". For online help, type: "get-help New-FloppyDrive -online"

New-Folder

NAME New-Folder

SYNOPSIS This cmdlet creates a new folder on a vCenter Server system.

SYNTAX New-Folder [-Name] <String> [-Location] <VIContainer> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new folder on the specified location.

PARAMETERS

-Name <String> Specifies a name for the new folder.

-Location <VIContainer> Specifies a container object (folder, datacenter, or cluster) where you want to place the new folder. If a datacenter is specified for the Location parameter, then the folder is created in its "hostFolder" folder and contains hosts and clusters. The "hostFolder" is a system folder and is guaranteed to exist.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-Folder -Name "Folder1" -Location (Get-Datacenter)[0]

Creates a new folder in a datacenter root.

————— Example 2 —————

C:PS>New-Folder -Name "Folder1" -Location (Get-Cluster)[0]

Creates a new folder in a cluster root.

————— Example 3 —————

C:PS>Get-Folder | Select -First 1 | New-Folder -Name "Folder2"

Creates a nested folder by using a pipeline command.

REMARKS To see the examples, type: "get-help New-Folder -examples". For more information, type: "get-help New-Folder -detailed". For technical information, type: "get-help New-Folder -full". For online help, type: "get-help New-Folder -online"

New-HardDisk

NAME New-HardDisk

SYNOPSIS This cmdlet creates a new hard disk on the specified location.

SYNTAX New-HardDisk [-AdvancedOption <AdvancedOption[]>] [[-Persistence] <String>] [-Controller <ScsiController>] [[-DiskType] <DiskType>] [-CapacityKB <Int64>] [-CapacityGB <Decimal>] [-Split] [-ThinProvisioned] [-StorageFormat <VirtualDiskStorageFormat>] [-DeviceName <String>] [-Datastore <StorageResource>] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-HardDisk [[-Persistence] <String>] [-Controller <ScsiController>] -DiskPath <String> [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new hard disk on the specified virtual machine or datastore. When a new virtual disk with raw disk mapping (RDM) backing is created, the compatibility mode of "virtual" or "physical" must be specified using the DiskType parameter. In "virtual" compatibility mode, the disk can use the specified disk modes. In "physical" compatibility mode, the disk modes are ignored and commands are passed directly to the backing Logical Unit Number (LUN). If "flat" mode is set by the DiskType parameter, the virtual disk backing is pre-allocated. If the hard disk is attached to no virtual machine, the value of the DiskType parameter might be Unknown, which means that no type is specified. Use the Persistence parameter to make the disk Persistent (changes are immediately and permanently written to the disk), Nonpersistent (changes to the disk are discarded when you power off or reset the virtual machine), IndependentPersistent, IndependentNonPersistent, or Undoable.

PARAMETERS

-AdvancedOption <AdvancedOption[]> Specifies advanced options for creating hard disks. Accepts only SdrsVMDiskAntiAffinityRule objects. You can define an anti-affinity SDRS rule for the disk by specifying a SdrsVMDiskAntiAffinityRule object to the AdvancedOption parameter and this will override any existing SdrsVMDiskAntiAffinityRule for the virtual machine.

The SdrsVMDiskAntiAffinityRule defines a Storage DRS intra-VM anti-affinity rule (vm disk anti-affinity rule). It is only applicable when creating a virtual machine or hard disk on a datastore cluster. An instance of the object is created by invoking its constructor. There are two constructors - "public SdrsVMDiskAntiAffinityRule(param string[] diskIdentifier)" and "public SdrsVMDiskAntiAffinityRule(param HardDisk[] disk)". For the first constructor, "diskIdentifier" can be either the disk key or the index of the disk in the disk array. The specified disks (and the disk to which the rule is applied) are placed in an anti-affinity rule on a DatastoreCluster. Only one such rule is supported per a virtual machine. You can pass the instance to the AdvancedOption parameter of the New-VM or New-HardDisk cmdlets.

The SDRS functionality is experimental.

-Persistence <String> Specifies the disk persistence mode. The valid values are Persistent, NonPersistent, IndependentPersistent, IndependentNonPersistent, and Undoable. This parameter is supported only when the disk type is set to “rawVirtual” or “flat”. The ‘NonPersistent’ and ‘Undoable’ values are deprecated and scheduled for removal. Their usage is not recommended because they do not work with snapshots and are not supported on ESX 3.5 and later.

-Controller <ScsiController> Specifies a SCSI controller to which you want to attach the new hard disk.

-DiskType <DiskType> Specifies the type of file backing you want to use. The valid values are rawVirtual, rawPhysical, flat, and unknown. If the hard disk is attached to no virtual machine, the value of the DiskType parameter might be Unknown, which means that no type is specified.

-CapacityKB <Int64> This parameter is obsolete. Use CapacityGB instead. Specifies the capacity of the new virtual disk in kilobytes (KB). You need to specify this parameter when you create hard disks of type Flat.

-CapacityGB <Decimal> Specifies the capacity of the new virtual disk in gigabytes (GB). You need to specify this parameter when you create hard disks of type Flat.

-Split	This parameter is deprecated and scheduled for removal. Use the StorageFormat instead. Specifies the type of the virtual disk file - split or monolithic. If the value is $true, the virtual disk is stored in multiple files, each 2GB. If the value is $false, the virtual disk is stored in a single file. This parameter is supported only if the DiskType parameter is set to “flat”.
-ThinProvisioned	This parameter is deprecated and scheduled for removal. Use the StorageFormat instead. Indicates to the underlying file system, that the virtual disk backing file should be allocated lazily (using thin provisioning). This parameter is only used for file systems that support configuring the provisioning policy on a per file basis, such as VMFS3. This parameter is supported only if the DiskType parameter is set to “flat”.

-StorageFormat <VirtualDiskStorageFormat> Specifies the storage format of the new hard disk. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-DeviceName <String> Specifies the host-specific device the LUN is being accessed through. If the target LUN is not available on the host then it is empty. For example, this could happen if it has been masked out accidentally. Only supported when DiskType is set to “rawVirtual” or “rawPhysical”. The device name is visible in the vSphere Client through the new raw hard disk wizard or can be retrieved using PowerCLI views.

-Datastore <StorageResource> Specifies the datastore where you want to place the new hard disk. If a DatastoreCluster object is passed to the Datastore parameter, the hard disk is added to the DatastoreCluster in an automated SDRS mode. You can define an anti-affinity SDRS rule for the disk by specifying an SdrsVMDiskAntiAffinityRule object to the AdvancedOption parameter and this will override any existing SdrsVMDiskAntiAffinityRule for the virtual machine.

-VM <VirtualMachine[]> Specifies the virtual machines to which you want to add the new disk.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-DiskPath <String> Specifies the path to the hard disk.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vm = Get-VM VM

$vm | New-HardDisk -CapacityGB 100 -Persistence persistent

Adds to the VM virtual machine a new hard disk in a persistent mode with capacity of 100 GB.

————— Example 2 —————

C:PS>$deviceName = ($vmhost | Get-ScsiLun | Where {$_.CanonicalName -match "naa"})[0].ConsoleDeviceName

New-HardDisk -VM $vm -DiskType RawPhysical -DeviceName $deviceName

Obtains a valid device name for Raw Disk Mapping. Then the command creates a RDM hard disk for the specified virtual machine, with the obtained device name.

————— Example 3 —————

C:PS>New-HardDisk -VM $vm -CapacityGB 100 -Persistence IndependentNonPersistent

Creates a non-persistent hard disk with the specified capacity.

————— Example 4 —————

C:PS>New-HardDisk -VM $vm -DiskPath "[storage1] OtherVM/OtherVM.vmdk"

Attaches an available disk from a *.vmdk file.

————— Example 5 —————

C:PS>$vm = Get-VM WebServerVM

$disk = $vm | Get-HardDisk

$antiAffinityRule = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMDiskAntiAffinityRu
$disk

New-HardDisk -VM $vm -AdvancedOption $antiAffinityRule -CapacityGB 40 -Datastore DatastoreCluster1

First retrieves the existing disk which will be part of the VMDK anti affinity rule. Then, creates an object describing the rule and creates the new hard disk.

REMARKS To see the examples, type: "get-help New-HardDisk -examples". For more information, type: "get-help New-HardDisk -detailed". For technical information, type: "get-help New-HardDisk -full". For online help, type: "get-help New-HardDisk -online"

New-IScsiHbaTarget

NAME New-IScsiHbaTarget

SYNOPSIS This cmdlet creates a new iSCSI HBA target.

SYNTAX New-IScsiHbaTarget -IScsiHba <IScsiHba[]> [-Address] <String[]> [[-Port] <Int32>] [-Type <IScsiHbaTargetType>] [[-IScsiName] <String>] [-ChapType <ChapType>] [-ChapName <String>] [-ChapPassword <String>] [-MutualChapEnabled <Boolean>] [-MutualChapName <String>] [-MutualChapPassword <String>] [-InheritChap <Boolean>] [-InheritMutualChap <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new iSCSI HBA target. The cmdlet also enables and configures the CHAP (Challenge Handshake Authentication Protocol) authentication settings of the new target.

The Address parameter supports both IPv4 and v6 and also supports the string representations of these types. e.g. "<address>:<port>". The Port parameter is used only when the value of the Address parameter does not contain the port. The default port number is 3260.

PARAMETERS

-IScsiHba <IScsiHba[]> Specifies the iSCSI HBA for which you want to create the new target.

-Address <String[]> Specifies the address of the new iSCSI HBA target.

-Port <Int32> Specifies the TCP port of the target.

-Type <IScsiHbaTargetType> Specifies the type of the target. The valid values are Static and Send.

-IScsiName <String> Specifies the iSCSI name of the target. It can be specified only for Static targets.

-ChapType <ChapType> Specifies the type of the CHAP (Challenge Handshake Authentication Protocol) you want the new target to use. The valid values are Prohibited, Discouraged, Preferred, and Required.

-ChapName <String> Specifies a CHAP authentication name for the new target.

-ChapPassword <String> Specifies a CHAP authentication password for the new target.

-MutualChapEnabled [<Boolean>] Indicates that Mutual CHAP is enabled.

-MutualChapName <String> Specifies a Mutual CHAP authentication name for the new target.

-MutualChapPassword <String> Specifies a Mutual CHAP authentication password for the new target.

-InheritChap [<Boolean>] Indicates that the CHAP setting is inherited from the iSCSI HBA.

-InheritMutualChap [<Boolean>] Indicates that the Mutual CHAP setting is inherited from the iSCSI HBA.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$hba = Get-VMHost | Get-VMHostHba -Type iScsi

New-IScsiHbaTarget -IScsiHba $hba -Address 10.23.84.73

Creates a new target with IP address 10.23.84.73 on the specified iSCSI HBA device.

————— Example 2 —————

C:PS>Get-VMHost | Get-VMHostHba -Type iScsi | New-IScsiHbaTarget -Address "10.23.84.73" -ChapType Preferred -ChapName user -ChapPassword pass

Creates a new target on the provided iSCSI HBA device and configures the CHAP settings of the target.

REMARKS To see the examples, type: “get-help New-IScsiHbaTarget -examples”. For more information, type: “get-help New-IScsiHbaTarget -detailed”. For technical information, type: “get-help New-IScsiHbaTarget -full”. For online help, type: “get-help New-IScsiHbaTarget -online”

New-NetworkAdapter

NAME New-NetworkAdapter

SYNOPSIS This cmdlet creates a new virtual network adapter.

SYNTAX New-NetworkAdapter [-MacAddress <String>] -NetworkName <String> [-StartConnected] [-WakeOnLan] [-Type <VirtualNetworkAdapterType>] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-NetworkAdapter [-MacAddress <String>] [-StartConnected] [-WakeOnLan] [-Type <VirtualNetworkAdapterType>] -PortId <String> -DistributedSwitch <DistributedSwitch> [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-NetworkAdapter [-MacAddress <String>] [-StartConnected] [-WakeOnLan] [-Type <VirtualNetworkAdapterType>] -Portgroup <VirtualPortGroupBase> [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new virtual network adapter for each of the provided virtual machines and sets the optional properties if provided.

PARAMETERS

-MacAddress <String> Specifies an optional MAC address for the new virtual network adapter.

-NetworkName <String> Specifies the name of the network to which you want to add the new virtual network adapter. Specifying a distributed port group name is obsolete. Use the Portgroup parameter instead.

-StartConnected Indicates that the virtual network adapter starts connected when the virtual machine associated with it powers on.

-WakeOnLan Indicates that wake-on-LAN is enabled on the newly created virtual network adapter.

-Type <VirtualNetworkAdapterType> Specifies the type of the new network adapter. The valid types are e1000, Flexible, Vmxnet, EnhancedVmxnet, and Vmxnet3, and Unknown. If no value is given to the parameter, the new network adapter is of the type recommended by VMware for the given guest OS.

-VM <VirtualMachine[]> Specifies the virtual machines to which you want to attach the new virtual network adapter.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-PortId <String> Specifies the port of the specified distributed switch to which you want to connect the network adapter. Use this parameter only if the DistributedSwitch parameter is specified.

-DistributedSwitch <DistributedSwitch> Specifies a virtual switch to which you want to connect the network adapter.

-Portgroup <VirtualPortGroupBase> Specifies a standard or a distributed port group to which you want to connect the new network adapter.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VM VM | New-NetworkAdapter -NetworkName "VM Network" -MacAddress '00:50:56:a1:00:00' -WakeOnLan -StartConnected -Type EnhancedVmxnet

Create a virtual network adapter with the specified parameters.

————— Example 2 —————

C:PS>$myVm = Get-VM -Name MyVM $MyVDPortgroup = Get-VDPortgroup -Name MyVDPortGroup New-NetworkAdapter -VM $myVM -Portgroup $MyVDPortgroup

Adds a new network adapter to the specified virtual machine and connects it to the specified distributed port group.

————— Example 3 —————

C:PS>$myVM = Get-VM -Name MyVM $MyVDSwitch = Get-VDSwitch -Name MyVDSwitch New-NetworkAdapter -VM $myVM -DistributedSwitch $MyVDSwitch -PortId 100

Adds a new network adapter to the specified virtual machine and connects it to the specified port on the specified vSphere distributed switch.

REMARKS To see the examples, type: "get-help New-NetworkAdapter -examples". For more information, type: "get-help New-NetworkAdapter -detailed". For technical information, type: "get-help New-NetworkAdapter -full". For online help, type: "get-help New-NetworkAdapter -online"

New-OSCustomizationNicMapping

NAME New-OSCustomizationNicMapping

SYNOPSIS This cmdlet adds NIC settings mappings to the specified OS customization specifications.

SYNTAX New-OSCustomizationNicMapping -OSCustomizationSpec <OSCustomizationSpec[]> [-Server <VIServer[]>] [-IpMode <OSCustomizationIPMode>] [-VCApplicationArgument <String>] [[-IpAddress] <String>] [[-SubnetMask] <String>] [[-DefaultGateway] <String>] [-AlternateGateway <String>] [[-Dns] <String[]>] [-Wins <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-OSCustomizationNicMapping -OSCustomizationSpec <OSCustomizationSpec[]> [-NetworkAdapterMac <String[]>] [-Server <VIServer[]>] [-IpMode <OSCustomizationIPMode>] [-VCApplicationArgument <String>] [[-IpAddress] <String>] [[-SubnetMask] <String>] [[-DefaultGateway] <String>] [-AlternateGateway <String>] [[-Dns] <String[]>] [-Wins <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-OSCustomizationNicMapping -OSCustomizationSpec <OSCustomizationSpec[]> [-Position <Int32[]>] [-Server <VIServer[]>] [-IpMode <OSCustomizationIPMode>] [-VCApplicationArgument <String>] [[-IpAddress] <String>] [[-SubnetMask] <String>] [[-DefaultGateway] <String>] [-AlternateGateway <String>] [[-Dns] <String[]>] [-Wins <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet adds NIC settings mappings to the specified OS customization specifications. If the given specification is server-side, it is updated on the server. If it is client-side, the reference that is kept in-memory is updated but the variable that is passed to the cmdlet is not modified.

PARAMETERS

-OSCustomizationSpec <OSCustomizationSpec[]> Specifies the OS customization specification to which you want to add the NIC setting mapping.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-IpMode <OSCustomizationIPMode> Specifies the IP configuration mode. The valid values are UseDhcp, PromptUser, UseVCApplication, and UseStaticIP.

-VCApplicationArgument <String> Specifies an optional argument you want to pass to the vCenter Server to obtain an IP address.

-IpAddress <String> Specifies an IP address. Using this parameter automatically sets the IpMode parameter to UseStaticIp.

-SubnetMask <String> Specifies a subnet mask.

-DefaultGateway <String> Specifies a default gateway.

-AlternateGateway <String> Specifies an alternate gateway.

-Dns <String[]> Specifies a DNS address. This parameter applies only to Windows operating systems.

-Wins <String[]> Specifies WINS servers. This parameter applies only to Windows operating systems.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-NetworkAdapterMac <String[]> Specifies the MAC addresses of the network adapters to which you want to map the new OS customization specifications.

-Position <Int32[]> Specifies the position of the NIC to which you want to map the OS customization specification.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-OSCustomizationNicMapping -OSCustomizationSpec $spec -IpMode UseStaticIP -IPAddress 10.0.0.1 -SubnetMask 255.255.255.0 -DefaultGateway 10.0.0.253 -DnsServer 10.0.0.253

Creates a new NIC mapping for the OS customization spec stored in $spec.

REMARKS To see the examples, type: "get-help New-OSCustomizationNicMapping -examples". For more information, type: "get-help New-OSCustomizationNicMapping -detailed". For technical information, type: "get-help New-OSCustomizationNicMapping -full". For online help, type: "get-help New-OSCustomizationNicMapping -online"

New-OSCustomizationSpec

NAME New-OSCustomizationSpec

SYNOPSIS This cmdlet creates a new OS customization specification.

SYNTAX New-OSCustomizationSpec [-OSType <String>] [-Server <VIServer[]>] [-Name <String>] [-Type <OSCustomizationSpecType>] [-DnsServer <String[]>] [-DnsSuffix <String[]>] [-Domain <String>] [-NamingScheme <String>] [-NamingPrefix <String>] [-Description <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-OSCustomizationSpec -OSCustomizationSpec <OSCustomizationSpec> [-Server <VIServer[]>] [-Name <String>] [-Type <OSCustomizationSpecType>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-OSCustomizationSpec -FullName <String> -OrgName <String> [-OSType <String>] [-ChangeSid] [-DeleteAccounts] [-Server <VIServer[]>] [-Name <String>] [-Type <OSCustomizationSpecType>] [-DnsServer <String[]>] [-DnsSuffix <String[]>] [-GuiRunOnce <String[]>] [-AdminPassword <String>] [-TimeZone <String>] [-AutoLogonCount <Int32>] [-Domain <String>] [-Workgroup <String>] [-DomainCredentials <PSCredential>] [-DomainUsername <String>] [-DomainPassword <String>] [-ProductKey <String>] [-NamingScheme <String>] [-NamingPrefix <String>] [-Description <String>] [-LicenseMode <LicenseMode>] [-LicenseMaxConnections <Int32>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new OS customization specification or clones an existing one. If a name is provided, creates and adds the specified customization specification to the server. Otherwise, creates and returns the requested specification object. If the Name parameter is not specified, the OSCustomizationSpec object is not persisted on the server. Either the Domain or the Workgroup parameters should be provided if a Windows specification is created. If a Linux specification is created, the Domain parameter is mandatory. New-OSCustomizationSpec automatically creates a default NIC mapping.

PARAMETERS

-OSType <String> Specifies the type of the operating system. The valid values are Linux and Windows.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Name <String> Specifies a name for the new specification.

-Type <OSCustomizationSpecType> Specifies the type of the OS customization specification. The valid values are Persistent and NonPersistent.

-DnsServer <String[]> Specifies the DNS server settings. This parameter applies only to Linux operating systems.

-DnsSuffix <String[]> Specifies the DNS suffix settings. This parameter applies only to Linux operating systems.

-Domain <String> Specifies a domain name.

-NamingScheme <String> Specifies the naming scheme for the virtual machine. The following values are valid:

Custom - Specifies that vCenter Server will launch an external application to generate the (hostname/IP). The command line for this application must be specified in the server configuration file (vpxd.cfg) in the vpxd/name-ip-generator key.

Fixed - Specifies that the name is fixed.

Prefix - Specifies that a unique name should be generated by concatenating the base string with a number. Virtual machine names are unique across the set of hosts and virtual machines known to the vCenter Server system. vCenter Server tracks the network names of virtual machines as well as hosts. VMware Tools runs in a guest operating system and reports information to vCenter Server, including the network name of the guest.

Vm - Specifies that vCenter Server should generate a virtual machine name from a base prefix comprising the virtual machine entity name. A number is appended, if necessary, to make it unique. Virtual machine names are unique across the set of hosts and virtual machines known to the vCenter Server system. VMware Tools reports the names of existing virtual machines.

-NamingPrefix <String> Depends on the customization naming scheme - Custom, NamingPrefix, or Prefix. If the "Custom" naming scheme is used, NamingPrefix is an optional argument that is passed to the utility for this IP address. The meaning of this field is user-defined in the script. If the "Fixed" naming scheme is used, NamingPrefix should indicate the name of the virtual machine. If the "Prefix" naming scheme is selected, NamingPrefix indicates the prefix to which a unique number is appended.

-Description <String> Provides a description for the new specification.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-OSCustomizationSpec <OSCustomizationSpec> Specifies an OS customization specification that you want to clone.

-FullName <String> Specifies the administrator's full name. This parameter applies only to Windows operating systems.

-OrgName <String> Specifies the name of the organization to which the administrator belongs.

-ChangeSid	Indicates that the customization should modify the system security identifier (SID). This parameter applies only to Windows operating systems.
-DeleteAccounts	Indicates that you want to delete all user accounts. This parameter applies only to Windows operating systems.

-GuiRunOnce <String[]> Specifies a list of commands. These commands run when a user logs in for the first time after the customization completes. This parameter applies only to Windows operating systems.

-AdminPassword <String> Specifies a new OS administrator's password. This parameter applies only to Windows operating systems.

-TimeZone <String> Specifies the name or ID of the time zone for a Windows guest OS only. Using wildcards is supported. The following time zones are available:

000 Int'l Dateline 001 Samoa 002 Hawaii 003 Alaskan 004 Pacific 010 Mountain (U.S. and Canada) 015 U.S. Mountain: Arizona 020 Central (U.S. and Canada) 025 Canada Central 030 Mexico 033 Central America 035 Eastern (U.S. and Canada) 040 U.S. Eastern: Indiana (East) 045 S.A. Pacific 050 Atlantic (Canada) 055 S.A. Western 056 Pacific S.A. 060 Newfoundland 065 E. South America 070 S.A. Eastern 073 Greenland 075 Mid-Atlantic 080 Azores 083 Cape Verde Islands 085 GMT (Greenwich Mean Time) 090 GMT Greenwich 095 Central Europe 100 Central European 105 Romance 110 W. Europe 113 W. Central Africa 115 E. Europe 120 Egypt 125 EET (Helsinki, Riga, Tallinn) 130 EET (Athens, Istanbul, Minsk) 135 Israel: Jerusalem 140 S. Africa: Harare, Pretoria 145 Russian 150 Arab 155 E. Africa 160 Iran 165 Arabian 170 Caucasus Pacific (U.S. and Canada) 175 Afghanistan 180 Russia Yekaterinburg 185 W. Asia 190 India 193 Nepal 195 Central Asia 200 Sri Lanka 201 N. Central Asia 203 Myanmar: Rangoon 205 S.E. Asia 207 N. Asia 210 China 215 Singapore 220 Taipei 225 W. Australia 227 N. Asia East 230 Korea: Seoul 235 Tokyo 240 Sakha Yakutsk 245 A.U.S. Central: Darwin 250 Central Australia 255 A.U.S. Eastern 260 E. Australia 265 Tasmania 270 Vladivostok 275 W. Pacific 280 Central Pacific 285 Fiji 290 New Zealand 300 Tonga

-AutoLogonCount <Int32> Specifies the number of times the virtual machine automatically logs in as administrator without prompting for user credentials. The valid values are in the range between 0 and Int32.MaxValue. Specifying 0 disables auto log-on. This parameter applies only to Windows operating systems.

-Workgroup <String> Specifies a workgroup. This parameter applies only to Windows operating systems.

-DomainCredentials <PSCredential> Specifies the credentials you want to use for domain authentication. This parameter applies only to Windows operating systems.

-DomainUsername <String> Specifies the user name you want to use for domain authentication. This parameter applies only to Windows operating systems.

-DomainPassword <String> Specifies the password you want to use for domain authentication. This parameter applies only to Windows operating systems.

-ProductKey <String> Specifies the MS product key. If the guest OS version is earlier than Vista, this parameter is required in order to make the customization unattended. For Vista or later, the OS customization is unattended no matter if the ProductKey parameter is set.

-LicenseMode <LicenseMode> Specifies the license mode of the Windows 2000/2003 guest operating system. The valid values are Perseat, Perserver, and Notspecified. If Perserver is set, use the LicenseMaxConnection parameter to define the maximum number of connections. This parameter applies only to Windows operating systems.

-LicenseMaxConnections <Int32> Specifies the maximum connections for server license mode. Use this parameter only if the LicenseMode parameter is set to Perserver. This parameter applies only to Windows operating systems.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-OSCustomizationSpec -Name Spec -OSType Windows -FullName Administrator -OrgName Organization -NamingScheme Fixed -NamingPrefix Computer -ProductKey “xxxx-xxxx” -LicenseMode PerSeat -Workgroup Workgroup -ChangeSid

Generates a new SID for the machine and sets the name of the machine to “Computer”.

————— Example 2 —————

C:PS>New-OSCustomizationSpec -Name Spec -OSType Windows -Description “This spec adds a computer in a domain.” -FullName Administrator -OrgName Organization -NamingScheme Fixed -NamingPrefix “Computer” -ProductKey “xxxx-xxxx” -LicenseMode Perserver -LicenseMaxConnections 30 -AdminPassword pass -Domain Domain -DomainUsername Root -DomainPassword pass

Creates a customization specification that adds a computer in the domain named “Domain”.

REMARKS To see the examples, type: “get-help New-OSCustomizationSpec -examples”. For more information, type: “get-help New-OSCustomizationSpec -detailed”. For technical information, type: “get-help New-OSCustomizationSpec -full”. For online help, type: “get-help New-OSCustomizationSpec -online”

New-ResourcePool

NAME New-ResourcePool

SYNOPSIS This cmdlet creates a new resource pool.

SYNTAX New-ResourcePool -Location <VIContainer> -Name <String> [-CpuExpandableReservation <Boolean>] [-CpuLimitMhz <Int64>] [-CpuReservationMhz <Int64>] [-CpuSharesLevel <SharesLevel>] [-MemExpandableReservation <Boolean>] [-MemLimitMB <Int64>] [-MemLimitGB <Decimal>] [-MemReservationMB <Int64>] [-MemReservationGB <Decimal>] [-MemSharesLevel <SharesLevel>]

[-NumCpuShares <Int32>] [-NumMemShares <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new resource pool with the provided inputs on the location that is specified by the Location parameter.

PARAMETERS

-Location <VIContainer> Specifies a container object (ResourcePool, Cluster, or VMHost) where you want to place the new resource pool. If a host or a cluster is specified for the Location parameter, the resource pool is created in the "Resources" resource pool. The "Resources" resource pool is a system resource pool and is guaranteed to exist.

-Name <String> Specifies a name for the new resource pool.

-CpuExpandableReservation [<Boolean>] Indicates that the CPU reservation can grow beyond the specified value if the parent resource pool has unreserved resources.

-CpuLimitMhz <Int64> Specifies a CPU usage limit in MHz. Utilization will not exceed this limit even if there are available resources.

-CpuReservationMhz <Int64> Specifies the CPU size in MHz that is guaranteed to be available.

-CpuSharesLevel <SharesLevel> Specifies the CPU allocation level for this pool. This property is used in relative allocation between resource consumers. The valid values are Custom, High, Low, and Normal.

-MemExpandableReservation [<Boolean>] If the value is $true, the memory reservation can grow beyond the specified value if the parent resource pool has unreserved resources.

-MemLimitMB <Int64> This parameter is obsolete. Use MemLimitGB instead. Specifies a memory usage limit in megabytes (MB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemLimitGB <Decimal> Specifies a memory usage limit in gigabytes (GB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemReservationMB <Int64> This parameter is obsolete. Use MemReservationGB instead. Specifies the guaranteed available memory in megabytes (MB).

-MemReservationGB <Decimal> Specifies the guaranteed available memory in gigabytes (GB).

-MemSharesLevel <SharesLevel> Specifies the memory allocation level for this pool. This property is used in relative allocation between resource consumers. The valid values are Custom, High, Low, and Normal.

-NumCpuShares <Int32> Specifies the CPU allocation level for this pool. This property is used in relative allocation between resource consumers. This parameter is ignored unless the CpuSharesLevel parameter is set to Custom.

-NumMemShares <Int32> Specifies the memory allocation level for this pool. This property is used in relative allocation between resource consumers. This parameter is ignored unless the MemSharesLevel parameter is set to Custom.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$resourcepool1 = Get-ResourcePool -Location Cluster -Name ResourcePool1

New-ResourcePool -Location $resourcepool1 -Name ResourcePool2 -CpuExpandableReservation $true -CpuReservationMhz 500 -CpuSharesLevel high -MemExpandableReservation $true -MemReservationGB 5 -MemSharesLevel high

Creates a new resource pool named ResourcePool2 in the cluster's root resource pool ResourcePool1.

REMARKS To see the examples, type: "get-help New-ResourcePool -examples". For more information, type: "get-help New-ResourcePool -detailed". For technical information, type: "get-help New-ResourcePool -full". For online help, type: "get-help New-ResourcePool -online"

New-ScsiController

NAME New-ScsiController

SYNOPSIS This cmdlet creates a new SCSI controller.

SYNTAX New-ScsiController [-HardDisk] <HardDisk[]> [[-Type] <ScsiControllerType>] [[-BusSharingMode] <ScsiBusSharingMode>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new SCSI controller.

PARAMETERS

-HardDisk <HardDisk[]> Specifies the hard disks you want to attach to the new SCSI controller.

-Type <ScsiControllerType> Specifies the type of the SCSI controller. The valid values are ParaVirtual, VirtualBusLogic, VirtualLsiLogic, and VirtualLsiLogicSAS.

-BusSharingMode <ScsiBusSharingMode> Specifies the bus sharing mode of the SCSI controller. The valid values are NoSharing, Physical, and Virtual.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vm = Get-VM VM | New-HardDisk -CapacityKB 10485760 | New-ScsiController

Creates a new 10GB hard disk and a new SCSI controller with default values for the BusSharingMode and Type properties.

————— Example 2 —————

C:PS>$disk = Get-HardDisk -VM VM | Select -First 2

New-ScsiController -HardDisk $disk -BusSharingMode Physical -Type VirtualLsiLogicSAS

Creates for the first two hard disks of VM a new SCSI controller of VirtualLsiLogicSAS type and with Physical bus sharing mode.

REMARKS To see the examples, type: "get-help New-ScsiController -examples". For more information, type: "get-help New-ScsiController -detailed". For technical information, type: "get-help New-ScsiController -full". For online help, type: "get-help New-ScsiController -online"

New-Snapshot

NAME New-Snapshot

SYNOPSIS This cmdlet creates a new snapshot of a virtual machine.

SYNTAX New-Snapshot [-VM] <VirtualMachine[]> [-Name] <String> [-Description <String>] [-Memory] [-Quiesce] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new snapshot of a virtual machine with the provided inputs.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines you want to snapshot.

-Name <String> Specifies a name for the new snapshot.

-Description <String> Provide a description of the new snapshot.

-Memory If the value is $true and if the virtual machine is powered on, the virtual machine's memory state is preserved with the snapshot.

-Quiesce If the value is $true and the virtual machine is powered on, VMware Tools are used to quiesce the file system of the virtual machine. This assures that a disk snapshot represents a consistent state of the guest file systems. If the virutal machine is powered off or VMware Tools are not available, the Quiesce parameter is ignored.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>New-Snapshot -VM VM -Name BeforePatch

Creates a new snapshot of the VM virtual machine named BeforePatch.

REMARKS To see the examples, type: "get-help New-Snapshot -examples". For more information, type: "get-help New-Snapshot -detailed". For technical information, type: "get-help New-Snapshot -full". For online help, type: "get-help New-Snapshot -online"

New-StatInterval

NAME New-StatInterval

SYNOPSIS This cmdlet creates a statistics interval with the specified parameters.

SYNTAX New-StatInterval [-Name] <String> [-SamplingPeriodSecs] <Int32> [-StorageTimeSecs] <Int32> [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a statistics interval with the specified parameters.

PARAMETERS

-Name <String> Specifies a name for the new statistics interval.

-SamplingPeriodSecs <Int32> Specifies a sampling period in seconds.

-StorageTimeSecs <Int32> Specifies the length of time (in seconds) that the statistics information is kept in the database.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-StatInterval -Name Minute -SamplingPeriodSecs 60 -StorageTimeSecs 600

Creates a new statistics interval named Minute. Note that creating statistics intervals is allowed only on VirtualCenter 2.0.

REMARKS To see the examples, type: "get-help New-StatInterval -examples". For more information, type: "get-help New-StatInterval -detailed". For technical information, type: "get-help New-StatInterval -full". For online help, type: "get-help New-StatInterval -online"

New-Tag

NAME New-Tag

SYNOPSIS This cmdlet creates a new tag in the specified tag category with the specified parameters.

SYNTAX New-Tag [-Name] <String> [-Category] <TagCategory> [-Description <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new tag in the specified tag category with the specified parameters.

PARAMETERS

-Name <String> Specifies the name of the new tag.

-Category <TagCategory> Specifies the tag category in which the new tag will be created.

-Description <String> Specifies the description of the new tag.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-TagCategory -Name "MyTagCategory" | New-Tag -Name "MyTag" -Description "Create MyTag tag in MyTagCategory category."

Retrieves a specific tag category, named "MyTagCategory", creates a tag named "MyTag" inside it, and sets the tag description.

REMARKS To see the examples, type: "get-help New-Tag -examples". For more information, type: "get-help New-Tag -detailed". For technical information, type: "get-help New-Tag -full". For online help, type: "get-help New-Tag -online"

New-TagAssignment

NAME New-TagAssignment

SYNOPSIS This cmdlet assigns the specified tag to the specified entity.

SYNTAX New-TagAssignment [-Tag] <Tag> [-Entity] <VIObjectCore> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet assigns the specified tag to the specified entity.

PARAMETERS

-Tag <Tag> Specifies the tag to be assigned to the entity.

-Entity <VIObjectCore> Specifies the object on which to assign the specified tag.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myTag = Get-Tag "MyTag" Get-VM "*MyVM*" | New-TagAssignment -Tag $myTag

Assigns the "MyTag" tag to all virtual machines whose name contains the "*MyVM*" wildcard pattern.

REMARKS To see the examples, type: "get-help New-TagAssignment -examples". For more information, type: "get-help New-TagAssignment -detailed". For technical information, type: "get-help New-TagAssignment -full". For online help, type: "get-help New-TagAssignment -online"

New-TagCategory

NAME New-TagCategory

SYNOPSIS This cmdlet creates a new tag category on the specified vCenter Server systems with the specified parameters.

SYNTAX New-TagCategory [-Name] <String> [-Description <String>] [-Cardinality <Cardinality>] [-EntityType <String[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new tag category on the specified vCenter Server systems with the specified parameters.

PARAMETERS

-Name <String> Specifies the name of the new tag category.

-Description <String> Specifies the description of the new tag category.

-Cardinality <Cardinality> Specifies the cardinality of the tag category. If not specified, the default value is "Single".

"Single" means that only a single tag from this category can be assigned to a specific object at a time. "Multiple" means that more than one tag from this category can be assigned to a specific object at a time.

-EntityType <String[]> Defines the type of objects to which the tags in this category will be applicable. If you do not specify this parameter or specify "All" as a value, the tags in this category will be applicable to all valid entity types.

This parameter accepts both PowerCLI type names and vSphere API type names. The valid PowerCLI type names are: Cluster, Datacenter, Datastore, DatastoreCluster, DistributedPortGroup, DistributedSwitch, Folder, ResourcePool, VApp, VirtualPortGroup, VirtualMachine, VM, VMHost.

For vSphere API type names that are not vCenter Server API type names, a namespace prefix is used. The format is: <namespace>/<type> Example: 'vco/WorkflowItem'

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-TagCategory -Name "MyTagCategory" -Cardinality "Single" -EntityType "VirtualMachine"

Creates a new tag category, named "MyTagCategory", defining the "VirtualMachine" type as applicable to the tags inside that category and specifying that only a single tag from that category can be assigned to a specific VirtualMachine object at a time.

————— Example 2 —————

C:PS>New-TagCategory -Name "MyTagCategory" -Cardinality "Multiple" -Description "MyTagCategory description"

Creates a new tag category, named "MyTagCategory", that has "MyTagCategory description" set as a description and specifies that multiple tags from that category can be assigned to an object. Tags from this category are applicable to all valid entity types.

REMARKS To see the examples, type: "get-help New-TagCategory -examples". For more information, type: "get-help New-TagCategory -detailed". For technical information, type: "get-help New-TagCategory -full". For online help, type: "get-help New-TagCategory -online"

New-Template

NAME New-Template

SYNOPSIS This cmdlet creates a new virtual machine template.

SYNTAX New-Template [-VM] <VirtualMachine> [-Name] <String> [-Location] <VIContainer> [-Datastore <StorageResource>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

New-Template [[-Name] <String>] [-Location <VIContainer>] [-VMHost <VMHost>] [-Datastore <StorageResource>] [-DiskStorageFormat <VirtualDiskStorageFormat>] -Template <Template> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

New-Template [[-Name] <String>] [-Location <VIContainer>] -VMHost <VMHost> [-TemplateFilePath] <String> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new template based on the specified virtual machine. You can also create a new template by cloning an existing one. You can also register an existing template to the vCenter Server inventory.

PARAMETERS

-VM <VirtualMachine> Specifies the virtual machine from which you want to create the new template.

-Name <String> Specifies a name for the new template.

-Location <VIContainer> Specifies the location where you want to place the new template.

-Datastore <StorageResource> Specifies the datastore or the datastore cluster where you want to store the new template.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-VMHost <VMHost> Specifies the host where you want to create the new template.

-DiskStorageFormat <VirtualDiskStorageFormat> Specifies the disk storage format of the new template. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-Template <Template> Specifies a template you want to clone.

-TemplateFilePath <String> Specifies the datastore path to the file you want to use to register the new template.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

——— Example 1 ———

C:PS>$myVM = Get-VM -Name "MyVM1" $drsCluster=Get-DatastoreCluster "MyDatastoreCluster" New-Template -VM $myVM -Name "MyTemplate" -Datastore $drsCluster -Location Datacenter2

Creates a template named MyTemplate from the MyVM1 virtual machine and stores it in the MyDatastoreCluster datastore cluster in the Datacenter2 datacenter.

——— Example 2 ———

C:PS>$myFolder = Get-Folder -Name "MyFolder1" New-Template -Name "MyTemplate1" -TemplateFilePath "[Storage1] templatefolder/template.vmtx" -Location $myFolder -VMHost (Get-VMHost)

Registers the existing MyTemplate1 template to a vCenter Server inventory folder by using the specified template file.

——— Example 3 ———

C:PS>$myTemplate = Get-Template -Name "MyTemplate1" $myDs = Get-Datastore -Name "MyDatastore1" New-Template -Template $myTemplate -Name "MyTemplate2" -Datastore $myDs -Location "Datacenter2"

Creates the MyTemplate2 template by cloning an existing template and stores the new template in the specified datastore in the Datacenter2 datacenter.

REMARKS To see the examples, type: "get-help New-Template -examples". For more information, type: "get-help New-Template -detailed". For technical information, type: "get-help New-Template -full". For online help, type: "get-help New-Template -online"

New-VApp

NAME New-VApp

SYNOPSIS This cmdlet creates a new vApp.

SYNTAX New-VApp -Location <VIContainer> [-Name] <String> [-CpuExpandableReservation <Boolean>] [-CpuLimitMhz <Int64>] [-CpuReservationMhz <Int64>] [-CpuSharesLevel <SharesLevel>] [-MemExpandableReservation <Boolean>] [-MemLimitMB <Int64>] [-MemLimitGB <Decimal>] [-MemReservationMB <Int64>] [-MemReservationGB <Decimal>] [-MemSharesLevel <SharesLevel>] [-NumCpuShares <Int32>] [-NumMemShares <Int32>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VApp -Location <VIContainer> [[-Name] <String>] -VApp <VApp> [-VMHost <VMHost>] [-Datastore <Datastore>] [-DiskStorageFormat <VirtualDiskStorageFormat>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VApp -Location <VIContainer> [[-Name] <String>] [-Datastore <Datastore>] -ContentLibraryItem <ContentLibraryItem> [-DiskStorageFormat <VirtualDiskStorageFormat>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new vApp.

PARAMETERS

-Location <VIContainer> Specifies a VApp, ResourcePool, VMHost, or Cluster object where you want to place the new vApp.

-Name <String> Specifies a name for the new vApp.

-CpuExpandableReservation [<Boolean>] Indicates that the CPU reservation can grow beyond the specified value if there are available resources.

-CpuLimitMhz <Int64> Specifies a CPU usage limit in MHz. Utilization will not exceed this limit even if there are available resources.

-CpuReservationMhz <Int64> Specifies the CPU size in MHz that is guaranteed to be available.

-CpuSharesLevel <SharesLevel> Specifies the CPU allocation level for this vApp. This property is used in relative allocation between resource consumers. The valid values are Custom, High, Low, and Normal.

-MemExpandableReservation [<Boolean>] If the value is $true, the memory reservation can grow beyond the specified value if there are available resources.

-MemLimitMB <Int64> This parameter is obsolete. Use MemLimitGB instead. Specifies a memory usage limit in megabytes (MB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemLimitGB <Decimal> Specifies a memory usage limit in gigabytes (GB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemReservationMB <Int64> This parameter is obsolete. Use MemReservationGB instead. Specifies the guaranteed available memory in megabytes (MB).

-MemReservationGB <Decimal> Specifies the guaranteed available memory in gigabytes (GB).

-MemSharesLevel <SharesLevel> Specifies the memory allocation level for this vApp. This property is used in relative allocation between resource consumers. The valid values are Custom, High, Low, and Normal.

-NumCpuShares <Int32> Specifies the CPU allocation level for this vApp. This property is used in relative allocation between resource consumers. This parameter is ignored unless the CpuSharesLevel parameter is set to Custom.

-NumMemShares <Int32> Specifies the memory allocation level for this vApp. This property is used in relative allocation between resource consumers. This parameter is ignored unless the MemSharesLevel parameter is set to Custom.

-Server <VIServer[]> Specifies the Center Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-VApp <VApp> Specifies a vApp you want to copy.

-VMHost <VMHost> Specifies the host where you want to run the copied vApp.

-Datastore <Datastore> Specifies the datastore where you want to store the copied vApp. If you do not specify a datastore, the cmdlet takes the first datastore of the host or cluster.

-DiskStorageFormat <VirtualDiskStorageFormat> Specifies the storage format of the disks of the vApp.

-ContentLibraryItem <ContentLibraryItem> Specifies the content library template to deploy the vApp from.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-VApp -Name MyVApp1 -CpuLimitMhz 4000 -CpuReservationMhz 1000 -Location MyVMHost1

Creates a new vApp on the MyVMHost1 host.

REMARKS To see the examples, type: "get-help New-VApp -examples". For more information, type: "get-help New-VApp -detailed". For technical information, type: "get-help New-VApp -full". For online help, type: "get-help New-VApp -online"

New-VDPortgroup

NAME New-VDPortgroup

SYNOPSIS This cmdlet creates distributed port groups.

SYNTAX New-VDPortgroup [-VDSwitch] <VDSwitch> -Name <String> [-Notes <String>] [-NumPorts <Int32>] [-VlanId <Int32>] [-VlanTrunkRange <VlanRangeList>] [-PortBinding <DistributedPortGroupPortBinding>] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VDPortgroup [-VDSwitch] <VDSwitch> [-Name <String>] -ReferencePortgroup <VDPortgroup> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VDPortgroup [-VDSwitch] <VDSwitch> [-Name <String>] -BackupPath <String> [-KeepIdentifiers] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates distributed port groups. You can create a new distributed port group with custom properties, specify a reference port group to clone its properties, or provide a backup profile to import the port group configuration.

PARAMETERS

-VDSwitch <VDSwitch> Specifies the vSphere distributed switch on which you want to create the new distributed port group.

-Name <String> Specifies the name of the new distributed port group that you want to create.

-Notes <String> Specifies a description for the new distributed port group that you want to create.

-NumPorts <Int32> Specifies the number of ports that the distributed port group will have. If you do not set this parameter, the number of ports for the new distributed port group is set to 128 ports.

-VlanId <Int32> Specifies the VLAN ID of the distributed port group that you want to create. Valid values are integers in the range of 1 to 4094.

-VlanTrunkRange <VlanRangeList> Specifies the VLAN trunk range for the distributed port group that you want to create. Valid values are strings representing ranges of IDs. For example, "1-4, 6, 8-9".

-PortBinding <DistributedPortGroupPortBinding> Specifies the port binding setting for the distributed port group that you want to create. This parameter accepts Static, Dynamic, and Ephemeral values. Note: Dynamic port binding is deprecated. For better performance, static port binding is recommended.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-ReferencePortgroup <VDPortgroup> Specifies a reference distributed port group. The properties of the new distributed port group will be cloned from the reference distributed port group.

Note: This parameter is supported only on vSphere 5.1 and later.

-BackupPath <String> Specifies the full file path to the .zip file containing the backup configuration that you want to import. Only .zip files created with the Export-VDPortgroup cmdlet are supported.

Note: This parameter is supported only on vSphere 5.1 and later.

-KeepIdentifiers	Indicates that the original vSphere distributed port group identifiers will be preserved. Note: This parameter is supported only on vSphere 5.1 and later.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch -Name "MyVDSwitch" | New-VDPortgroup -Name "MyVDPortGroup" -NumPorts 8 -VLanId 4

Creates a new distributed port group on the specified vSphere distributed switch with the specified number of ports and VLAN ID.

————— Example 2 —————

C:PS>$myReferncePortroup = Get-VDPortgroup -Name "MyReferencePortGroup" Get-VDSwitch -Name "MyVDSwitch" | New-VDPortgroup -Name "MyVDPortGroup" -ReferencePortgroup $myReferncePortroup

Creates a new distributed port group on the specified vSphere distributed switch by cloning the configuration of the distributed port group named "MyReferencePortGroup".

————— Example 3 —————

C:PS>Get-VDSwitch -Name "MyVDSwitch" | New-VDPortgroup -Name "MyVDPortGroup" -RunAsync

Creates asynchronously a new distributed port group on the specified vSphere distributed switch.

————— Example 4 —————

C:PS>$myBackupFilePath = 'c:Backup.zip' Get-VDSwitch -Name "MyVDSwitch" | New-VDPortgroup -Name "MyVDPortgroup" -BakupPath $myBackupFilePath

Creates a new distributed port group on the specified vSphere distributed switch by importing the specified backup profile.

————— Example 5 —————

C:PS>Get-VDSwitch -Name "MyVDSwitch" | New-VDPortgroup -Name "MyVDPortGroup" -VlanTrunkRange "1-5, 10-20"

Creates a new distributed port group on the specified vSphere distributed switch with the specified name and VLAN trunk range settings.

REMARKS To see the examples, type: "get-help New-VDPortgroup -examples". For more information, type: "get-help New-VDPortgroup -detailed". For technical information, type: "get-help New-VDPortgroup -full". For online help, type: "get-help New-VDPortgroup -online"

New-VDSwitch

NAME New-VDSwitch

SYNOPSIS This cmdlet creates vSphere distributed switches.

SYNTAX New-VDSwitch [-ContactDetails <String>] [-ContactName <String>] [-LinkDiscoveryProtocol <LinkDiscoveryProtocol>] [-LinkDiscoveryProtocolOperation <LinkDiscoveryOperation>] [-MaxPorts <Int32>] [-Mtu <Int32>] [-Notes <String>] [-NumUplinkPorts <Int32>] [-Version <String>] -Name <String> -Location <VIContainer> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VDSwitch -ReferenceVDSwitch <VDSwitch> -Name <String> -Location <VIContainer> [-WithoutPortGroups] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VDSwitch -BackupPath <String> [-KeepIdentifiers] [-Name <String>] -Location <VIContainer> [-WithoutPortGroups] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates vSphere distributed switches. You can create a new vSphere distributed switch with custom properties, specify a reference vSphere distributed switch to clone its configuration, or provide a backup profile to import the switch configuration.

Note: Creating vSphere distributed switches from a reference switch or a backup profile requires vCenter Server 5.1 or later.

PARAMETERS

-ContactDetails <String> Specifies the contact details of the vSphere distributed switch administrator.

-ContactName <String> Specifies the name of the vSphere distributed switch administrator.

-LinkDiscoveryProtocol <LinkDiscoveryProtocol> Specifies the discovery protocol type of the vSphere distributed switch that you want to create. This parameter accepts CDP and LLDP values. If you do not set a value for this parameter, the default server setting is used.

-LinkDiscoveryProtocolOperation <LinkDiscoveryOperation> Specifies the link discovery protocol operation for the vSphere distributed switch that you want to create. This parameter accepts Advertise, Listen, Both, and Disabled values. If you do not set a value for this parameter, the default server setting is used.

-MaxPorts <Int32> Specifies the maximum number of ports allowed on the vSphere distributed switch that you want to create.

-Mtu <Int32> Specifies the maximum MTU size for the vSphere distributed switch that you want to create. Valid values are positive integers only.

-Notes <String> Specifies a description for the vSphere distributed switch that you want to create.

-NumUplinkPorts <Int32> Specifies the number of uplink ports on the vSphere distributed switch that you want to create.

-Version <String> Specifies the version of the vSphere distributed switch that you want to create. This parameter accepts 4.0, 4.1.0, 5.0.0, 5.1.0, 5.5.0, and 6.0.0 values. You cannot specify a version that is incompatible with the version of the vCenter Server system you are connected to.

-Name <String> Specifies a name for the new vSphere distributed switch that you want to create. You cannot specify this parameter, when the KeepIdentifiers parameter is specified.

-Location <VIContainer> Specifies the location where you want to create the vSphere distributed switch. This parameter accepts Datacenter and Folder objects.

Note: You cannot create distributed port groups with identical names in the same location. If you want to import or clone a vSphere distributed switch with its distributed port groups, you need to specify a location that does not contain identically named distributed port groups.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-ReferenceVDSwitch <VDSwitch> Specifies a reference vSphere distributed switch. The properties of the new vSphere distributed switch will be cloned from the reference vSphere distributed switch.

Note: This parameter is supported only on vSphere 5.1 and later.

-WithoutPortGroups Indicates that the new vSphere distributed switch will be created without importing the port groups from the specified backup file or reference vSphere distributed switch.

-BackupPath <String> Specifies the full file path to the .zip file containing the backup configuration that you want to import. Only .zip files created with the Export-VDSwitch cmdlet are supported.

Note: This parameter is supported only on vSphere 5.1 and later.

-KeepIdentifiers Indicates that the original vSphere distributed switch and port group identifiers will be preserved. You cannot specify this parameter, when the Name parameter is specified.

Note: This parameter is supported only on vSphere 5.1 and later.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myDatacenter = Get-Datacenter -Name "MyDatacenter" New-VDSwitch -Name "MyVDSwitch" -Location $myDatacenter -LinkDiscoveryProtocol "LLDP" -LinkDiscoveryProtocolOperation "Listen" -MaxPorts 256 -Version "5.0.0"

Creates a new vSphere distributed switch with the specified name, version, maximum number of ports, and link discovery protocol settings in the specified datacenter.

————— Example 2 —————

C:PS>$myFolder = Get-Folder -Name "MyFolder" Get-VDSwitch -Name "MyReferenceSwitch" | New-VDSwitch -Name "MyVDSwitch" -Location $myFolder -WithoutPortGroups

Creates a new vSphere distributed switch by cloning the configuration of the existing vSphere distributed switch named "MyReferenceSwitch". The new vSphere distributed switch is created without cloning the existing port groups and is stored in the specified folder.

————— Example 3 —————

C:PS>$myFolder = Get-Folder -Name "MyFolder" New-VDSwitch -Name "MyVDSwitch" -Location $myFolder -WithoutPortGroups -BackupPath "c:MyDistributedSwitchProfile.zip"

Creates a new vSphere distributed switch by importing the specified backup profile.

REMARKS To see the examples, type: "get-help New-VDSwitch -examples". For more information, type: "get-help New-VDSwitch -detailed". For technical information, type: "get-help New-VDSwitch -full". For online help, type: "get-help New-VDSwitch -online"

New-VDSwitchPrivateVlan

NAME New-VDSwitchPrivateVlan

SYNOPSIS This cmdlet creates private VLAN configuration entries on a vSphere distributed switch.

SYNTAX New-VDSwitchPrivateVlan [-VDSwitch] <VDSwitch> -PrimaryVlanId <Int32> -SecondaryVlanId <Int32> -PrivateVlanType <PrivateVlanType> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates private VLAN configuration entries on a vSphere distributed switch.

PARAMETERS

-VDSwitch <VDSwitch> Specifies the vSphere distributed switch on which to create a private VLAN configuration.

-PrimaryVlanId <Int32> Specifies the primary VLAN ID. The VLAN IDs of 0 and 4095 are reserved and cannot be used.

-SecondaryVlanId <Int32> Specifies the secondary VLAN ID. The VLAN IDs of 0 and 4095 are reserved and cannot be used.

-PrivateVlanType <PrivateVlanType> Specifies the private VLAN port type: community, isolated, or promiscuous.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch "MyVDSwitch" | New-VDSwitchPrivateVlan -PrimaryVlanId 1 -SecondaryVlanId 1 -PrivateVlanType Promiscuous

Creates a private VLAN inside a specific vSphere distributed switch with a promiscuous VLAN port type.

REMARKS To see the examples, type: "get-help New-VDSwitchPrivateVlan -examples". For more information, type: "get-help New-VDSwitchPrivateVlan -detailed". For technical information, type: "get-help New-VDSwitchPrivateVlan -full". For online help, type: "get-help New-VDSwitchPrivateVlan -online"

New-VIPermission

NAME New-VIPermission

SYNOPSIS This cmdlet creates new permissions on the specified inventory objects for the provided users and groups in the role.

SYNTAX New-VIPermission [-Entity] <VIObject[]> [-Principal] <VIAccount[]> [-Role] <Role> [-Propagate <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates new permissions on the specified inventory objects for the provided users and groups in the role. By default, new permissions are propagated down the hierarchy to sub-entities. You cannot create new permissions for the following objects: - direct child folders of a datacenter - root resource pools of clusters and standalone hosts. These objects always inherit the permissions of their parent.

PARAMETERS

-Entity <VIObject[]> Specifies the inventory objects for which you want to create new permissions.

-Principal <VIAccount[]> Specifies users and groups to which you want to apply the new permissions. If you specify principal names by using the "domainname" syntax, wildcards are not supported.

-Role <Role> Specifies the roles for which you want to create new permissions.

-Propagate [<Boolean>] Indicates that you want to propagate the new permissions to the child inventory objects.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-VIRole -Name Role -Server $server -Privilege (Get-VIPrivilege -PrivilegeGroup)

$permission = New-VIPermission -Role Role -Principal Administrator -Entity (Get-Datacenter)

Creates a permission on the provided server for a role with the specified privileges.

REMARKS To see the examples, type: “get-help New-VIPermission -examples”. For more information, type: “get-help New-VIPermission -detailed”. For technical information, type: “get-help New-VIPermission -full”. For online help, type: “get-help New-VIPermission -online”

New-VIProperty

NAME New-VIProperty

SYNOPSIS This cmdlet creates a new extension property on the specified object type.

SYNTAX New-VIProperty [-Name] <String> [-ObjectType] <String[]> [-Value] <ScriptBlock> [-Force] [-BasedOnExtensionProperty <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VIProperty [-Name] <String> [-ObjectType] <String[]> [-Force] [-ValueFromExtensionProperty] <String> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new extension property on the specified object type. Changes take effect upon the next retrieval of the corresponding objects.

PARAMETERS

-Name <String> Specifies a name for the new extension property. Names are case-sensitive and can include only letters, numbers, and the underscore symbol. The name of a property must start with a letter or underscore.

-ObjectType <String[]> Specifies the object types to which you want to append the new property. All PowerCLI object types are supported.

-Value <ScriptBlock> Specifies a script block you want to use to compute the value of the new extended property.

-Force Indicates that you want to create the new property even if another property with the same name exists for the specified object type. This parameter is not applicable for core properties of an object type.

-BasedOnExtensionProperty <String[]> Specifies a list of strings that maps the properties of the $this.ExtensionData object. Use this parameter to specify which members of ExtensionData are used by the script block provided for the Value parameter. This parameter is case-sensitive.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-ValueFromExtensionProperty <String> Specifies a string that maps a property of the $this.ExtensionData object. This parameter is case-sensitive.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-VIProperty -ObjectType VirtualMachine -Name CommittedSpaceMB -Value { $vm = $args[0]; $sum = 0; $vm.ExtensionData.Storage.PerDatastoreUsage | foreach { $sum += $_.Committed} ; $sum = [int]($sum / 1024 / 1024); return $sum }

Get-VM | select Name, CommittedSpaceMB

Creates a script-based property for the VirtualMachine object type that calculates the committed space of a virtual machine.

————— Example 2 —————

C:PS>New-VIProperty -ObjectType VirtualMachine -Name CommittedSpaceMB -Value { $vm = $args[0]; $sum = 0; $vm.ExtensionData.Storage.PerDatastoreUsage | foreach { $sum += $_.Committed} ; $sum = [int]($sum / 1024 / 1024); return $sum } -BasedOnExtensionProperty 'Storage.PerDatastoreUsage.Committed' -Force

Get-VM | select Name, CommittedSpaceMB

Creates a property that calculates the committed space of a virtual machine. The cmdlet uses the BasedOnExtensionProperty parameter to specify which ExtensionData member is used by the script block. This mean that during the creation of each virtual machine, only the specified property of extension data - Storage.PerDatastoreUsage.Committed will be filled up.

————— Example 3 —————

C:PS>New-VIProperty -ObjectType VirtualMachine -Name CommittedSpace -ValueFromExtensionProperty 'SUM Storage.PerDatastoreUsage.Committed'

Creates a new property that calculates the committed storage based on the property and aggregation function SUM specified by the ValueFromExtensionProperty parameter.

————— Example 4 —————

C:PS>New-VIProperty -ObjectType InventoryItem -Name OverallStatus -ValueFromExtensionProperty 'OverallStatus'

Get-VM | select Name, OverallStatus

Get-VMHost | select Name, OverallStatus

Creates a new property based on the OverallStatus property for all inventory types.

————— Example 5 —————

C:PS>New-VIProperty -ObjectType VIObjectCore -Value { if ($args[0].UId -match "/VIserver=[w]+@(.*):.*") { $matches[1] } else { " } } -Name VIServerName

Get-Inventory | select Name, VIServerName

Creates a script-based property to VIObjectCore that parses the UId property and extracts the name of the server to which a given object belongs.

REMARKS To see the examples, type: "get-help New-VIProperty -examples". For more information, type: "get-help New-VIProperty -detailed". For technical information, type: "get-help New-VIProperty -full". For online help, type: "get-help New-VIProperty -online"

New-VIRole

NAME New-VIRole

SYNOPSIS This cmdlet creates a new role on the specified servers and applies the provided privileges.

SYNTAX New-VIRole [-Name] <String> [[-Privilege] <Privilege[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new role on the specified servers and applies the provided privileges.

PARAMETERS

-Name <String> Specifies a name for the new role.

-Privilege <Privilege[]> Specifies the privileges you want to apply to the new role.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-VIRole -Name Role -Privilege (Get-VIPrivilege -PrivilegeGroup)

Creates a new role with the provided group privileges.

REMARKS To see the examples, type: "get-help New-VIRole -examples". For more information, type: "get-help New-VIRole -detailed". For technical information, type: "get-help New-VIRole -full". For online help, type: "get-help New-VIRole -online"

New-VirtualPortGroup

NAME New-VirtualPortGroup

SYNOPSIS This cmdlet creates a new port group on the specified host.

SYNTAX New-VirtualPortGroup [-Name] <String> [-VirtualSwitch] <VirtualSwitch> [-VLanId <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new port group on the host using the provided parameters.

PARAMETERS

-Name <String> Specifies a name for the new port group.

-VirtualSwitch <VirtualSwitch> Specifies the virtual switch for which you want to create a new port group.

-VLanId <Int32> Specifies the VLAN ID for ports using this port group. The following values are valid:

0 - specifies that you do not want to associate the port group with a VLAN. 1 to 4094 - specifies a VLAN ID for the port group. 4095 - specifies that the port group should use trunk mode, which allows the guest operating system to manage its own VLAN tags.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.114.234 -Name VSwitch

$vportgroup = New-VirtualPortGroup -VirtualSwitch $vswitch -Name VPortGroup

Creates a virtual port group named VPortGroup on the virtual switch VSwitch.

REMARKS To see the examples, type: "get-help New-VirtualPortGroup -examples". For more information, type: "get-help New-VirtualPortGroup -detailed". For technical information, type: "get-help New-VirtualPortGroup -full". For online help, type: "get-help New-VirtualPortGroup -online"

New-VirtualSwitch

NAME New-VirtualSwitch

SYNOPSIS This cmdlet creates a new virtual switch.

SYNTAX New-VirtualSwitch [-VMHost] <VMHost> [-Name] <String> [[-NumPorts] <Int32>] [[-Nic] <PhysicalNic[]>] [[-Mtu] <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new virtual switch on the host that is specified by the VMHost parameter.

PARAMETERS

-VMHost <VMHost> Specifies the host on which you want to create the new virtual switch.

-Name <String> Specifies a name for the new virtual switch.

-NumPorts <Int32> Specifies the virtual switch port number. The value is rounded to the closest exact power of two that is greater than the given number (for example, if the user specifies 67, this number is rounded to 128). Note that the port number displayed in the vSphere Client might differ from the value that you specified for the NumPorts parameter.

Note: In ESX 5.5 or later, standard virtual switches are always elastic, so the NumPorts parameter is no longer applicable and its value is ignored.

-Nic <PhysicalNic[]> Specifies the physical network interface cards you want to add to the Active NICs of the new virtual switch. This parameter accepts both objects and strings.

-Mtu <Int32> Specifies the maximum transmission unit (MTU) associated with the specified virtual switch (in bytes). The MTU value is always greater than 0.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.112.234 -Name VSwitch

Creates a new virtual switch named VSwitch on the virtual machine host with IP address 10.23.112.234.

———— Example 2 ————

C:PS>$network = Get-VMHostNetwork -VMHost 10.23.112.234

$phNic = $network.PhysicalNic[0].DeviceName

$vswitch = New-VirtualSwitch -VMHost 10.23.112.234 -Name VSwitch -Nic $phNic

Creates a new virtual switch named VSwitch on the virtual machine host with IP address 10.23.112.234 with a physical network adapter.

———— Example 3 ————

C:PS>Get-VMHost *.128 | New-VirtualSwitch -Name VSwitch -Nic vmnic5,vmnic6

Create a virtual switch named VSwitch with two physical network adapters - 'vmnic5' and 'vmnic6'. Note that the 'vmnic5' and 'vmnic6' adapters must not be assigned to other virtual switches.

REMARKS To see the examples, type: "get-help New-VirtualSwitch -examples". For more information, type: "get-help New-VirtualSwitch -detailed". For technical information, type: "get-help New-VirtualSwitch -full". For online help, type: "get-help New-VirtualSwitch -online"

New-VM

NAME New-VM

SYNOPSIS This cmdlet creates a new virtual machine.

SYNTAX New-VM [-AdvancedOption <AdvancedOption[]>] [[-VMHost] <VMHost>] [-Version <VMVersion>] -Name <String> [-ResourcePool <VIContainer>] [-VApp <VApp>] [-Location <Folder>] [-Datastore <StorageResource>] [-DiskMB <Int64[]>] [-DiskGB <Decimal[]>] [-DiskPath <String[]>] [-DiskStorageFormat <VirtualDiskStorageFormat>] [-MemoryMB <Int64>] [-MemoryGB <Decimal>] [-NumCpu <Int32>] [-Floppy] [-CD] [-GuestId <String>] [-AlternateGuestName <String>] [-NetworkName <String[]>] [-Portgroup <VirtualPortGroupBase[]>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] [-VMSwapfilePolicy <VMSwapfilePolicy>] [-Server <VIServer[]>] [-RunAsync] [-Notes <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VM [-AdvancedOption <AdvancedOption[]>] [[-VMHost] <VMHost>] [-Name <String>] [-ResourcePool <VIContainer>] [-VApp <VApp>] [-Location <Folder>] [-Datastore <StorageResource>] [-DiskStorageFormat <VirtualDiskStorageFormat>] [-OSCustomizationSpec <OSCustomizationSpec>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] [-LinkedClone] [-ReferenceSnapshot <Snapshot>] [-Server <VIServer[]>] [-RunAsync] [-Notes <String>] -VM <VirtualMachine[]> [-WhatIf] [-Confirm] [<CommonParameters>]

New-VM [-AdvancedOption <AdvancedOption[]>] [[-VMHost] <VMHost>] -Name <String> [-ResourcePool <VIContainer>] [-VApp <VApp>] [-Location <Folder>] [-Datastore <StorageResource>] [-Template] <Template> [-DiskStorageFormat <VirtualDiskStorageFormat>] [-OSCustomizationSpec <OSCustomizationSpec>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VM [[-VMHost] <VMHost>] [-Name <String>] [-ResourcePool <VIContainer>] [-VApp <VApp>] [-Location <Folder>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] -VMFilePath <String> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VM [[-VMHost] <VMHost>] [-Name <String>] [-ResourcePool <VIContainer>] [-Location <Folder>] [-Datastore <StorageResource>] [-DiskStorageFormat <VirtualDiskStorageFormat>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] [-ContentLibraryItem] <ContentLibraryItem> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new virtual machine with the provided parameters. The network adapter and the SCSI adapter of the new virtual machine are created of the recommended type for the OS that is specified by the GuestId parameter. If the OSCustomizationSpec parameter is used, the virtual machine is customized according to the spec. You must specify values for at least one of the ResourcePool, VMHost, and VApp parameters.

PARAMETERS

-AdvancedOption <AdvancedOption[]> Specifies advanced options for creating virtual machines. Accepts only SdrsVMDiskAntiAffinityRule and SdrsVMAntiAffinityRule objects.

The SdrsVMDiskAntiAffinityRule defines a Storage DRS intra-VM anti-affinity rule (vm disk anti-affinity rule). It is only applicable when creating a virtual machine or hard disk on a datastore cluster. An instance of the object is created by invoking its constructor. There are two constructors - “public SdrsVMDiskAntiAffinityRule(param string[] diskIdentifier)” and “public SdrsVMDiskAntiAffinityRule(param HardDisk[] disk)”. For the first constructor, “diskIdentifier” can be either the disk key or the index of the disk in the disk array. The specified disks (and the disk to which the rule is applied) are placed in an anti-affinity rule on a DatastoreCluster. Only one such rule is supported per a virtual machine. You can pass the instance to the AdvancedOption parameter of the New-VM or New-HardDisk cmdlets.

The SdrsVMAntiAffinityRule defines a Storage DRS inter-VM anti-affinity rule. It is only applicable when creating a virtual machine on a DatastoreCluster. An instance of the object is created by invoking its constructor. The constructor has one parameter - an array of virtual machines - “public SdrsVMAntiAffinityRule(param VirtualMachine[] vm)”. Then, you can pass the instance to the AdvancedOption parameter of the New-VM cmdlet. This will place the new virtual machine and the virtual machines specified in the constructor in an inter-VM anti-affinity rule on a DatastoreCluster.

The SDRS functionality is experimental.

-VMHost <VMHost> Specifies the host on which you want to create the new virtual machine.

-Version <VMVersion> Specifies the version of the new virtual machine. The valid values are v4, v7, v8, v9, v10, and v11. By default, the new virtual machine is created with the latest available version.

-Name <String> Specifies a name for the new virtual machine. If you are registering or cloning an existing virtual machine, this parameter is not mandatory.

-ResourcePool <VIContainer> Specifies where you want to place the new virtual machine. The parameter accepts VMHost, Cluster, ResourcePool, and VApp objects. If no value is specified, the virtual machine is added to the resource pool of its host.

-VApp <VApp> This parameter is deprecated. Use the ResourcePool parameter instead. Specifies the vApp where you want to create the new virtual machine.

-Location <Folder> Specifies the folder where you want to place the new virtual machine.

-Datastore <StorageResource> Specifies the datastore where you want to place the new virtual machine. If a DatastoreCluster is passed to the Datastore parameter, the virtual machine is placed in the DatastoreCluster in an automated SDRS mode and with enabled intra-VM affinity rule (unless another rule is specified). You can specify a SDRS rule when creating the virtual machine in a DatastoreCluster by passing either a SdrsVMDiskAntiAffinityRule or SdrsVMAntiAffinityRule objects to the AdvancedOption parameter. These two rules are mutually exclusive.

-DiskMB <Int64[]> This parameter is obsolete. Use DiskGB instead. Specifies the size in megabytes (MB) of the disks that you want to create and add to the new virtual machine.

-DiskGB <Decimal[]> Specifies the size in gigabytes (GB) of the disks that you want to create and add to the new virtual machine.

-DiskPath <String[]> Specifies paths to virtual disks you want to add to the new virtual machine.

-DiskStorageFormat <VirtualDiskStorageFormat> Specifies the storage format of the disks of the virtual machine. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-MemoryMB <Int64> This parameter is obsolete. Use MemoryGB instead. Specifies the memory size in megabytes (MB) of the new virtual machine.

-MemoryGB <Decimal> Specifies the memory size in gigabytes (GB) of the new virtual machine.

-NumCpu <Int32> Specifies the number of the virtual CPUs of the new virtual machine.

-Floppy	Indicates that you want to add a floppy drive to the new virtual machine.
-CD	Indicates that you want to add a CD drive to the new virtual machine.

-GuestId <String> Specifies the guest operating system of the new virtual machine. The valid values for specific ESX versions are listed in the description of the VirtualMachineGuestOsIdentifier enumeration type in the vSphere API Reference available at http://www.vmware.com/support/developer/vc-sdk/. Depending on the hardware configuration of the host, some of the guest operating systems might be inapplicable.

-AlternateGuestName <String> Specifies the full OS name of the new virtual machine. Use this parameter if the GuestID parameter is set to otherGuest or otherGuest64.

-NetworkName <String[]> Specifies the networks to which you want to connect the new virtual machine. Specifying a distributed port group name is obsolete. Use the Portgroup parameter instead.

-Portgroup <VirtualPortGroupBase[]> Specifies standard or distributed port groups to which you want to connect the virtual machine. For each specified port group, a new network adapter is created.

-HARestartPriority <HARestartPriority> Specifies the HA restart priority of the new virtual machine. The valid values are Disabled, Low, Medium, High, and ClusterRestartPriority. VMware HA is a feature that detects failed virtual machines and automatically restarts them on alternative ESX hosts. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. Specifiesing this parameter is only supported when the virtual machine is inside a cluster. Otherwise, an error is generated.

-HAIsolationResponse <HAIsolationResponse> Indicates whether the virtual machine should be powered off if a host determines that it is isolated from the rest of the compute resource. The available values are

AsSpecifiedByCluster, PowerOff, and DoNothing. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. Specifying this parameter is only supported when the virtual machine is inside a cluster. Otherwise, an error is generated.

-DrsAutomationLevel <DrsAutomationLevel> Specifies a DRS (Distributed Resource Scheduler) automation level. The valid values are FullyAutomated, Manual, PartiallyAutomated, AsSpecifiedByCluster, and Disabled. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. Specifying this parameter is only supported when the virtual machine is inside a cluster. Otherwise, an error is generated.

-VMSwapfilePolicy <VMSwapfilePolicy> Specifies the swapfile placement policy. The following values are valid:

InHostDataStore - Stores the swapfile in the datastore specified by the VMSwapfileDatastoreID property of the virtual machine host. If the VMSwapfileDatastoreID property is not set or indicates a datastore with insufficient free space, the swapfile is stored in the same directory as the virtual machine. This setting might degrade the VMotion performance.

WithVM - Stores the swapfile in the same directory as the virtual machine.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-Notes <String> Provides a description of the new virtual machine. The alias of this parameter is Description.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-OSCustomizationSpec <OSCustomizationSpec> Specifies a customization specification that is to be applied to the new virtual machine.

-LinkedClone Indicates that you want to create a linked clone. When you set the LinkedClone parameter, the ReferenceSnapshot parameter becomes mandatory.

-ReferenceSnapshot <Snapshot> Specifies a source snapshot for the linked clone that you want to create. When you set the LinkedClone parameter, the ReferenceSnapshot parameter becomes mandatory.

-VM <VirtualMachine[]> Specifies a virtual machine to clone.

-Template <Template> Specifies the virtual machine template you want to use for the creation of the new virtual machine. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-VMFilePath <String> Specifies a path to the virtual machine you want to register.

-ContentLibraryItem <ContentLibraryItem> Specifies the content library template to deploy the virtual machine from.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myTargetVMHost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM1 -ResourcePool $my-TargetVMHost -Datastore MyDatastore1 -NumCPU 2 -MemoryGB 4 -DiskGB 40 -NetworkName "VM Net-work" -Floppy -CD -DiskStorageFormat Thin -GuestID winNetDatacenterGuest

Creates a virtual machine by specifying a target host, a target datastore, and a network to connect to, and configures the settings for the virtual machine.

————— Example 2 —————

C:PS>$myCluster = Get-Cluster -Name MyCluster1 New-VM -Name MyVM1 -ResourcePool $myCluster

Creates a virtual machine by specifying a cluster. The ResourcePool parameter accepts ResourcePool, Cluster, VApp, and standalone VMHost objects.

————— Example 3 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myCluster = Get-Cluster -Name MyCluster1 New-VM -Name MyVM1 -VMHost $vmhost -ResourcePool $myCluster -DiskGB 4 -MemoryGB 1

Creates a virtual machine by specifying a cluster and explicitly selecting the host, instead of allowing auto-selection of a target host.

————— Example 4 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM1 -ResourePool $vmhost -DiskGB 40,100

Creates a virtual machine with multiple disks.

————— Example 5 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM1 -ResourcePool $vmhost -DiskPath "[Storage1] WindowsXP/WindowsXP.vmdk"

Creates a virtual machine by specifying an existing disk.

————— Example 6 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM1 -ResourcePool $vmhost -Version v4

Creates a virtual machine by explicitly specifying the version of the virtual machine hardware through the Version parameter.

————— Example 7 —————

C:PS>$myDatastore = Get-Datastore -Name MyDatastore1 $vmhost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM2 -VM MyVM1 -Datastore $myDatastore -VMHost $vmhost

Creates a new virtual machine named MyVM2 by cloning the MyVM1 virtual machine on the specified datastore and host.

————— Example 8 —————

C:PS>New-VM -VM MyVM1, MyVM2 -Location MyFolder1 -VMHost MyHost1

Copies the MyVM1 and MyVM2 virtual machines to the MyFolder1 folder on the MyHost1 host.

————— Example 9 —————

C:PS>$myResourcePool = Get-ResourcePool -Name MyResourcePool1 $mySpecification = Get-OSCustomizationSpec -Name WindowsSpec New-VM -VM MyVM1 -Name MyVM2 -OSCustomizationSpec $mySpecification -ResourcePool $myResourcePool

Clones the virtual machine MyVM1 to MyVM2 and applies a customization specification on the cloned virtual machine.

————— Example 10 —————

C:PS>$myResourcePool = Get-ResourcePool -Name MyResourcePool1 $myTemplate = Get-Template -Name WindowsTemplate $mySpecification = Get-OSCustomizationSpec -Name WindowsSpec New-VM -Name MyVM2 -Template $myTemplate -ResourcePool $myResourcePool -OSCustomizationSpec $mySpecification

Creates a virtual machine from the specified template and applies the specified customization specification.

————— Example 11 —————

C:PS>cd vmstores:myserver@443DatacenterStorage1MyVM1$vmxFile = Get-Item MyVM1.vmx $vmhost = Get-VMHost -Name MyVMHost1 New-VM -VMHost $vmhost -VMFilePath $vmxFile.DatastoreFullPath

Retrieves the specified configuration file for the MyVM1 virtual machine and registers the MyVM1 virtual machine on the specified host.

————— Example 12 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster1 New-VM -Name MyVM1 -DiskGB 40,40 -Datastore $myDatastoreCluster -ResourcePool $vmhost

Creates a virtual machine on a datastore cluster. By default, the new virtual machine has an intra-VM affinity rule.

————— Example 13 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster1 $myAdvancedOption = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMDiskAntiAffinityRule' 1,2 New-VM -Name MyVM1 -DiskGB 40,40,40 -Datastore $myDatastoreCluster -AdvancedOption $myAdvancedOption -ResourcePool $vmhost

Creates a virtual machine on a datastore cluster. The machine has three hard disks. For the first two disks, intra-VM anti-affinity rule is specified and they will be placed on a datastore different from the datastore cluster. Identifying the hard disk in the Intra-VM anti-affinity happens by indexing the disks starting from 1.

————— Example 14 —————

C:PS>$myVM1 = Get-VM -Name WindowsXP $myResourcePool = Get-ResourcePool -Name MyResourcePool1 $myAdvancedOption = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMAntiAffinityRule' $myVM1 $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster1 New-VM -Name MyVM1 -DiskGB 40,40,40 -Datastore $myDatastoreCluster -AdvancedOption $myAdvancedOption -ResourcePool $myResourcePool

Creates a virtual machine on a datastore cluster and specifies a VM anti-affinity rule between the new virtual machine and an existing virtual machine.

————— Example 15 —————

C:PS>$mySourceVM = Get-VM -Name MySourceVM1 $vmhost = Get-VMHost -Name MyVMHost1 $hardDiskList = Get-HardDisk -VM $vm | select -First 2 $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster1 $myAdvancedOption = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMDiskAntiAffinityRule' $hardDiskList New-VM -Name MyVM1 -VM $mySourceVM -Datastore $myDatastoreCluster -AdvancedOption $myAdvancedOption -ResourcePool $vmhost

Clones a virtual machine on a datastore cluster and specifies an intra-VM anti-affinity rule by using references to the hard disks of the source virtual machine. When you apply this rule to the AdvancedOption parameter of

New-VM, the first and second disk of the new virtual machine will be placed on different datastores within the specified datastore cluster.

————— Example 16 —————

C:PS>$mySourceVM = Get-VM -Name MySourceVM1 $myVM1 = Get-VM -Name WindowsXP $myAdvancedOption = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMAntiAffinityRule' $myVM1 $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster1 $vmhost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM2 -VM $mySourceVM -Datastore $myDatastoreCluster -AdvancedOption $myAdvancedOption -ResourcePool $vmhost

Clones a virtual machine on a datastore cluster and specifies a VM anti-afffinity rule between the new virtual machine and an existing virtual machine. When you apply this rule to the AdvancedOption parameter of New-VM, the new virtual machine and the WindowsXP virtual machine will be placed on different datastores within the specified datastore cluster.

————— Example 17 —————

C:PS>$mySourceTemplate = Get-Template -Name WindowsTemplate $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster $myAdvancedOption = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMDiskAntiAffinityRule' $hardDiskList $vmhost = Get-VMHost -Name MyVMHost1 New-VM -Name MyVM1 -Template $mySourceTemplate -Datastore $myDatastoreCluster -AdvancedOption $myAdvancedOption -ResourcePool $vmhost

Creates a virtual machine from a template, specifies a VM anti-affinity rule, and stores the virtual machine on a specified datastore cluster.

————— Example 18 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVM = Get-VM -Name WindowsXP $mySourceTemplate = Get-Template -Name WindowsTemplate $myDatastoreCluster = Get-DatastoreCluster -Name MyStorageCluster1 $myAdvancedOption = New-Object 'VMware.VimAutomation.ViCore.Types.V1.DatastoreManagement.SdrsVMAntiAffinityRule' $myVM New-VM -Name VM -Template $mySourceTemplate -Datastore $myDatastoreCluster -AdvancedOption $myAdvancedOption -ResourcePool $vmhost

Creates a virtual machine from a template, specifies a VM anti-affinity rule, and stores the virtual machine on a specified datastore cluster.

————— Example 19 —————

C:PS>$mySourceVM = Get-VM -Name MySourceVM1 $myReferenceSnapshot = Get-Snapshot -VM $mySourceVM -Name "InitialState" $vmhost = Get-VMHost -Name MyVMHost1 $myDatastore = Get-Datastore -Name MyDatastore1 New-VM -Name MyLinkedCloneVM1 -VM $mySourceVM -LinkedClone -ReferenceSnapshot $myReferenceSnapshot -ResourcePool $vmhost -Datastore $myDatastore

Creates a linked clone from the specified snapshot of the parent virtual machine. The linked clone is stored to the specified VM host and datastore.

————— Example 20 —————

C:PS>$myCluster = Get-Cluster -Name "MyCluster" $myVDPortGroup = Get-VDPortgroup -Name "MyVDPortGroup" $mySharedDatastore = Get-Datastore -Name "MySharedDatastore" New-VM -Name MyVM -ResourcePool $myCluster -Portgroup $myVDPortGroup -DiskGB 40 -MemoryGB 4 -Datastore $mySharedDatastore

Creates a new virtual machine with the specified configuration and connects it to the specified distributed port group.

REMARKS To see the examples, type: "get-help New-VM -examples". For more information, type: "get-help New-VM -detailed". For technical information, type: "get-help New-VM -full". For online help, type: "get-help New-VM -online"

New-VMGuestRoute

NAME New-VMGuestRoute

SYNOPSIS This cmdlet adds a new route to the routing table of the provided virtual machines and guests.

SYNTAX New-VMGuestRoute [-VM <VirtualMachine[]>] [-Destination] <IPAddress> [[-Netmask] <String>] [-Gateway] <IPAddress> [-Interface <Object>] [-VMGuest <VMGuest[]>] [-Server <VIServer[]>] [-ToolsWaitSecs <Int32>] [-GuestPassword <SecureString>] [-GuestUser <String>] [-GuestCredential <PSCredential>] [-HostPassword <SecureString>] [-HostUser <String>] [-HostCredential <PSCredential>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use Invoke-VMGuestScript instead.

This cmdlet adds a new route to the routing table of the provided virtual machines and guests. The cmdlet adds only persistent routes. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Execute and VirtualMachine.GuestOperations.Modify privileges.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines to which you want to add the new route.

-Destination <IPAddress> Specifies a destination IP address for the new route.

-Netmask <String> Specifies a network mask for the new route.

-Gateway <IPAddress> Specifies a gateway for the new route.

-Interface <Object> Specifies a network interface for the new route. For Linux guest operating systems, this parameter is mandatory.

-VMGuest <VMGuest[]> Specifies the guests to which you want to add the new route.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from VMware Tools. If a non-positive value is provided, the system waits infinitely long time.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest OS.

-GuestUser <String> Specifies the user name you want to use for authenticating with the guest OS.

-GuestCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the guest OS. Do not use this parameter if the GuestUser and GuestPassword parameters are used.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host. Do not use this parameter if the HostUser and HostPassword parameters are used. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-VMGuestRoute -GuestUser user -GuestPassword pass2 -VM $vm -Destination '192.168.100.10' -Netmask '255.255.255.255' -Gateway '10.23.112.58'

Creates a new guest route with the specified parameters.

REMARKS To see the examples, type: "get-help New-VMGuestRoute -examples". For more information, type: "get-help New-VMGuestRoute -detailed". For technical information, type: "get-help New-VMGuestRoute -full". For online help, type: "get-help New-VMGuestRoute -online"

New-VMHostAccount

NAME New-VMHostAccount

SYNOPSIS This cmdlet creates a new host user or group account.

SYNTAX New-VMHostAccount [-Id] <String> [-Password] <String> [-Description <String>] [-UserAccount] [-AssignGroups <String[]>] [-GrantShellAccess] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

New-VMHostAccount [-Id] <String> [-GroupAccount] [-AssignUsers <String[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new host user or group account using the provided parameters.

PARAMETERS

-Id <String> Specifies an ID for the new host account.

-Password <String> Specifies a password for the new host account.

-Description <String> Provide a description of the new host account. The maximum length of the text is 255 symbols.

-UserAccount Indicates that the new host account is a user account.

-AssignGroups <String[]> If the UserAccount parameter is set to $true, use AssignGroups to specify the groups to which the newly created user belongs.

-GrantShellAccess Indicates that the new account is allowed to access the ESX shell.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-GroupAccount Indicates that the new host account is a group account. Starting with ESXi 5.1, this parameter is not supported.

-AssignUsers <String[]> If the GroupAccount parameter is set to $true, use AssignUsers to specify the users that belong to the newly created group account.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>New-VMHostAccount -ID MyUser1 -Password MyPassword1 -UserAccount

Creates a user account with a specified user ID and password. The user account is created on the default server.

————— Example 2 —————

C:PS>$myUser1 = Get-VMHostAccount -ID MyUser1 -User New-VMHostAccount -Id MyGroup1 -GroupAccount -AssignUsers $myUser1

Creates a group account with a specified ID and assigns a specified user to the group account. Starting with ESXi 5.1, you cannot create group host accounts.

REMARKS To see the examples, type: “get-help New-VMHostAccount -examples”. For more information, type: “get-help New-VMHostAccount -detailed”. For technical information, type: “get-help New-VMHostAccount -full”. For online help, type: “get-help New-VMHostAccount -online”

New-VMHostNetworkAdapter

NAME New-VMHostNetworkAdapter

SYNOPSIS This cmdlet creates a new HostVirtualNIC (Service Console or VMKernel) on the specified host.

SYNTAX New-VMHostNetworkAdapter [[-VMHost] <VMHost>] [[-PortGroup] <String>] [-PortId <String>] [-VirtualSwitch] <VirtualSwitchBase> [[-IP] <String>] [[-SubnetMask] <String>] [[-Mac] <String>] [-Mtu <Int32>] [-ConsoleNic] [-VMotionEnabled <Boolean>] [-FaultToleranceLoggingEnabled <Boolean>] [-IPv6ThroughDhcp] [-AutomaticIPv6] [-IPv6 <String[]>] [-ManagementTrafficEnabled <Boolean>] [-VsanTrafficEnabled <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new HostVirtualNIC (Service Console or VMKernel) on the specified host. Creates a port group with a name specified by the PortGroup parameter on the virtual switch passed through the VirtualSwitch parameter. Adds either a Console NIC if ConsoleNIC is set, or a VMKernel NIC otherwise.

PARAMETERS

-VMHost <VMHost> Specifies the host to which you want to add the new adapter. This parameter is mandatory when creating a network adapter on a distributed switch.

-PortGroup <String> Specifies the port group to which you want to add the new adapter. If a distributed switch is passed to the VirtualSwitch parameter, an existing port group name should be specified. For standard virtual switches, if the port group is non-existent, a new port group with the specified name will be created and the new adapter will be added to the port group.

-PortId <String> Specifies the port of the specified distributed switch to which you want to connect the network adapter. Use this parameter only if a distributed switch is passed to the VirtualSwitch parameter.

-VirtualSwitch <VirtualSwitchBase> Specifies the virtual switch to which you want to add the new network adapter.

-IP <String> Specifies an IP address for the new network adapter. All IP addresses are specified using IPv4 dot notation. If IP is not specified, DHCP mode is enabled. For VMKernel network adapters, the DHCP mode is supported only on vCenter Server 4.1, ESX 4.1, and later.

-SubnetMask <String> Specifies a subnet mask for the new network adapter.

-Mac <String> Specifies a media access control (MAC) address for the new virtual network adapter.

-Mtu <Int32> Specifies the MTU size. This parameter is supported only on ESX/vCenter Server 4.1 and later.

-ConsoleNic If the value is $true, indicates that you want to create a service console virtual network adapter. If the value is $false, indicates that you want to create a virtual host/VMkernel network adapter. Not supported on ESXi.

-VMotionEnabled [<Boolean>] Indicates that you want to use the new virtual host/VMKernel network adapter for VMotion.

-FaultToleranceLoggingEnabled [<Boolean>] Indicates that the network adapter is enabled for Fault Tolerance (FT) logging. This parameter is supported only on ESX/vCenter Server 4.1 and later.

-IPv6ThroughDhcp Indicates that the IPv6 address is obtained through DHCP.

-AutomaticIPv6 Indicates that the IPv6 address is obtained through a router advertisement.

-IPv6 <String[]> Specifies multiple static addresses using the following format: <IPv6>/<subnet_prefix_length> or <IPv6>. If you skip <subnet_prefix_length>, the default value of 64 is used.

-ManagementTrafficEnabled [<Boolean>] Indicates that you want to enable the network adapter for management traffic. This parameter is supported only on ESX/ESXi/vCenter Server 4.1 and later.

-VsanTrafficEnabled [<Boolean>] Indicates that Virtual SAN traffic is enabled on this network adapter.

-Server <VIServer[]> The Server parameter is required when the host is specified by name. In this case, the host with the specified name is searched for on the specified Servers and a network adapter is added to it. If a VMHost object is passed to the VMHost parameter, the Server parameter is not used.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVirtualSwitch = Get-VirtualSwitch -VMHost $vmhost -Name MyVirtualSwitch1 New-VMHostNetworkAdapter -VMHost $vmhost -PortGroup MyVMKernelPortGroup1 -VirtualSwitch $myVirtualSwitch -Mtu 4000

Creates a VMKernel port group at the MyVirtualSwitch1 virtual switch. The IP address is obtained via DHCP.

———— Example 2 ————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVirtualSwitch = Get-VirtualSwitch -VMHost $vmhost -Name MyVirtualSwitch1 New-VMHostNetworkAdapter -VMHost $vmhost -PortGroup MyVMKernelPortGroup1 -VirtualSwitch $myVirtualSwitch -IP 192.168.168.110 -SubnetMask 255.255.255.0

Creates a VMKernel port group at the MyVirtualSwitch1 virtual switch and assigns a static IP address.

————— Example 3 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVirtualSwitch = Get-VirtualSwitch -VMHost $vmhost -Name MyVirtualSwitch1 New-VMHostNetworkAdapter -VMHost $vmhost -VirtualSwitch $myVirtualSwitch -PortGroup MyVMKernelPortGroup1 -IP 192.168.0.1 -SubnetMask 255.255.255.0 -IPv6 “0200:2342::1/32”

Creates a VMKernel NIC that has an IPv4 address and an IPv6 address.

————— Example 4 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVirtualSwitch = Get-VirtualSwitch -VMHost $vmhost -Name MyVirtualSwitch1 New-VMHostNetworkAdapter -VMHost $vmhost -VirtualSwitch $myVirtualSwitch -PortGroup MyVMKernelPortGroup1 -IP 192.168.0.1 -SubnetMask 255.255.255.0 -AutomaticIPv6

Creates a VMKernel NIC that obtains IPv6 automatically (by router advertisement) and takes the desired IPv4 address.

————— Example 5 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVirtualSwitch = Get-VirtualSwitch -VMHost $vmhost -Name MyVirtualSwitch1 New-VMHostNetworkAdapter -VMHost $vmhost -VirtualSwitch $myVirtualSwitch -PortGroup MyVMKernelPortGroup1 -IPv6ThroughDhcp

Creates a VMKernel NIC that obtains the IPv4 and IPv6 addresses by DHCP.

————— Example 6 —————

C:PS>$vmhost = Get-VMHost -Name MyVMHost1 $myVirtualSwitch = Get-VirtualSwitch -VMHost $vmhost -Name MyVirtualSwitch1 New-VMHostNetworkAdapter -VMHost $vmhost -PortGroup MyConsolePortGroup1 -VirtualSwitch $myVirtualSwitch -ConsoleNic

Creates a Service Console port group at the vSwitch virtual switch. The IP address is obtained via DHCP.

————— Example 7 —————

C:PS>$myVMHost = Get-VMHost -Name “MyVMHost” $myVDSwitch = Get-VDSwitch -Name “MyVDSwitch” New-VMHostNetworkAdapter -VMHost $myVMHost -PortGroup “MyVDPortgroup” -VirtualSwitch $myVDSwitch -IP 192.168.0.50 -SubnetMask 255.255.255.0

Creates a new VMKernel network adapter and connects it to the specified port group on the specified distributed switch.

————— Example 8 —————

C:PS>$myVMHost = Get-VMHost -Name “MyVMHost” $myVDSwitch = Get-VDSwitch -Name “MyVDSwitch” New-VMHostNetworkAdapter -VMHost $myVMHost -VirtualSwitch $myVDSwitch -PortId 100 -IP 192.168.0.50 -SubnetMask 255.255.255.0

Creates a new VMKernel network adapter and connects it to a specified port on a specified distributed switch.

REMARKS To see the examples, type: “get-help New-VMHostNetworkAdapter -examples”. For more information, type: “get-help New-VMHostNetworkAdapter -detailed”. For technical information, type: “get-help New-VMHostNetworkAdapter -full”. For online help, type: “get-help New-VMHostNetworkAdapter -online”

New-VMHostProfile

NAME New-VMHostProfile

SYNOPSIS This cmdlet creates a new host profile based on a reference host.

SYNTAX New-VMHostProfile [-Name] <String> [-ReferenceHost] <VMHost> [-Description <String>] [-CompatibilityMode] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new host profile based on a reference host.

PARAMETERS

-Name <String> Specifies a name for the new host profile.

-ReferenceHost <VMHost> Specifies the reference host, on which the new virtual machine host profile is based.

-Description <String> Provides a description for the new host profile.

-CompatibilityMode If you are connected to a vCenter Server/ESX 5.0 or later, use this parameter to indicate that you want the new profile to be compatible with hosts running ESX/vCenter Server versions earlier than 5.0.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$h = Get-VMHost 10.23.134.133

New-VMHostProfile -Name testProfile -Description "This is my first test profile." -ReferenceHost $h

Creates a profile based on the virtual machine host with an IP address 10.23.134.133.

REMARKS To see the examples, type: "get-help New-VMHostProfile -examples". For more information, type: "get-help New-VMHostProfile -detailed". For technical information, type: "get-help New-VMHostProfile -full". For online help, type: "get-help New-VMHostProfile -online"

New-VMHostRoute

NAME New-VMHostRoute

SYNOPSIS This cmdlet creates a new route in the routing table of a host.

SYNTAX New-VMHostRoute [-VMHost] <VMHost[]> -Destination <IPAddress> -Gateway <IPAddress> -PrefixLength <Int32> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet creates a new route in the routing table of a host.

PARAMETERS

-VMHost <VMHost[]> Specifies the host for which you want to create a new route.

-Destination <IPAddress> Specifies a destination IP address for the new route.

-Gateway <IPAddress> Specifies a gateway IP address for the new route.

-PrefixLength <Int32> Specifies the prefix length of the destination IP address. For IPv4, the valid values are from 0 to 32, and for IPv6 - from 0 to 128.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>New-VMHostRoute -VMHost 10.23.114.195 -Destination 192.168.103.102 -PrefixLength 32 -Gateway 10.23.84.53

Creates a route for the specified host.

REMARKS To see the examples, type: "get-help New-VMHostRoute -examples". For more information, type: "get-help New-VMHostRoute -detailed". For technical information, type: "get-help New-VMHostRoute -full". For online help, type: "get-help New-VMHostRoute -online"

CHAPTER 22

Open Commands

This page contains details on **Open** commands.

Open-VMConsoleWindow

NAME Open-VMConsoleWindow

SYNOPSIS This cmdlet opens a window to the virtual machine's console.

SYNTAX Open-VMConsoleWindow [-VM] <RemoteConsoleVM[]> [-FullScreen] [-UrlOnly] [-Server <VIConnection[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet opens a window to the virtual machine's console. The window is opened in a Web page in the browser configured in the VMConsoleWindowBrowser setting (in Set-PowerCLIConfiguration), or in the default Web browser, if the setting is not configured.

If the default browser is not 32-bit, you must configure a 32-bit browser to be used by this cmdlet. This configuration is done through the Set-PowerCLIConfiguration cmdlet, by specifying the VMConsoleWindowBrowser setting and providing the full path to a browser's executable file. The officially supported browsers are listed on the VMware site, under the VMRC distributable (which is used to display the virtual machine's console).

PARAMETERS

-VM <RemoteConsoleVM[]> Specifies the virtual machines for which to open a remote console. Supports vCloud and vSphere virtual machines.

-FullScreen	If specified, opens the virtual machine console window in full screen mode.
-UrlOnly	If specified, the cmdlet returns the URL for opening a console window to the virtual machine, without opening the console window. Note: The URL is valid for 30 seconds. After 30 seconds, the screen authentication ticket contained in the URL expires.

-Server <VIConnection[]> Specifies the vCenter Server systems or cloud server instances on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-CIVM myVM | Open-VMConsoleWindow -FullScreen

Opens the console of the specified virtual machine in full screen mode.

REMARKS To see the examples, type: “get-help Open-VMConsoleWindow -examples”. For more information, type: “get-help Open-VMConsoleWindow -detailed”. For technical information, type: “get-help Open-VMConsoleWindow -full”. For online help, type: “get-help Open-VMConsoleWindow -online”

CHAPTER 23

Remove Commands

This page contains details on **Remove** commands.

Remove-AdvancedSetting

NAME Remove-AdvancedSetting

SYNOPSIS This cmdlet removes the specified advanced setting.

SYNTAX Remove-AdvancedSetting [-AdvancedSetting] <AdvancedSetting[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified advanced setting.

PARAMETERS

-AdvancedSetting <AdvancedSetting[]> Specifies the advanced settings you want to remove.

Note: You can only remove advanced settings from virtual machines in ESXi or vCenter Server environments version 5.5 or later.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-AdvancedSetting -Entity (Get-Cluster -Name Cluster) | Remove-AdvancedSetting -Confirm:$false

Removes the advanced settings of the cluster named Cluster.

REMARKS To see the examples, type: "get-help Remove-AdvancedSetting -examples". For more information, type: "get-help Remove-AdvancedSetting -detailed". For technical information, type: "get-help Remove-AdvancedSetting -full". For online help, type: "get-help Remove-AdvancedSetting -online"

Remove-AlarmAction

NAME Remove-AlarmAction

SYNOPSIS This cmdlet removes an alarm action.

SYNTAX Remove-AlarmAction [-AlarmAction] <AlarmAction[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes an alarm action.

PARAMETERS

-AlarmAction <AlarmAction[]> Specifies the alarm actions you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AlarmDefinition -Name "Alarm1" | Get-AlarmAction | Remove-AlarmAction -Confirm:$false

Removes all actions for an alarm definition.

REMARKS To see the examples, type: "get-help Remove-AlarmAction -examples". For more information, type: "get-help Remove-AlarmAction -detailed". For technical information, type: "get-help Remove-AlarmAction -full". For online help, type: "get-help Remove-AlarmAction -online"

Remove-AlarmActionTrigger

NAME Remove-AlarmActionTrigger

SYNOPSIS This cmdlet removes the alarm action triggers.

SYNTAX Remove-AlarmActionTrigger [-AlarmActionTrigger] <AlarmActionTrigger[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the selected alarm action triggers.

PARAMETERS

-AlarmActionTrigger <AlarmActionTrigger[]> Specifies the alarm action triggers you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AlarmDefinition -Name "Alarm1" | Get-AlarmAction | Get-AlarmActionTrigger | select -First 1 | Remove-AlarmActionTrigger -Confirm:$false

Removes the first action trigger found for an alarm definition.

REMARKS To see the examples, type: "get-help Remove-AlarmActionTrigger -examples". For more information, type: "get-help Remove-AlarmActionTrigger -detailed". For technical information, type: "get-help Remove-AlarmActionTrigger -full". For online help, type: "get-help Remove-AlarmActionTrigger -online"

Remove-CDDrive

NAME Remove-CDDrive

SYNOPSIS This cmdlet removes virtual CD drives from their locations.

SYNTAX Remove-CDDrive [-CD] <CDDrive[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes virtual CD drives from their locations.

PARAMETERS

-CD <CDDrive[]> Specifies the virtual CD drives you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$cd = Get-CDDrive -VM $vm -Template $template

Remove-CDDrive -CD $cd

Removes all CD drives for the specified virtual machines and templates.

REMARKS To see the examples, type: "get-help Remove-CDDrive -examples". For more information, type: "get-help Remove-CDDrive -detailed". For technical information, type: "get-help Remove-CDDrive -full". For online help, type: "get-help Remove-CDDrive -online"

Remove-Cluster

NAME Remove-Cluster

SYNOPSIS This cmdlet deletes the specified clusters.

SYNTAX Remove-Cluster [-Cluster] <Cluster[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet deletes the specified clusters.

PARAMETERS

-Cluster <Cluster[]> Specifies the clusters you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$cluster = New-Cluster -Name Cluster -Location Datacenter

Remove-Cluster $cluster -Confirm:$false

Creates and then removes, without asking for user confirmation, the Custer cluster on the Datacenter datacenter.

REMARKS To see the examples, type: "get-help Remove-Cluster -examples". For more information, type: "get-help Remove-Cluster -detailed". For technical information, type: "get-help Remove-Cluster -full". For online help, type: "get-help Remove-Cluster -online"

Remove-CustomAttribute

NAME Remove-CustomAttribute

SYNOPSIS This cmdlet removes custom attributes.

SYNTAX Remove-CustomAttribute [-CustomAttribute] <CustomAttribute[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes custom attributes.

PARAMETERS

-CustomAttribute <CustomAttribute[]> Specifies the custom attributes you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-CustomAttribute -CustomAttribute “CompanyID”, “Owner” -Server $agent007

Removes the CompanyID and Owner custom attributes from the server stored in the $agent007 variable.

REMARKS To see the examples, type: “get-help Remove-CustomAttribute -examples”. For more information, type: “get-help Remove-CustomAttribute -detailed”. For technical information, type: “get-help Remove-CustomAttribute -full”. For online help, type: “get-help Remove-CustomAttribute -online”

Remove-Datacenter

NAME Remove-Datacenter

SYNOPSIS This cmdlet removes the specified datacenters from their locations.

SYNTAX Remove-Datacenter [-Datacenter] <Datacenter[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified datacenters and their children objects from their locations.

PARAMETERS

-Datacenter <Datacenter[]> Specifies the datacenters you want to remove.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-Datacenter Datacenter

Removes the Datacenter datacenter.

————— Example 2 —————

C:PS>$task = Remove-Datacenter Datacenter -RunAsync

Asynchronously removes Datacenter08.

REMARKS To see the examples, type: “get-help Remove-Datacenter -examples”. For more information, type: “get-help Remove-Datacenter -detailed”. For technical information, type: “get-help Remove-Datacenter -full”. For online help, type: “get-help Remove-Datacenter -online”

Remove-Datastore

NAME Remove-Datastore

SYNOPSIS This cmdlet removes the specified datastores from their locations.

SYNTAX Remove-Datastore [-Datastore] <Datastore[]> [-VMHost] <VMHost> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified datastores from their locations. The cmdlet permanently deletes the content of the removed datastores, unless they are shared (NFS).

PARAMETERS

-Datastore <Datastore[]> Specifies the datastores you want to remove.

-VMHost <VMHost> Specifies the host to which the datastore you want to remove belongs.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Remove-Datastore -Datastore Datastore -VMHost 10.23.112.234 -Confirm:$false

Removes the Datastore datastore from the host.

REMARKS To see the examples, type: “get-help Remove-Datastore -examples”. For more information, type: “get-help Remove-Datastore -detailed”. For technical information, type: “get-help Remove-Datastore -full”. For online help, type: “get-help Remove-Datastore -online”

Remove-DatastoreCluster

NAME Remove-DatastoreCluster

SYNOPSIS This cmdlet deletes the specified datastore clusters.

SYNTAX Remove-DatastoreCluster [-DatastoreCluster] <DatastoreCluster[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet deletes the specified datastore clusters.

PARAMETERS

-DatastoreCluster <DatastoreCluster[]> Specifies the datastore cluster that you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-DatastoreCluster -Name ‘MyDatastoreCluster’ | Remove-DatastoreCluster -Confirm $false

Removes the specified datastore cluster without asking for confirmation.

REMARKS To see the examples, type: “get-help Remove-DatastoreCluster -examples”. For more information, type: “get-help Remove-DatastoreCluster -detailed”. For technical information, type: “get-help Remove-DatastoreCluster -full”. For online help, type: “get-help Remove-DatastoreCluster -online”

Remove-DrsRule

NAME Remove-DrsRule

SYNOPSIS This cmdlet removes the specified DRS rules.

SYNTAX Remove-DrsRule [-Rule] <DrsRule[]> [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified DRS rules.

PARAMETERS

-Rule <DrsRule[]> Specifies the DRS rules you want to remove.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$rules = Get-DrsRule -Cluster $cluster -Name "*Rule1*"

Remove-DrsRule $rules -Confirm:$false

Removes the DRS rules for the $cluster cluster, whose names contain "Rule1".

REMARKS To see the examples, type: "get-help Remove-DrsRule -examples". For more information, type: "get-help Remove-DrsRule -detailed". For technical information, type: "get-help Remove-DrsRule -full". For online help, type: "get-help Remove-DrsRule -online"

Remove-FloppyDrive

NAME Remove-FloppyDrive

SYNOPSIS This cmdlet removes the virtual floppy drives from their locations.

SYNTAX Remove-FloppyDrive [-Floppy] <FloppyDrive[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the virtual floppy drives from their locations.

PARAMETERS

-Floppy <FloppyDrive[]> Specifies the virtual floppy drives you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$floppy = Get-FloppyDrive -VM VM

Remove-FloppyDrive -Floppy $floppy

Removes the floppy drive of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Remove-FloppyDrive -examples". For more information, type: "get-help Remove-FloppyDrive -detailed". For technical information, type: "get-help Remove-FloppyDrive -full". For online help, type: "get-help Remove-FloppyDrive -online"

Remove-Folder

NAME Remove-Folder

SYNOPSIS This cmdlet removes the specified folders from their locations.

SYNTAX Remove-Folder [-Folder] <Folder[]> [-DeletePermanently] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified folders and their children objects from their locations.

PARAMETERS

-Folder <Folder[]> Specifies the folders you want to remove.

-DeletePermanently Indicates that you want to delete from the disk any virtual machines contained in the specified folder, and not only to remove them from the inventory. This parameter is supported only for VirtualMachine folders.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-Folder -Folder "testFolder"

Removes a folder by name.

————— Example 2 —————

C:PS>Get-Folder -Name "testFolder" | Remove-Folder

Removes a folder by object.

————— Example 3 —————

C:PS>Get-Folder -Name "testFolder" | Remove-Folder -DeletePermanently

Permanently removes a folder.

REMARKS To see the examples, type: "get-help Remove-Folder -examples". For more information, type: "get-help Remove-Folder -detailed". For technical information, type: "get-help Remove-Folder -full". For online help, type: "get-help Remove-Folder -online"

Remove-HardDisk

NAME Remove-HardDisk

SYNOPSIS This cmdlet removes the specified virtual hard disks.

SYNTAX Remove-HardDisk [-HardDisk] <HardDisk[]> [-DeletePermanently] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified virtual hard disks.

PARAMETERS

-HardDisk <HardDisk[]> Specifies the hard disks you want to remove.

-DeletePermanently Indicates that you want to delete the hard disks not only from the inventory, but from the datastore as well.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-HardDisk -VM $vm | Remove-HardDisk

Removes the hard disks of the virtual machine stored in the $vm variable.

————— Example 2 —————

C:PS>$hdd = Get-HardDisk -VM 'MyVM' -Name 'Hard disk 4' Remove-HardDisk -HardDisk $hdd

Removes the 'Hard disk 4' hard disk of the 'MyVM' virtual machine.

REMARKS To see the examples, type: "get-help Remove-HardDisk -examples". For more information, type: "get-help Remove-HardDisk -detailed". For technical information, type: "get-help Remove-HardDisk -full". For online help, type: "get-help Remove-HardDisk -online"

Remove-Inventory

NAME Remove-Inventory

SYNOPSIS This cmdlet removes the specified inventory items from their locations.

SYNTAX Remove-Inventory [-Item] <InventoryItem[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified inventory items and their children from their locations.

PARAMETERS

-Item <InventoryItem[]> Specifies the inventory items you want to remove. This parameter accepts Folder, ResourcePool, Datacenter, VirtualMachine, VMHost, Cluster, Template, and VApp objects.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Folder Folder| Get-Inventory -NoRecursion | Remove-Inventory

Removes all objects from the Folder folder.

REMARKS To see the examples, type: "get-help Remove-Inventory -examples". For more information, type: "get-help Remove-Inventory -detailed". For technical information, type: "get-help Remove-Inventory -full". For online help, type: "get-help Remove-Inventory -online"

Remove-IScsiHbaTarget

NAME Remove-IScsiHbaTarget

SYNOPSIS This cmdlet removes targets from their iSCSI HBAs.

SYNTAX Remove-IScsiHbaTarget [-Target] <IScsiHbaTarget[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes targets from their iSCSI HBAs.

PARAMETERS

-Target <IScsiHbaTarget[]> Specifies the iSCSI HBA targets you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-IScsiHbaTarget -Address 10.23.84.73 -Type Send | Remove-IScsiHbaTarget

Retrieves and removes the targets of type Send on the specified address.

———— Example 2 ————

C:PS>Remove-IScsiHbaTarget -Target (Get-IScsiHbaTarget -Address 10.23.84.73)

Removes the specified iSCSI HBA targets.

REMARKS To see the examples, type: "get-help Remove-IScsiHbaTarget -examples". For more information, type: "get-help Remove-IScsiHbaTarget -detailed". For technical information, type: "get-help Remove-IScsiHbaTarget -full". For online help, type: "get-help Remove-IScsiHbaTarget -online"

Remove-NetworkAdapter

NAME Remove-NetworkAdapter

SYNOPSIS This cmdlet removes the virtual network adapters from their locations.

SYNTAX Remove-NetworkAdapter [-NetworkAdapter] <NetworkAdapter[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the virtual network adapters from their locations.

PARAMETERS

-NetworkAdapter <NetworkAdapter[]> Specifies the virtual network adapters you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$nic = Get-NetworkAdapter -VM VM

Remove-NetworkAdapter -NetworkAdapter $nic

Removes the network adapter of the VM virtual machine.

REMARKS To see the examples, type: "get-help Remove-NetworkAdapter -examples". For more information, type: "get-help Remove-NetworkAdapter -detailed". For technical information, type: "get-help Remove-NetworkAdapter -full". For online help, type: "get-help Remove-NetworkAdapter -online"

Remove-OSCustomizationNicMapping

NAME Remove-OSCustomizationNicMapping

SYNOPSIS This cmdlet removes the specified OS customization NIC mappings.

SYNTAX Remove-OSCustomizationNicMapping [-OSCustomizationNicMapping] <OSCustomizationNicMapping[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified OS customization NIC mappings.

PARAMETERS

-OSCustomizationNicMapping <OSCustomizationNicMapping[]> Specifies the OSCustomizationNicMapping objects you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$nicMapping = Get-OSCustomization MyCustomizationSpec | Get-OSCustomizationNicMapping Remove-OSCustomizationNicMapping $nicMapping -Confirm:$false

Removes the NIC mappings of the specified OS customization spec without asking for confirmation.

REMARKS To see the examples, type: "get-help Remove-OSCustomizationNicMapping -examples". For more information, type: "get-help Remove-OSCustomizationNicMapping -detailed". For technical information, type: "get-help Remove-OSCustomizationNicMapping -full". For online help, type: "get-help Remove-OSCustomizationNicMapping -online"

Remove-OSCustomizationSpec

NAME Remove-OSCustomizationSpec

SYNOPSIS This cmdlet removes the specified OS customization specifications.

SYNTAX Remove-OSCustomizationSpec [-OSCustomizationSpec] <OSCustomizationSpec[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified OS customization specifications.

PARAMETERS

-OSCustomizationSpec <OSCustomizationSpec[]> Specifies the customization specifications you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-OSCustomizationSpec Spec -Confirm

Removes the Spec OS customization specification from the server.

REMARKS To see the examples, type: "get-help Remove-OSCustomizationSpec -examples". For more information, type: "get-help Remove-OSCustomizationSpec -detailed". For technical information, type: "get-help Remove-OSCustomizationSpec -full". For online help, type: "get-help Remove-OSCustomizationSpec -online"

Remove-PassthroughDevice

NAME Remove-PassthroughDevice

SYNOPSIS This cmdlet removes the specified pass-through devices.

SYNTAX Remove-PassthroughDevice [-PassthroughDevice] <PassThroughDevice[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified pass-through devices. You can remove only those pass-through devices that are retrieved from virtual machines.

PARAMETERS

-PassthroughDevice <PassThroughDevice[]> Specifies the pass-through devices you want to remove. You can remove only those pass-through devices that are retrieved from virtual machines.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-PassthroughDevice -VM VM | Remove-PassthroughDevice

Removes all pass-through devices of the VM virtual machine.

REMARKS To see the examples, type: “get-help Remove-PassthroughDevice -examples”. For more information, type: “get-help Remove-PassthroughDevice -detailed”. For technical information, type: “get-help Remove-PassthroughDevice -full”. For online help, type: “get-help Remove-PassthroughDevice -online”

Remove-ResourcePool

NAME Remove-ResourcePool

SYNOPSIS This cmdlet removes the specified resource pools from their locations.

SYNTAX Remove-ResourcePool [-ResourcePool] <ResourcePool[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified resource pools and their children objects from their locations.

PARAMETERS

-ResourcePool <ResourcePool[]> Specifies the resource pools you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Remove-ResourcePool -ResourcePool ResourcePool

Removes the resource pool named ResourcePool.

REMARKS To see the examples, type: “get-help Remove-ResourcePool -examples”. For more information, type: “get-help Remove-ResourcePool -detailed”. For technical information, type: “get-help Remove-ResourcePool -full”. For online help, type: “get-help Remove-ResourcePool -online”

Remove-Snapshot

NAME Remove-Snapshot

SYNOPSIS This cmdlet removes the specified virtual machine snapshots.

SYNTAX Remove-Snapshot [-Snapshot] <Snapshot[]> [-RemoveChildren] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified virtual machine snapshots. If the value of the RemoveChildren parameter is $true, the cmdlet removes the child snapshots as well.

PARAMETERS

-Snapshot <Snapshot[]> Specifies the snapshots you want to remove.

-RemoveChildren	Indicates that you want to remove the children of the specified snapshots as well.
-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-Snapshot -Snapshot $snapshot1 -RemoveChildren

Removes the snapshot in the $snapshot variable and its children.

REMARKS To see the examples, type: “get-help Remove-Snapshot -examples”. For more information, type: “get-help Remove-Snapshot -detailed”. For technical information, type: “get-help Remove-Snapshot -full”. For online help, type: “get-help Remove-Snapshot -online”

Remove-StatInterval

NAME Remove-StatInterval

SYNOPSIS This cmdlet removes the statistics interval specified by the provided sampling period or name.

SYNTAX Remove-StatInterval [-Interval] <StatInterval[]> [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the statistics interval specified by the provided sampling period or name.

PARAMETERS

-Interval <StatInterval[]> Specifies the statistics intervals you want to remove. The values of this parameter can be statistics interval objects, names, or refresh periods in seconds.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-StatInterval -Interval *

Removes all the statistics intervals. Note that deleting statistics intervals is allowed only on VirtualCenter 2.0.

REMARKS To see the examples, type: "get-help Remove-StatInterval -examples". For more information, type: "get-help Remove-StatInterval -detailed". For technical information, type: "get-help Remove-StatInterval -full". For online help, type: "get-help Remove-StatInterval -online"

Remove-Tag

NAME Remove-Tag

SYNOPSIS This cmdlet removes the specified tags from the server.

SYNTAX Remove-Tag [-Tag] <Tag[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified tags from the server.

PARAMETERS

-Tag <Tag[]> Specifies the tags you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$tagCategory = Get-TagCategory “MyTagCategory” Get-Tag -Name “MyTag1”, “MyTag2” -Category $tagCategory | Remove-Tag

Retrieves the tags named “MyTag1” and “MyTag2” from the specified tag category named “MyTagCategory” and then removes the tags from the vCenter Server system.

REMARKS To see the examples, type: “get-help Remove-Tag -examples”. For more information, type: “get-help Remove-Tag -detailed”. For technical information, type: “get-help Remove-Tag -full”. For online help, type: “get-help Remove-Tag -online”

Remove-TagAssignment

NAME Remove-TagAssignment

SYNOPSIS This cmdlet removes the specified tag assignment.

SYNTAX Remove-TagAssignment [-TagAssignment] <TagAssignment[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified tag assignment. The cmdlet removes the assignment of the tag in TagAssignment.Tag from the entity in TagAssignment.Entity.

PARAMETERS

-TagAssignment <TagAssignment[]> Specifies the assigned tags to be removed.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myVM = Get-VM myvm $myTagAssignment = Get-TagAssignment $myVM Remove-TagAssignment $myTagAssignment

Removes all connections to tags from the specified virtual machine entity.

REMARKS To see the examples, type: “get-help Remove-TagAssignment -examples”. For more information, type: “get-help Remove-TagAssignment -detailed”. For technical information, type: “get-help Remove-TagAssignment -full”. For online help, type: “get-help Remove-TagAssignment -online”

Remove-TagCategory

NAME Remove-TagCategory

SYNOPSIS This cmdlet removes the specified tag categories from the server.

SYNTAX Remove-TagCategory [-Category] <TagCategory[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified tag categories from the server.

Note: This will remove all tags in the category and any assignments of these tags.

PARAMETERS

-Category <TagCategory[]> Specifies the categories you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-TagCategory "MyTagCategory" | Remove-TagCategory

Retrieves a tag category named "MyTagCategory" and then removes it from the vCenter Server system.

REMARKS To see the examples, type: "get-help Remove-TagCategory -examples". For more information, type: "get-help Remove-TagCategory -detailed". For technical information, type: "get-help Remove-TagCategory -full". For online help, type: "get-help Remove-TagCategory -online"

Remove-Template

NAME Remove-Template

SYNOPSIS This cmdlet removes the specified virtual machine templates from the inventory.

SYNTAX Remove-Template [-Template] <Template[]> [-DeletePermanently] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified virtual machine templates from the inventory. If the value of the DeletePermanently parameter is $true, the cmdlet removes the templates from the inventory and deletes them from the disk.

PARAMETERS

-Template <Template[]> Specifies the virtual machine templates you want to remove.

-DeletePermanently Indicates that you want to delete the templates not only from the inventory, but from the datastore as well.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is passed to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-Template -Template $template

Removes the virtual machine template saved in the $template variable.

REMARKS To see the examples, type: "get-help Remove-Template -examples". For more information, type: "get-help Remove-Template -detailed". For technical information, type: "get-help Remove-Template -full". For online help, type: "get-help Remove-Template -online"

Remove-UsbDevice

NAME Remove-UsbDevice

SYNOPSIS This cmdlet removes the specified USB devices from a virtual machine.

SYNTAX Remove-UsbDevice [-UsbDevice] <UsbDevice[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified USB devices from a virtual machine.

PARAMETERS

-UsbDevice <UsbDevice[]> Specifies the USB devices you want to remove.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-UsbDevice -VM (Get-VM -Location $vmhost) | Remove-UsbDevice

Retrieves the virtual machines on the host stored in the $vmhost variable and removes their USB devices.

REMARKS To see the examples, type: "get-help Remove-UsbDevice -examples". For more information, type: "get-help Remove-UsbDevice -detailed". For technical information, type: "get-help Remove-UsbDevice -full". For online help, type: "get-help Remove-UsbDevice -online"

Remove-VApp

NAME Remove-VApp

SYNOPSIS This cmdlet removes vApps from the server.

SYNTAX Remove-VApp [-DeletePermanently] [-VApp] <VApp[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes vApps from the server.

PARAMETERS

-DeletePermanently Indicates that you want not only to remove the vApps from the inventory, but also to delete the virtual machines they contain from the datastore.

-VApp <VApp[]> Specifies the vApp you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost -Name MyVMHost1 | Get-VApp | Remove-VApp

Retrieves and removes all vApps available on the MyVMHost1 host.

REMARKS To see the examples, type: "get-help Remove-VApp -examples". For more information, type: "get-help Remove-VApp -detailed". For technical information, type: "get-help Remove-VApp -full". For online help, type: "get-help Remove-VApp -online"

Remove-VDPortGroup

NAME Remove-VDPortGroup

SYNOPSIS This cmdlet removes distributed port groups.

SYNTAX Remove-VDPortGroup [-VDPortGroup] <VDPortgroup[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes distributed port groups.

PARAMETERS

-VDPortGroup <VDPortgroup[]> Specifies the distributed port group that you want to remove.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDPortGroup -Name “MyVDPortGroup” | Remove-VDPortGroup

Removes the specified distributed port group from the vSphere distributed switch that it belongs to.

REMARKS To see the examples, type: “get-help Remove-VDPortGroup -examples”. For more information, type: “get-help Remove-VDPortGroup -detailed”. For technical information, type: “get-help Remove-VDPortGroup -full”. For online help, type: “get-help Remove-VDPortGroup -online”

Remove-VDSwitch

NAME Remove-VDSwitch

SYNOPSIS This cmdlet removes vSphere distributed switches.

SYNTAX Remove-VDSwitch [-VDSwitch] <VDSwitch[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes vSphere distributed switches.

PARAMETERS

-VDSwitch <VDSwitch[]> Specifies the vSphere distributed switches that you want to remove.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch -Name MyVDSwitch | Remove-VDSwitch

Removes the specified vSphere distributed switch.

REMARKS To see the examples, type: "get-help Remove-VDSwitch -examples". For more information, type: "get-help Remove-VDSwitch -detailed". For technical information, type: "get-help Remove-VDSwitch -full". For online help, type: "get-help Remove-VDSwitch -online"

Remove-VDSwitchPhysicalNetworkAdapter

NAME Remove-VDSwitchPhysicalNetworkAdapter

SYNOPSIS This cmdlet removes host physical network adapters from the vSphere distributed switches they are connected to.

SYNTAX Remove-VDSwitchPhysicalNetworkAdapter [-VMHostNetworkAdapter] <PhysicalNic[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes host physical network adapters from the vSphere distributed switches they are connected to.

PARAMETERS

-VMHostNetworkAdapter <PhysicalNic[]> Specifies the host physical network adapters that you want to remove from the vSphere distributed switch it is connected to.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMhost -Name "MyVMhost" | Get-VMHostNetworkAdapter -Physical -Name vmnic0 | Remove-VDSwitchPhysicalNetworkAdapter

Removes the specified host physical network adapter from the vSphere distributed switch that it is connected to.

REMARKS To see the examples, type: "get-help Remove-VDSwitchPhysicalNetworkAdapter -examples". For more information, type: "get-help Remove-VDSwitchPhysicalNetworkAdapter -detailed". For technical information, type: "get-help Remove-VDSwitchPhysicalNetworkAdapter -full". For online help, type: "get-help Remove-VDSwitchPhysicalNetworkAdapter -online"

Remove-VDSwitchPrivateVlan

NAME Remove-VDSwitchPrivateVlan

SYNOPSIS This cmdlet removes private VLAN configuration entries from vSphere distributed switches.

SYNTAX Remove-VDSwitchPrivateVlan [-VDSwitchPrivateVlan] <VDSwitchPrivateVlan[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes private VLAN configuration entries from vSphere distributed switches.

PARAMETERS

-VDSwitchPrivateVlan <VDSwitchPrivateVlan[]> Specifies the private VLAN configuration entry that you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitchPrivateVlan -VDSwitch "MyVDSwitch" -PrimaryVlanId 1,3,5 | Remove-VDSwitchPrivateVlan

Removes the private VLAN configuration entries with specified primary identities from a vSphere distributed switch named "MyVDSwitch".

REMARKS To see the examples, type: "get-help Remove-VDSwitchPrivateVlan -examples". For more information, type: "get-help Remove-VDSwitchPrivateVlan -detailed". For technical information, type: "get-help Remove-VDSwitchPrivateVlan -full". For online help, type: "get-help Remove-VDSwitchPrivateVlan -online"

Remove-VDSwitchVMHost

NAME Remove-VDSwitchVMHost

SYNOPSIS This cmdlet removes hosts from the specified vSphere distributed switches.

SYNTAX Remove-VDSwitchVMHost -VDSwitch <VDSwitch> -VMHost <VMHost[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes hosts from the specified vSphere distributed switches.

PARAMETERS

-VDSwitch <VDSwitch> Specifies the vSphere distributed switch from which you want to remove hosts.

-VMHost <VMHost[]> Specifies the hosts that you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch -Name "MySwitch" | Remove-VDSwitchVMHost -VMHost "VMHost1", "VMHost2"

Removes two hosts from the specified vSphere distributed switch.

REMARKS To see the examples, type: "get-help Remove-VDSwitchVMHost -examples". For more information, type: "get-help Remove-VDSwitchVMHost -detailed". For technical information, type: "get-help Remove-VDSwitchVMHost -full". For online help, type: "get-help Remove-VDSwitchVMHost -online"

Remove-VIPermission

NAME Remove-VIPermission

SYNOPSIS This cmdlet removes the specified permissions.

SYNTAX Remove-VIPermission [-Permission] <Permission[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified permissions.

PARAMETERS

-Permission <Permission[]> Specifies the permissions you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-VIPermission -Permission $permission -Confirm:$false

Removes the $permission permission without asking for confirmation.

REMARKS To see the examples, type: "get-help Remove-VIPermission -examples". For more information, type: "get-help Remove-VIPermission -detailed". For technical information, type: "get-help Remove-VIPermission -full". For online help, type: "get-help Remove-VIPermission -online"

Remove-VIProperty

NAME Remove-VIProperty

SYNOPSIS This cmdlet removes the extended properties from the specified object types.

SYNTAX Remove-VIProperty [-VIProperty] <VIProperty[]> [-WhatIf] [-Confirm] [<CommonParameters>]

Remove-VIProperty [-Name] <String[]> [-ObjectType] <String[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the extended properties from the specified object types. Changes take effect upon the next retrieval of the corresponding objects.

PARAMETERS

-VIProperty <VIProperty[]> Specifies the extended object properties you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Name <String[]> Specifies the names of the extended properties you want to remove.

-ObjectType <String[]> Specifies the object types to which the extended properties you want to remove belong.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-VIProperty -Name * -ObjectType *

Removes all custom properties.

————— Example 2 —————

C:PS>Remove-VIProperty -Name * -ObjectType VirtualMachine

Removes all custom properties for the VirtualMachine object type.

————— Example 3 —————

C:PS>Remove-VIProperty -Name OverallStatus, ConfigStatus -ObjectType VirtualMachine

Removes the OverallStatus and ConfigStatus for the VirtualMachine object type.

————— Example 4 —————

C:PS>Remove-VIProperty -Name OverallStatus -ObjectType VirtualMachine, Datacenter

Removes the OverallStatus property for the VirtualMachine and Datacenter types.

————— Example 5 —————

C:PS>Remove-VIProperty -Name *status* -ObjectType Virt*

Removes all properties that contain "status" in their names for object types with names that start with "Virt".

REMARKS To see the examples, type: "get-help Remove-VIProperty -examples". For more information, type: "get-help Remove-VIProperty -detailed". For technical information, type: "get-help Remove-VIProperty -full". For online help, type: "get-help Remove-VIProperty -online"

Remove-VIRole

NAME Remove-VIRole

SYNOPSIS This cmdlet removes the specified roles.

SYNTAX Remove-VIRole [-Role] <Role[]> [-Force] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified roles. To remove a role that is associated with a permission, you need to set the Force parameter to $true.

PARAMETERS

-Role <Role[]> Specifies the roles you want to remove.

-Force	Indicates that you want to remove the role even if it is associated with a permission.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VIRole -Server $server -Name "Customer*" | Remove-VIRole

Removes the roles with names that start with "Customer".

REMARKS To see the examples, type: "get-help Remove-VIRole -examples". For more information, type: "get-help Remove-VIRole -detailed". For technical information, type: "get-help Remove-VIRole -full". For online help, type: "get-help Remove-VIRole -online"

Remove-VirtualPortGroup

NAME Remove-VirtualPortGroup

SYNOPSIS This cmdlet removes the specified virtual port groups.

SYNTAX Remove-VirtualPortGroup [-VirtualPortGroup] <VirtualPortGroup[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified virtual port groups.

PARAMETERS

-VirtualPortGroup <VirtualPortGroup[]> Specifies the virtual port groups you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation..

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.112.234 -Name VirtualSwitch

$vportgroup = New-VirtualPortGroup -VirtualSwitch $vswitch -Name VPortGroup

Remove-VirtualPortGroup -VirtualPortGroup $vportgroup

Creates a new virtual switch named VirtualSwitch and a virtual ports group VPortGroup for this switch. Then removes the virtual ports group.

REMARKS To see the examples, type: “get-help Remove-VirtualPortGroup -examples”. For more information, type: “get-help Remove-VirtualPortGroup -detailed”. For technical information, type: “get-help Remove-VirtualPortGroup -full”. For online help, type: “get-help Remove-VirtualPortGroup -online”

Remove-VirtualSwitch

NAME Remove-VirtualSwitch

SYNOPSIS This cmdlet removes the specified virtual switches from their locations.

SYNTAX Remove-VirtualSwitch [-VirtualSwitch] <VirtualSwitch[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified virtual switches from their locations.

PARAMETERS

-VirtualSwitch <VirtualSwitch[]> Specifies the virtual switches you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.122.145 -Name VirtualSwitch

Remove-VirtualSwitch -VirtualSwitch $vswitch

Creates a new virtual switch named VirtualSwitch on the host with an IP address 10.23.122.145. Then removes the virtual switch.

REMARKS To see the examples, type: “get-help Remove-VirtualSwitch -examples”. For more information, type: “get-help Remove-VirtualSwitch -detailed”. For technical information, type: “get-help Remove-VirtualSwitch -full”. For online help, type: “get-help Remove-VirtualSwitch -online”

Remove-VirtualSwitchPhysicalNetworkAdapter

NAME Remove-VirtualSwitchPhysicalNetworkAdapter

SYNOPSIS This cmdlet removes the specified host physical NICs from the standard virtual switch.

SYNTAX Remove-VirtualSwitchPhysicalNetworkAdapter [-VMHostNetworkAdapter] <PhysicalNic[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified host physical NICs from the standard virtual switch.

PARAMETERS

-VMHostNetworkAdapter <PhysicalNic[]> Specifies the network adapters you want to remove.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMhost "myVMhost" | Get-VMHostNetworkAdapter -Physical -Name "vmnic1" | Remove-VirtualSwitchPhysicalNetworkAdapter

Removes a VMHost NIC from the virtual switch it is attached to.

REMARKS To see the examples, type: "get-help Remove-VirtualSwitchPhysicalNetworkAdapter -examples". For more information, type: "get-help Remove-VirtualSwitchPhysicalNetworkAdapter -detailed". For technical information, type: "get-help Remove-VirtualSwitchPhysicalNetworkAdapter -full". For online help, type: "get-help Remove-VirtualSwitchPhysicalNetworkAdapter -online"

Remove-VM

NAME Remove-VM

SYNOPSIS This cmdlet removes the specified virtual machines from the vCenter Server system.

SYNTAX Remove-VM [-DeletePermanently] [-RunAsync] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified virtual machines from the vCenter Server system. If the value of the DeletePermanently parameter is $true, the cmdlet not only removes the virtual machines from the inventory, but also deletes them from the disk.

PARAMETERS

-DeletePermanently Indicates that you want to delete the virtual machines not only from the inventory, but from the datastore.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-VM <VirtualMachine[]> Specifies the virtual machines you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-VM VM -DeletePermanently

Removes the VM virtual machine and deletes its files from the ESX host.

REMARKS To see the examples, type: "get-help Remove-VM -examples". For more information, type: "get-help Remove-VM -detailed". For technical information, type: "get-help Remove-VM -full". For online help, type: "get-help Remove-VM -online"

Remove-VMGuestRoute

NAME Remove-VMGuestRoute

SYNOPSIS This cmdlet removes the specified routes from the routing table of their corresponding virtual machines.

SYNTAX Remove-VMGuestRoute [-VMGuestRoute] <VMGuestRoute[]> [-ToolsWaitSecs <Int32>] [-GuestPassword <SecureString>] [-GuestUser <String>] [-GuestCredential <PSCredential>] [-HostPassword <SecureString>] [-HostUser <String>] [-HostCredential <PSCredential>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use Invoke-VMGuestScript instead.

This cmdlet removes the specified routes from the routing table of their corresponding virtual machines. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Execute and VirtualMachine.GuestOperations.Modify privileges.

PARAMETERS

-VMGuestRoute <VMGuestRoute[]> Specifies the route you want to remove.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from VMware Tools. If a non-positive value is provided, the system waits infinitely long time.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest OS.

-GuestUser <String> Specifies the user name you want to use for authenticating with the guest OS.

-GuestCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the guest OS. Do not use this parameter if the GuestUser and GuestPassword parameters are used.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host. Do not use this parameter if the HostUser and HostPassword parameters are used. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Remove-VMGuestRoute -GuestUser user -GuestPassword pass2 -VMGuestRoute $route

Removes the guest route stored in the $route variable.

REMARKS To see the examples, type: “get-help Remove-VMGuestRoute -examples”. For more information, type: “get-help Remove-VMGuestRoute -detailed”. For technical information, type: “get-help Remove-VMGuestRoute -full”. For online help, type: “get-help Remove-VMGuestRoute -online”

Remove-VMHost

NAME Remove-VMHost

SYNOPSIS This cmdlet removes the specified hosts from the inventory.

SYNTAX Remove-VMHost [-VMHost] <VMHost[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified hosts from the inventory.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myServer = Connect-VIServer -Server 10.23.112.235 Get-VMHost -Server $myServer -Location MyDatacenter1 | Remove-VMHost -Confirm:$false

Removes a specified VM host from a vCenter Server system without asking for a confirmation.

REMARKS To see the examples, type: "get-help Remove-VMHost -examples". For more information, type: "get-help Remove-VMHost -detailed". For technical information, type: "get-help Remove-VMHost -full". For online help, type: "get-help Remove-VMHost -online"

Remove-VMHostAccount

NAME Remove-VMHostAccount

SYNOPSIS This cmdlet removes the specified host accounts.

SYNTAX Remove-VMHostAccount [-HostAccount] <HostAccount[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified host accounts . These can be HostGroupAccount objects, HostUserAccount objects, or both.

PARAMETERS

-HostAccount <HostAccount[]> Specifies the host accounts you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostAccount -Group -ID user | Remove-VMHostAccount -Confirm

Removes the group account with ID "user". Asks for confirmation before running the command.

REMARKS To see the examples, type: "get-help Remove-VMHostAccount -examples". For more information, type: "get-help Remove-VMHostAccount -detailed". For technical information, type: "get-help Remove-VMHostAccount -full". For online help, type: "get-help Remove-VMHostAccount -online"

Remove-VMHostNetworkAdapter

NAME Remove-VMHostNetworkAdapter

SYNOPSIS This cmdlet removes the specified host network adapters.

SYNTAX Remove-VMHostNetworkAdapter [-Nic] <HostVirtualNic[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified host network adapters.

PARAMETERS

-Nic <HostVirtualNic[]> Specifies the network adapters you want to remove.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$network = Get-VMHostNetwork

Remove-VMHostNetworkAdapter $network.VirtualNic[0] -Confirm

Removes the first virtual network adapter of the host.

REMARKS To see the examples, type: “get-help Remove-VMHostNetworkAdapter -examples”. For more information, type: “get-help Remove-VMHostNetworkAdapter -detailed”. For technical information, type: “get-help Remove-VMHostNetworkAdapter -full”. For online help, type: “get-help Remove-VMHostNetworkAdapter -online”

Remove-VMHostNtpServer

NAME Remove-VMHostNtpServer

SYNOPSIS This cmdlet removes the specified NTP servers from the NTP server list of the specified hosts.

SYNTAX Remove-VMHostNtpServer [-NtpServer] <String[]> [-VMHost] <VMHost[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified NTP servers from the NTP server list of the specified hosts.

PARAMETERS

-NtpServer <String[]> Specifies the NTP servers you want to remove from the NTP servers list of the specified host.

-VMHost <VMHost[]> Specifies the host whose NTP servers you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Remove-VmHostNtpServer -NtpServer 192.168.1.5 -VMHost $vmhost -Confirm

Removes the NTP server with an IP address 192.168.1.5 from the virtual machine hosts stored in the $vmhost variable.

————— Example 2 —————

C:PS>Remove-VmHostNtpServer -NtpServer "old-ntp-server.com" -VMHost (Get-VMHost) -Confirm

Removes the NTP server with a domain name "old-ntp-server.com" from the virtual machine hosts pipelined through the Get-VMHost cmdlet.

REMARKS To see the examples, type: "get-help Remove-VMHostNtpServer -examples". For more information, type: "get-help Remove-VMHostNtpServer -detailed". For technical information, type: "get-help Remove-VMHostNtpServer -full". For online help, type: "get-help Remove-VMHostNtpServer -online"

Remove-VMHostProfile

NAME Remove-VMHostProfile

SYNOPSIS This cmdlet removes the specified host profiles.

SYNTAX Remove-VMHostProfile [-Profile] <VMHostProfile[]> [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Remove-VMHostProfile -Entity <InventoryItem[]> [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes the specified host profiles. If the Entity parameter is provided, the cmdlet removes the profile association for the specified entity. Otherwise, the cmdlet removes the profile object.

PARAMETERS

-Profile <VMHostProfile[]> Specifies the host profiles you want to remove.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Entity <InventoryItem[]> Specifies the host or cluster whose host profile association you want to remove.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostProfile -Name Profile | Remove-VMHostProfile -Confirm:$false

Deletes the Profile host profile.

REMARKS To see the examples, type: "get-help Remove-VMHostProfile -examples". For more information, type: "get-help Remove-VMHostProfile -detailed". For technical information, type: "get-help Remove-VMHostProfile -full". For online help, type: "get-help Remove-VMHostProfile -online"

Remove-VMHostRoute

NAME Remove-VMHostRoute

SYNOPSIS This cmdlet removes host routes.

SYNTAX Remove-VMHostRoute [-VMHostRoute] <VMHostRoute[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet removes host routes.

PARAMETERS

-VMHostRoute <VMHostRoute[]> Specifies the host routes you want to remove.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$destIpList = ('192.168.111.101', '192.168.111.102')

$routes = Get-VMHostRoute -VMHost ($script:vmhost1, $script:vmhost2) | where {$destIpList -contains $_.Destination.IPAddressToString}

Remove-VMHostRoute -VMHostRoute $routes -Confirm:$false

Removes the host routes that have the specified destination IP addresses.

REMARKS To see the examples, type: "get-help Remove-VMHostRoute -examples". For more information, type: "get-help Remove-VMHostRoute -detailed". For technical information, type: "get-help Remove-VMHostRoute -full". For online help, type: "get-help Remove-VMHostRoute -online"

CHAPTER 24

Restart Commands

This page contains details on **Restart** commands.

Restart-VM

NAME Restart-VM

SYNOPSIS This cmdlet restarts the specified virtual machines.

SYNTAX Restart-VM [-RunAsync] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet restarts the specified virtual machines.

PARAMETERS

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-VM <VirtualMachine[]> Specifies the virtual machines you want to restart.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Restart-VM -VM VM -RunAsync -Confirm

Restarts the VM virtual machine after user confirmation. The cmdlet returns without waiting for the task to complete.

REMARKS To see the examples, type: "get-help Restart-VM -examples". For more information, type: "get-help Restart-VM -detailed". For technical information, type: "get-help Restart-VM -full". For online help, type: "get-help Restart-VM -online"

Restart-VMGuest

NAME Restart-VMGuest

SYNOPSIS This cmdlet restarts the virtual machine guest operating systems.

SYNTAX Restart-VMGuest [[-VM] <VirtualMachine[]>] [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Restart-VMGuest [[-Guest] <VMGuest[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet restarts the virtual machine guest operating systems.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines whose operating systems you want to restart. The specified virtual machines must have VMware Tools installed.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Guest <VMGuest[]> Specifies the virtual machine guest operating systems you want to restart.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VM VM | Restart-VMGuest

Restarts the guest OS of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Restart-VMGuest -examples". For more information, type: "get-help Restart-VMGuest -detailed". For technical information, type: "get-help Restart-VMGuest -full". For online help, type: "get-help Restart-VMGuest -online"

Restart-VMHost

NAME Restart-VMHost

SYNOPSIS This cmdlet restarts the specified hosts.

SYNTAX Restart-VMHost [-VMHost] <VMHost[]> [-Force] [-Evacuate] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet restarts the specified hosts.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to restart.

-Force Indicates that you want to restart the hosts even if they are not in a maintenance mode.

-Evacuate Indicates that vCenter Server automatically reregisters the virtual machines that are compatible for reregistration. If they are not compatible, they remain on the rebooted host. If there are powered-on virtual machines that cannot be reregistered, the operation waits until they are powered off manually. The Evacuate parameter is valid only if the cmdlet is run against a vCenter Server system and the host is in a DRS-enabled cluster.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Restart-VMHost 10.23.112.235 -RunAsync -Confirm

Restarts the specified host after user confirmation. The cmdlet returns without waiting for the task to complete.

REMARKS To see the examples, type: "get-help Restart-VMHost -examples". For more information, type: "get-help Restart-VMHost -detailed". For technical information, type: "get-help Restart-VMHost -full". For online help, type: "get-help Restart-VMHost -online"

Restart-VMHostService

NAME Restart-VMHostService

SYNOPSIS This cmdlet restarts the specified host services.

SYNTAX Restart-VMHostService [-HostService] <HostService[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet restarts the specified host services.

PARAMETERS

-HostService <HostService[]> Specifies the host service you want to restart.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Restart-VMHostService -Service $vmHostService -Confirm:$false

Restarts a host service.

REMARKS To see the examples, type: "get-help Restart-VMHostService -examples". For more information, type: "get-help Restart-VMHostService -detailed". For technical information, type: "get-help Restart-VMHostService -full". For online help, type: "get-help Restart-VMHostService -online"

CHAPTER 25

Set Commands

This page contains details on **Set** commands.

Set-AdvancedSetting

NAME Set-AdvancedSetting

SYNOPSIS This cmdlet modifies the specified advanced setting.

SYNTAX Set-AdvancedSetting [-AdvancedSetting] <AdvancedSetting[]> [-Value] <Object> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified advanced setting.

PARAMETERS

-AdvancedSetting <AdvancedSetting[]> Specifies the advanced setting you want to modify.

-Value <Object> Specifies a new value for the advanced setting.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AdvancedSetting -Entity (Get-Cluster -Name Cluster) -Name SettingName | Set-AdvancedSetting -Value NewValue

Changes the value of the advanced setting SettingName of the Cluster cluster.

————— Example 2 —————

C:PS>Get-AdvancedSetting -Entity Server -Name 'mail.smtp.server' | Set-AdvancedSetting -Value 'test.vmware.com'

Changes the value of the advanced setting mail.smtp.server of the specified server to test.vmware.com.

REMARKS To see the examples, type: "get-help Set-AdvancedSetting -examples". For more information, type: "get-help Set-AdvancedSetting -detailed". For technical information, type: "get-help Set-AdvancedSetting -full". For online help, type: "get-help Set-AdvancedSetting -online"

Set-AlarmDefinition

NAME Set-AlarmDefinition

SYNOPSIS This cmdlet modifies the specified alarm definitions.

SYNTAX Set-AlarmDefinition [-AlarmDefinition] <AlarmDefinition[]> [-ActionRepeatMinutes <Int32>] [-Description <String>] [-Enabled <Boolean>] [-Name <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified alarm definitions.

PARAMETERS

-AlarmDefinition <AlarmDefinition[]> Specifies the alarm definition you want to modify.

-ActionRepeatMinutes <Int32> Specifies a time period in minutes to define how often the alarm action repeats if the alarm is active.

-Description <String> Specifies a new description for the alarm definition.

-Enabled [<Boolean>] Indicates that the alarm definition is enabled.

-Name <String> Specifies a new name for the alarm definition.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-AlarmDefinition -Name 'alarms' | Set-AlarmDefinition -ActionRepeatMinutes ($_.ActionRepeatMinutes + 1)

Increase all selected alarms action repeat minutes.

————— Example 2 —————

C:PS>Get-AlarmDefinition -Name 'alarm' | foreach {$_ | Set-AlarmDefinition -Name 'alarm new name' -Description 'new description' -Enabled:$true}

Changes the name, description, and the Enabled flag of the selected alarms.

REMARKS To see the examples, type: "get-help Set-AlarmDefinition -examples". For more information, type: "get-help Set-AlarmDefinition -detailed". For technical information, type: "get-help Set-AlarmDefinition -full". For online help, type: "get-help Set-AlarmDefinition -online"

Set-Annotation

NAME Set-Annotation

SYNOPSIS This cmdlet modifies the value of a custom attribute.

SYNTAX Set-Annotation [-Entity] <InventoryItem[]> [-CustomAttribute] <CustomAttribute> [-Value] <String> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the value of a custom attribute that applies to one or more inventory items.

PARAMETERS

-Entity <InventoryItem[]> Specifies the entities to which the new annotation value applies.

-CustomAttribute <CustomAttribute> Specifies the custom attribute whose annotation you want to change.

-Value <String> Specifies a new value for the annotation.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-Annotation -Entity $vmhost -CustomAttribute "PhysicalLocation" -Value Office

Modifies the annotation of the PhysicalLocation custom attribute for the host stored in the $vmhost variable.

————— Example 2 —————

C:PS>Get-Cluster Cluster | Set-Annotation -CustomAttribute "PhysicalLocation" -Value California

Modifies the annotation of the PhysicalLocation custom attribute for the Cluster cluster.

REMARKS To see the examples, type: "get-help Set-Annotation -examples". For more information, type: "get-help Set-Annotation -detailed". For technical information, type: "get-help Set-Annotation -full". For online help, type: "get-help Set-Annotation -online"

Set-CDDrive

NAME Set-CDDrive

SYNOPSIS This cmdlet modifies the configuration of a virtual CD drive.

SYNTAX Set-CDDrive [-CD] <CDDrive[]> [-IsoPath <String>] [-HostDevice <String>] [-NoMedia] [-StartConnected <Boolean>] [-Connected <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet updates a virtual CD drive. If an ISO location is provided, sets the CD to point to the ISO. Changes the StartConnected and Connected flags if StartConnected and/or Connected is set. If NoMedia parameter is set to $true, removes the CD drive's media backing and disconnects it. Note that the Connected parameter can be specified only if the corresponding virtual machine is powered on.

PARAMETERS

-CD <CDDrive[]> Specifies the virtual CD drive you want to configure.

-IsoPath <String> Specifies the datastore path to the ISO (CD image) file that backs the virtual CD drive. Do not use this parameter when the HostDevice and NoMedia parameters are specified.

-HostDevice <String> Specifies the path to the CD drive on the host which backs this virtual CD drive. Do not use this parameter when the ISOPath and NoMedia parameters are specified.

-NoMedia Indicates that you want to detach from the CD drive any type of connected media - ISO from datastore or host device. Do not use this parameter when the ISOPath or HostDevice parameters are specified.

-StartConnected [<Boolean>] Indicates that the virtual CD drive starts connected when the virtual machine associated with it powers on.

-Connected [<Boolean>] Indicates that the virtual CD drive is connected after its creation. This parameter can be specified only if the corresponding virtual machine is powered on.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$cd = New-CDDrive -VM VM -ISOPath "[sof-20666-esx:storage1] ISOtestISO.iso"

Set-CDDrive -CD $cd -NoMedia

Creates a CD drive on the VM virtual machine and attaches testISO.iso, previously uploaded. Then disconnects the ISO.

REMARKS To see the examples, type: "get-help Set-CDDrive -examples". For more information, type: "get-help Set-CDDrive -detailed". For technical information, type: "get-help Set-CDDrive -full". For online help, type: "get-help Set-CDDrive -online"

Set-Cluster

NAME Set-Cluster

SYNOPSIS This cmldlet modifies the configuration of a cluster.

SYNTAX Set-Cluster [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-VMSwapfilePolicy <VMSwapfilePolicy>] [-Cluster] <Cluster[]> [[-Name] <String>] [-HAEnabled <Boolean>] [-HAAdmissionControlEnabled <Boolean>] [-HAFailoverLevel <Int32>] [-DrsEnabled

<Boolean>] [-DrsMode <DrsMode>] [-DrsAutomationLevel <DrsAutomationLevel>] [-VsanEnabled <Boolean>] [-VsanDiskClaimMode <VsanDiskClaimMode>] [-Profile <VMHostProfile>] [-EVCMode <String>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of a cluster. HAEnabled is automatically set to $true if some of the HA settings, HAAdmissionControlEnabled, HAFailoverLevel, HARestartPriority, HAIsolationResponse, are specified. DrsEnabled is automatically set to $true if some of the DRS settings, DrsAutomationLevel, DrsMode, are specified.

PARAMETERS

-HARestartPriority <HARestartPriority> Specifies the cluster HA restart priority. The valid values are Disabled, Low, Medium, and High. VMware HA is a feature that detects failed virtual machines and automatically restarts them on alternative ESX/ESXi hosts. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-HAIsolationResponse <HAIsolationResponse> Specifies whether the virtual machine should be powered off if a host determines that it is isolated from the rest of the compute resource. The valid values are PowerOff and DoNothing. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-VMSwapfilePolicy <VMSwapfilePolicy> Specifies the swapfile placement policy. The following values are valid:

InHostDataStore - Store the swapfile in the datastore specified by the VMSwapfileDatastoreID property of the virtual machine host. If the VMSwapfileDatastoreID property is not set or indicates a datastore with insufficient free space, the swapfile is stored in the same directory as the virtual machine. This setting might degrade the VMotion performance.

WithVM - Store the swapfile in the same directory as the virtual machine.

-Cluster <Cluster[]> Specifies the name of the cluster you want to configure.

-Name <String> Specifies a new name for the cluster.

-HAEnabled [<Boolean>] Indicates that VMware High Availability is enabled.

-HAAdmissionControlEnabled [<Boolean>] Indicates that the virtual machines in the cluster will not start if they violate availability constraints.

-HAFailoverLevel <Int32> Specifies a failover level. This is the number of physical host failures that can be tolerated without impacting the ability to meet minimum thresholds for all running virtual machines. The valid values range from one to four.

-DrsEnabled [<Boolean>] Indicates that VMware DRS (Distributed Resource Scheduler) is enabled.

-DrsMode <DrsMode> This parameter is deprecated and scheduled for removal. Use the DrsAutomationLevel parameter instead.

Specifies a DRS (Distributed Resource Scheduler) mode. The valid values are FullyAutomated, Manual, and PartiallyAutomated.

-DrsAutomationLevel <DrsAutomationLevel> Specifies a DRS (Distributed Resource Scheduler) automation level. The valid values are FullyAutomated, Manual, and PartiallyAutomated.

-VsanEnabled [<Boolean>] Specifies whether the Virtual SAN feature is enabled on this cluster.

-VsanDiskClaimMode <VsanDiskClaimMode> Specifies the mode by which disks are claimed by the Virtual SAN.

-Profile <VMHostProfile> Specifies a host profile you want to associate with the cluster. If the value of this parameter is $null, the current profile association is removed.

-EVCMode <String> Specifies the VMware Enhanced vMotion Compatibility (EVC) mode of the newly created cluster. If not specified or set to $null, EVC is disabled.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-Cluster -Name "MyClusterName" | Set-Cluster -Name "NewClusterName" -HAEnabled:$true -HAAdmissionControlEnabled:$true -HAFailoverLevel 2 -VMSwapfilePolicy "InHostDatastore" -HARestartPriority "Low" -HAIsolationResponse "PowerOff"

Renames the "MyClusterName" cluster to "NewClusterName" and changes its VMware HA (Hgh Availability) settings.

———— Example 2 ————

C:PS>Set-Cluster -Cluster "MyClusterName" -DRSEnabled:$true -DRSAutomationLevel "Manual"

Changes the VMware DRS (Distributed Resource Scheduler) settings of the "MyClusterName" cluster.

———— Example 3 ————

C:PS>Set-Cluster -Cluster "MyClusterName" -EVCMode "intel-nehalem"

Changes the VMware EVC (Enhanced vMotion Compatibility) settings of the "MyClusterName" cluster.

REMARKS To see the examples, type: "get-help Set-Cluster -examples". For more information, type: "get-help Set-Cluster -detailed". For technical information, type: "get-help Set-Cluster -full". For online help, type: "get-help Set-Cluster -online"

Set-CustomAttribute

NAME Set-CustomAttribute

SYNOPSIS This cmdlet renames a custom attribute.

SYNTAX Set-CustomAttribute [-CustomAttribute] <CustomAttribute[]> [-Name] <String> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet renames a custom attribute.

PARAMETERS

-CustomAttribute <CustomAttribute[]> Specifies the custom attribute you want to rename.

-Name <String> Specifies a new name for the custom attribute.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-CustomAttribute -Name CreationDetails -CustomAttribute CreationDate

Renames a custom attribute from CreationDate to CreationDetails.

————— Example 2 —————

C:PS>Get-CustomAttribute -Name CreationDate -Server server1, server2 | Set-CustomAttribute -Name CreationDetails

Renames the custom attributes retrieved from the specified vSphere servers from CreationDate to CreationDetails.

REMARKS To see the examples, type: “get-help Set-CustomAttribute -examples”. For more information, type: “get-help Set-CustomAttribute -detailed”. For technical information, type: “get-help Set-CustomAttribute -full”. For online help, type: “get-help Set-CustomAttribute -online”

Set-Datacenter

NAME Set-Datacenter

SYNOPSIS This cmdlet modifies the properties of the specified datacenter.

SYNTAX Set-Datacenter [-Datacenter] <Datacenter[]> [-Name] <String> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified datacenter.

PARAMETERS

-Datacenter <Datacenter[]> Specifies the datacenter whose properties you want to modify.

-Name <String> Specifies a new name for the datacenter.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-Datacenter -Datacenter Datacenter1 -Name Datacenter2

Renames the Datacenter1 to Datacenter2.

REMARKS To see the examples, type: "get-help Set-Datacenter -examples". For more information, type: "get-help Set-Datacenter -detailed". For technical information, type: "get-help Set-Datacenter -full". For online help, type: "get-help Set-Datacenter -online"

Set-Datastore

NAME Set-Datastore

SYNOPSIS This cmdlet modifies the properties of the specified datastore.

SYNTAX Set-Datastore [-Datastore] <Datastore[]> [[-Name] <String>] [-CongestionThresholdMillisecond <Int32>] [-StorageIOControlEnabled <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-Datastore [-Datastore] <Datastore[]> -MaintenanceMode <Boolean> [-EvacuateAutomatically] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified datastore. You can use the following characters in a path, but not in a datastore name: slash (/), backslash (), and percent (%).

PARAMETERS

-Datastore <Datastore[]> Specifies the datastore whose properties you want to change.

-Name <String> Specifies a new name for the datastore. Do not use slash (/), backslash (), and percent (%) characters in datastore names.

-CongestionThresholdMillisecond <Int32> Specifies the latency period beyond which the storage array is considered congested. The range of this value is between 10 to 100 milliseconds.

-StorageIOControlEnabled [<Boolean>] Indicates whether you want to enable the IO control.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-MaintenanceMode [<Boolean>] Specifies whether you want to put the datastore in maintenance mode. The operation completes when no virtual machines are present and no provisioning processes are running on the datastore.

-EvacuateAutomatically Specifies whether you want to automatically migrate all virtual machines to another datastore if the value of MaintenanceMode is $true. When the Storage DRS automation level is set to Fully Automated, you do not need to specify the EvacuateAutomatically parameter because Storage DRS will migrate all virtual machines automatically.

If EvacuateAutomatically is specified, the SDRS placement and migration recommendations are automatically applied. If SDRS generates cluster DRS faults, an error report is displayed and the operation is cancelled. The report contains information about each datastore cluster DRS fault.

If EvacuateAutomatically is not specified, an error report is displayed and the operation is cancelled. The error report contains information about each SDRS recommendation. If SDRS generates cluster DRS faults, an error report is displayed and the operation is cancelled. The error report contains information about each fault.

If EvacuateAutomatically is explicitly set to false, the cmdlet blocks execution without displaying an error message. If SDRS generates datastore cluster DRS faults, the cmdlet stops responding and an error report is displayed. The report contains information about each cluster DRS fault.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Datastore -Name Datastore1 | Set-Datastore -Name Datastore2

Renames the Datastore1 datastore to Datastore2.

————— Example 2 —————

C:PS>Set-Datastore $datastore1, $datastore2 -StorageIOControlEnabled $true -CongestionThresholdMillisecond 80

Enables the Storage IO Control and set a congestion threshold of 80 milliseconds for the specified datastores.

————— Example 3 —————

C:PS>Get-Datastore -Name 'MyDatastore1' | Set-Datastore -MaintenanceMode $true -EvacuateAutomatically

Puts the MyDatastore1 datastore in maintenance mode and specifies that all virtual machines on the datastore will be automatically migrated to another datastore.

REMARKS To see the examples, type: "get-help Set-Datastore -examples". For more information, type: "get-help Set-Datastore -detailed". For technical information, type: "get-help Set-Datastore -full". For online help, type: "get-help Set-Datastore -online"

Set-DatastoreCluster

NAME Set-DatastoreCluster

SYNOPSIS This cmdlet modifies the configuration of the specified datastore cluster.

SYNTAX Set-DatastoreCluster -DatastoreCluster <DatastoreCluster[]> [-IOLatencyThresholdMillisecond <Int32>] [-IOLoadBalanceEnabled <Boolean>] [-Name <String>] [-SdrsAutomationLevel <DrsAutomationLevel>] [-SpaceUtilizationThresholdPercent <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of the specified datastore cluster.

PARAMETERS

-DatastoreCluster <DatastoreCluster[]> Specifies the datastore cluster that you want to configure.

-IOLatencyThresholdMillisecond <Int32> Specifies the maximum I/O latency in milliseconds allowed before Storage DRS is triggered for the datastore cluster. The parameter accepts values in the range of 5 to 100. If the value of IOLoadBalancing is $false, the setting for the I/O latency threshold is not applied.

-IOLoadBalanceEnabled [<Boolean>] Specifies whether I/O load balancing is enabled for the datastore cluster. If the value is $false, I/O load balancing is disabled and the settings for the I/O latency threshold and utilized space threshold are not applied.

-Name <String> Specifies a new name for the datastore cluster.

-SdrsAutomationLevel <DrsAutomationLevel> Specifies the Storage DRS automation level for the datastore cluster. This parameter accepts Disabled, Manual, and FullyAutomated values.

-SpaceUtilizationThresholdPercent <Int32> Specifies the maximum percentage of consumed space allowed before Storage DRS is triggered for the datastore cluster. The parameter accepts values in the range of 50 to 100. If the value of IOLoadBalancing is $false, the setting for the utilized space threshold is not applied.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-DatastoreCluster -DatastoreCluster MyDatastoreCluster1 -Name 'MyDatastoreCluster2'

Changes the name of the specified datastore cluster.

————— Example 2 —————

C:PS>Set-DatastoreCluster -DatastoreCluster MyDatastoreCluster -IOLatencyThresholdMillisecond 5

Sets the maximum I/O latency in milliseconds allowed before Storage DRS is triggered for the specified datastore cluster to 5 milliseconds.

————— Example 3 —————

C:PS>Set-DatastoreCluster -DatastoreCluster MyDatastoreCluster - SdrsAutomationLevel FullyAutomated

Changes the Storage DRS automation level of the specified datastore cluster to Fully Automated.

REMARKS To see the examples, type: "get-help Set-DatastoreCluster -examples". For more information, type: "get-help Set-DatastoreCluster -detailed". For technical information, type: "get-help Set-DatastoreCluster -full". For online help, type: "get-help Set-DatastoreCluster -online"

Set-DrsRule

NAME Set-DrsRule

SYNOPSIS This cmdlet modifies an existing DRS rule.

SYNTAX Set-DrsRule [[-Enabled] <Boolean>] [-Rule] <DrsRule[]> [-Name <String>] [-VM <VirtualMachine[]>] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies an existing DRS rule. Each rule defines the virtual machines that can run on the same host (affinity rule) or must run on different hosts (anti-affinity rule).

PARAMETERS

-Enabled [<Boolean>] Indicates that the DRS rule is enabled.

-Rule <DrsRule[]> Specifies the DRS rule you want to modify.

-Name <String> Specifies a new name for the DRS rule.

-VM <VirtualMachine[]> Specifies the virtual machines that can be referenced by the new DRS rule.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vm = Get-VM DrsRuleVM1*

Set-DrsRule -Rule $affinityRule -VM $vm -Enabled $true;

Updates the list of virtual machines that might be referenced by the DRS rule stored in the $affinityRule variable and enables the rule.

REMARKS To see the examples, type: "get-help Set-DrsRule -examples". For more information, type: "get-help Set-DrsRule -detailed". For technical information, type: "get-help Set-DrsRule -full". For online help, type: "get-help Set-DrsRule -online"

Set-FloppyDrive

NAME Set-FloppyDrive

SYNOPSIS This cmdlet modifies the configuration of the specified virtual floppy drive.

SYNTAX Set-FloppyDrive [-Floppy] <FloppyDrive[]> [-FloppyImagePath <String>] [-HostDevice <String>] [-NoMedia] [-StartConnected <Boolean>] [-Connected <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of the specified virtual floppy drive. If a floppy image path is provided, the cmdlet sets the floppy drive to point to the image. Also, the cmdlet updates the StartConnected and Connected properties. If the value of the NoMedia parameter is $true, the cmdlet removes the floppy drive's media backing and disconnects it. The FloppyImagePath, HostDevice, and NoMedia parameters cannot be used together. The Connected parameter can be specified only if the corresponding virtual machine is powered on.

PARAMETERS

-Floppy <FloppyDrive[]> Specifies the virtual floppy drive you want to configure.

-FloppyImagePath <String> Specifies the datastore path to the floppy image file that backs the virtual floppy drive. Do not use this parameter when the HostDevice and NoMedia parameters are specified.

-HostDevice <String> Specifies the path to the floppy drive on the host that backs this virtual floppy drive. Do not use this parameter when the FloppyImagePath and NoMedia parameters are specified.

-NoMedia Indicates that the floppy drive is to have no media (similar to removing the floppy from a physical drive). Do not use this parameter when the FloppyImagePath and HostDevice parameters are specified.

-StartConnected [<Boolean>] If the value is $true, the virtual floppy drive starts connected when its associated virtual machine powers on. If the value is $false, it starts disconnected.

-Connected [<Boolean>] If the value is $true, the virtual floppy drive is connected after its creation. If the value is $false, the floppy drive is disconnected. This parameter can be specified only if the corresponding virtual machine is powered on.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-FloppyDrive -Floppy $floppy -StartConnected:$true

Sets a floppy to start connected.

REMARKS To see the examples, type: "get-help Set-FloppyDrive -examples". For more information, type: "get-help Set-FloppyDrive -detailed". For technical information, type: "get-help Set-FloppyDrive -full". For online help, type: "get-help Set-FloppyDrive -online"

Set-Folder

NAME Set-Folder

SYNOPSIS This cmdlet modifies the properties of the specified folder.

SYNTAX Set-Folder -Folder <Folder[]> [-Name] <String> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified folder.

PARAMETERS

-Folder <Folder[]> Specifies the folder whose properties you want to change.

-Name <String> Specifies a new name for the folder.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Folder -Name "testFolder" | Set-Folder -Name "NewFolderName"

Renames the "testFolder" folder to "NewFolderName".

REMARKS To see the examples, type: "get-help Set-Folder -examples". For more information, type: "get-help Set-Folder -detailed". For technical information, type: "get-help Set-Folder -full". For online help, type: "get-help Set-Folder -online"

Set-HardDisk

NAME Set-HardDisk

SYNOPSIS This cmdlet modifies the properties of the specified virtual hard disk.

SYNTAX Set-HardDisk [-HardDisk] <HardDisk[]> [[-CapacityKB] <Int64>] [-CapacityGB <Decimal>] [[-Persistence] <String>] [[-Datastore] <Datastore>] [-StorageFormat <VirtualDiskStorageFormat>] [-Controller <ScsiController>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-HardDisk [-HardDisk] <HardDisk[]> [[-CapacityKB] <Int64>] [-CapacityGB <Decimal>] [[-Persistence] <String>] [[-Datastore] <Datastore>] [-StorageFormat <VirtualDiskStorageFormat>] [-Controller <ScsiController>] [-Server <VIServer[]>] [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-GuestCredential <PSCredential>] [-GuestUser <String>] [-GuestPassword <SecureString>] [-ToolsWaitSecs <Int32>] [-HelperVM <VirtualMachine>] [-Partition <String>] [-ResizeGuestPartition] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-HardDisk [-HardDisk] <HardDisk[]> [-Inflate] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-HardDisk [-HardDisk] <HardDisk[]> [-ZeroOut] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified virtual hard disk. You can change the size and the persistence type, and inflate or expand the specified virtual hard disk. Do not use the Inflate parameter at the same time with the Persistence and CapacityGB parameters. If a helper virtual machine is used, all virtual machines associated with the disk and the helper virtual machine should be powered off before expanding the disk. When you resize more than one disks using a helper virtual machine, the disks are resized one by one causing the helper machine to power on and off for each virtual machine and this might slow the cmdlet performance. For a list of supported operating systems, see the PowerCLI User's Guide.

PARAMETERS

-HardDisk <HardDisk[]> Specifies the virtual hard disk you want to configure.

-CapacityKB <Int64> This parameter is obsolete. Use CapacityGB instead. Specifies the updated capacity of the virtual disk in kilobytes (KB). If you are connected to a vCenter Server 2.0 or ESX 3.0 server, the size of the disk cannot be changed and the CapacityKB parameter is discarded. If you are connected to a vCenter Server 2.5 or ESX 3.5 server, the size of the disk can only be increased and the CapacityKB parameter is discarded if its value is less than the current disk size.

-CapacityGB <Decimal> Specifies the updated capacity of the virtual disk in gigabytes (GB). If you are connected to a vCenter Server 2.0 or ESX 3.0 server, the size of the disk cannot be changed and the CapacityGB parameter is discarded. If you are connected to a vCenter Server 2.5 or ESX 3.5 server, the size of the disk can only be increased and the CapacityGB parameter is discarded if its value is less than the current disk size.

-Persistence <String> Specifies the disk persistence mode. The valid values are Persistent, NonPersistent, IndependentPersistent, IndependentNonPersistent, and Undoable. This parameter is supported only when the disk type is rawVirtual or flat. The NonPersistent and Undoable values are deprecated and scheduled for removal. Their usage is not recommended because they do not work with snapshots and are not supported on ESX 3.5 and later.

-Datastore <Datastore> Specifies the datastore to which you want to move the specified hard disk. Moving a hard disk attached to a virtual machine to a different datastore is only supported on vCenter Server.

-StorageFormat <VirtualDiskStorageFormat> Specifies the storage format of the relocated hard disk. This parameter is applicable only when moving a virtual machine disk to a different datastore, using the Datastore parameter. This parameter accepts Thin, Thick, and EagerZeroedThick values.

-Controller <ScsiController> Specifies a SCSI controller to which you want to attach the hard disk.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-HostCredential <PSCredential> Specifies the PSCredential object that contains the credentials you want to use for authenticating with the host.

-HostUser <String> Specifies the username you want to use for authenticating with the host.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host.

-GuestCredential <PSCredential> Specifies the PSCredential object that contains the credentials you want to use for authenticating with the guest operating system.

-GuestUser <String> Specifies the username you want to use for authenticating with the guest operating system.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest operating system.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from VMware Tools. If a non-positive value is provided, the system waits infinitely long time.

-HelperVM <VirtualMachine> Specifies a helper virtual machine you want to use when expanding a Windows virtual machine system disk. LVM (logical volume manager) for Linux is not supported and Linux guest system disks cannot be expanded. When a helper virtual machine is used, all virtual machines associated with the disk and the helper virtual machine must be powered off before expanding the disk. When you resize more than one disks using a helper virtual machine, the disks are resized one by one causing the helper machine to power on and off for each virtual machine, and this might slow down the cmdlet performance.

-Partition <String> Specifies the partitions you want to resize. On Windows, you can specify which partition you want to resize by using the Partition parameter. If you do not specify a partition, the last partition of the disk is resized. On Linux, only the last partition can be expanded.

Resizing guest partitions is supported only for Windows OS and for ext3 partitions on RHEL 5. It is achieved by scripts, provided with the vSphere PowerCLI installation. You can modify these scripts or add new ones to support operating systems different than Windows and RHEL 5, and more specific disk resizing scenarios. The scripts are located in the "Scripts" folder in the PowerCLI installation directory and their names have the following format:

GuestDiskExpansion_<OS_Identifier>

<OS_Identifier> is the guest family or the guest ID (as returned by Get-VMGuest).

If no partition is specified, the last partition of the hard disk is resized.

-ResizeGuestPartition Note: This functionality is deprecated and is not functional on the currently supported guest operating systems. Resizing guest disks works only on Windows XP Service Pack 3 and Red Hat Enterprise Linux 5.

Indicates that you want to resize the guest partition of the disk. To use this feature, VMware Tools must be running on the virtual machine. On Windows, you can specify which partition you want to resize by using the Partition parameter. If you don't specify a partition, the last partition of the disk is resized. On Linux, only the last partition can be expanded.

Resizing guest partitions is supported only for Windows OS and for ext3 partitions on RHEL 5. It is achieved by scripts, provided with the vSphere PowerCLI installation. You can modify these scripts or add new ones to support operating systems different than Windows and RHEL 5, and more specific disk resizing scenarios. The scripts are located in the "Scripts" folder in the PowerCLI installation directory and their names have the following format:

GuestDiskExpansion_<OS_Identifier>

<OS_Identifier> is the guest family or the guest ID (as returned by Get-VMGuest).

-Inflate Indicates that you want to inflate the hard disk.

-ZeroOut Specifies that you want to fill the hard disk with zeros. This parameter is supported only if you are directly connected to an ESX/ESXi host. The ZeroOut functionality is experimental.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-HardDisk -VM $vm | Set-HardDisk -Persistence "IndependentNonPersistent"

Changes the persistence of a hard disk to IndependentNonPersistent.

————— Example 2 —————

C:PS>Set-HardDisk -HardDisk $harddisk -CapacityGB $extendedCapacity -GuestCredential $guestCred

Extends a hard disk with the specified capacity. The command also extends the disk on the guest OS.

————— Example 3 —————

C:PS>Set-HardDisk -HardDisk $harddisk -Datastore $datastore

Moves the hard disk to the specified datastore.

REMARKS To see the examples, type: "get-help Set-HardDisk -examples". For more information, type: "get-help Set-HardDisk -detailed". For technical information, type: "get-help Set-HardDisk -full". For online help, type: "get-help Set-HardDisk -online"

Set-IScsiHbaTarget

NAME Set-IScsiHbaTarget

SYNOPSIS This cmdlet modifies the configuration of an iSCSI HBA target.

SYNTAX Set-IScsiHbaTarget -Target <IScsiHbaTarget[]> [-ChapType <ChapType>] [-ChapName <String>] [-ChapPassword <String>] [-MutualChapEnabled <Boolean>] [-MutualChapName <String>] [-MutualChapPassword <String>] [-InheritChap <Boolean>] [-InheritMutualChap <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of an iSCSI HBA target. The cmdlet modifies the CHAP and Digest properties of an iSCSI HBA target. You must specify at least one of the CHAP-related (or Mutual CHAP) parameters. Otherwise, an error message is displayed.

PARAMETERS

-Target <IScsiHbaTarget[]> Specifies the iSCSI HBA target you want to configure. To identify the target, you can provide an IScsiTarget object or use an <Address>:<Port> string.

-ChapType <ChapType> Specifies the type of the CHAP authorization. The valid values are Prohibited, Discouraged, Preferred, and Required. If you set ChapType to Discouraged, Preferred, or Required, then you must specify the ChapPassword parameter as well.

-ChapName <String> Specifies the CHAP initiator name if CHAP is enabled.

-ChapPassword <String> Specifies the CHAP password if CHAP is enabled.

-MutualChapEnabled [<Boolean>] Indicates that mutual CHAP is enabled. In this case, you must specify the MutualChapPassword parameter as well.

-MutualChapName <String> Specifies the Mutual CHAP initiator name if CHAP is enabled.

-MutualChapPassword <String> Specifies the Mutual CHAP password if CHAP is enabled.

-InheritChap [<Boolean>] Indicates that the CHAP setting is inherited from the iSCSI HBA device.

-InheritMutualChap [<Boolean>] Indicates that the Mutual CHAP setting is inherited from the iSCSI HBA device.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-IScsiHbaTarget -Address "10.23.84.73" -Type Static | Set-IScsiHbaTarget -ChapType Prohibited

Retrieves the targets of type Static on the specified address and sets their CHAP type to Prohibited.

————— Example 2 —————

C:PS>$target = Get-IScsiHbaTarget -Address "10.23.84.73" -Type Send

Set-IScsiHbaTarget -Target $target -ChapType Required -ChapPassword pass1 -MutualChapEnabled -MutualChapPassword pass2

Modifies the CHAP and Mutual CHAP settings of the targets of type Send on the specified address.

REMARKS To see the examples, type: "get-help Set-IScsiHbaTarget -examples". For more information, type: "get-help Set-IScsiHbaTarget -detailed". For technical information, type: "get-help Set-IScsiHbaTarget -full". For online help, type: "get-help Set-IScsiHbaTarget -online"

Set-NetworkAdapter

NAME Set-NetworkAdapter

SYNOPSIS This cmdlet modifies the configuration of the virtual network adapter.

SYNTAX Set-NetworkAdapter [-NetworkAdapter] <NetworkAdapter[]> [-MacAddress <String>] [-NetworkName <String>] [-StartConnected <Boolean>] [-Connected <Boolean>] [-WakeOnLan <Boolean>] [-Type <VirtualNetworkAdapterType>] [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-NetworkAdapter [-NetworkAdapter] <NetworkAdapter[]> [-MacAddress <String>] [-StartConnected <Boolean>] [-Connected <Boolean>] [-WakeOnLan <Boolean>] [-Type <VirtualNetworkAdapterType>] -PortId <String> -DistributedSwitch <DistributedSwitch> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-NetworkAdapter [-NetworkAdapter] <NetworkAdapter[]> -Portgroup <VirtualPortGroupBase> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of the virtual network adapter. You can change the MAC address and the network name, and to configure the Connected, StartConnected, and WakeOnLan properties of the adapter.

PARAMETERS

-NetworkAdapter <NetworkAdapter[]> Specifies the virtual network adapter you want to configure.

-MacAddress <String> Specifies an optional MAC address for the virtual network adapter.

-NetworkName <String> Specifies the name of the network to which you want to connect the virtual network adapter. Specifying a distributed port group name is obsolete. Use the Portgroup parameter instead.

-StartConnected [<Boolean>] If the value is $true, the virtual network adapter starts connected when its associated virtual machine powers on. If the value is $false, it starts disconnected.

-Connected [<Boolean>] If the value is $true, the virtual network adapter is connected after its creation. If the value is $false, it is disconnected.

-WakeOnLan [<Boolean>] Indicates that wake-on-LAN is enabled on the virtual network adapter.

-Type <VirtualNetworkAdapterType> Specifies the type of the network adapter. The valid types are e1000, Flexible, Vmxnet, EnhancedVmxnet, and Vmxnet3, and Unknown.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-PortId <String> Specifies the port of the virtual switch to which you want to connect the network adapter. Use this parameter only if the DistributedSwitch parameter is specified.

-DistributedSwitch <DistributedSwitch> Specifies a virtual switch to which you want to connect the network adapter.

-Portgroup <VirtualPortGroupBase> Specifies a standard or a distributed port group to which you want to connect the network adapter.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VM VM | Get-NetworkAdapter | Set-NetworkAdapter -MacAddress '00:50:56:a1:00:00' -WakeOnLan:$true

Configures the Mac address and the WakeOnLan setting of a virtual network adapter.

———— Example 2 ————

C:PS>Get-VM VM | Get-NetworkAdapter | Set-NetworkAdapter -Type EnhancedVmxnet

Sets the type of the virtual network adapter.

———— Example 3 ————

C:PS>Get-VM VM | Get-NetworkAdapter | Set-NetworkAdapter -Connected:$true

Sets the connection state of the virtual network adapter.

———— Example 4 ————

C:PS>$myNetworkAdapters = Get-VM | Get-NetworkAdapter -Name "Network adapter 1" $myVDPortGroup = Get-VDPortgroup -Name MyVDPortGroup Set-NetworkAdapter -NetworkAdapter $myNetworkAdapters -Portgroup $myVDPortGroup

Retrieves all network adapters named "Network adapter 1" from all virtual machines and connects them to the specified distributed port group.

———— Example 5 ————

C:PS>$myNetworkAdapter = Get-VM -Name MyVM | Get-NetworkAdapter -Name "Network adapter 1" $myVDSwitch = Get-VDSwitch -Name MyVDSwitch Set-NetworkAdapter -NetworkAdapter $myNetworkAdapter -DistributedSwitch $MyVDSwitch -PortId 100

Retrieves the network adapter named "Network adapter 1" added to the specified virtual machine and connects it to the specified port on the specified distributed switch.

REMARKS To see the examples, type: "get-help Set-NetworkAdapter -examples". For more information, type: "get-help Set-NetworkAdapter -detailed". For technical information, type: "get-help Set-NetworkAdapter -full". For online help, type: "get-help Set-NetworkAdapter -online"

Set-NicTeamingPolicy

NAME Set-NicTeamingPolicy

SYNOPSIS This cmdlet modifies the specified NIC teaming policy.

SYNTAX Set-NicTeamingPolicy [-VirtualSwitchPolicy] <NicTeamingVirtualSwitchPolicy[]> [-BeaconInterval <Int32>] [-LoadBalancingPolicy <LoadBalancingPolicy>] [-NetworkFailoverDetectionPolicy <NetworkFailoverDetectionPolicy>] [-NotifySwitches <Boolean>] [-FailbackEnabled <Boolean>] [-MakeNicActive <PhysicalNic[]>] [-MakeNicStandby <PhysicalNic[]>] [-MakeNicUnused <PhysicalNic[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-NicTeamingPolicy [-VirtualPortGroupPolicy] <NicTeamingVirtualPortGroupPolicy[]> [-InheritLoadBalancingPolicy <Boolean>] [-InheritNetworkFailoverDetectionPolicy <Boolean>] [-InheritNotifySwitches <Boolean>] [-InheritFailback <Boolean>] [-InheritFailoverOrder <Boolean>] [-LoadBalancingPolicy <LoadBalancingPolicy>] [-NetworkFailoverDetectionPolicy <NetworkFailoverDetectionPolicy>] [-NotifySwitches <Boolean>] [-FailbackEnabled <Boolean>] [-MakeNicActive <PhysicalNic[]>] [-MakeNicStandby <PhysicalNic[]>] [-MakeNicUnused <PhysicalNic[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified NIC teaming policy. You can change the load balancing and failover settings. Default NIC teaming policies are set for the entire virtual switch and can be overridden at port group level.

PARAMETERS

-VirtualSwitchPolicy <NicTeamingVirtualSwitchPolicy[]> Specifies the virtual switch policy to configure.

-BeaconInterval <Int32> Specifies the interval at which the server sends out beacon probes on all NICs in the team. The value must be a positive integer. This parameter is used when the value of the NetworkFailoverDetectionPolicy parameter is BeaconProbing.

-LoadBalancingPolicy <LoadBalancingPolicy> Determines how network traffic is distributed between the network adapters assigned to a switch. The following values are valid:

LoadBalanceIP - Route based on IP hash. Choose an uplink based on a hash of the source and destination IP addresses of each packet. For non-IP packets, whatever is at those offsets is used to compute the hash.

LoadBalanceSrcMac - Route based on source MAC hash. Choose an uplink based on a hash of the source Ethernet.

LoadBalanceSrcId - Route based on the originating port ID. Choose an uplink based on the virtual port where the traffic entered the virtual switch.

ExplicitFailover - Always use the highest order uplink from the list of Active adapters that passes failover detection criteria.

-NetworkFailoverDetectionPolicy <NetworkFailoverDetectionPolicy> Specifies how to reroute traffic in the event of an adapter failure. The following values are valid:

LinkStatus - Relies solely on the link status that the network adapter provides. This option detects failures, such as cable pulls and physical switch power failures, but not configuration errors, such as a physical switch port being blocked by spanning tree or misconfigured to the wrong VLAN or cable pulls on the other side of a physical switch.

BeaconProbing - Sends out and listens for beacon probes on all NICs in the team and uses this information, in addition to link status, to determine link failure. This option detects many of the failures mentioned above that are not detected by link status alone.

-NotifySwitches [<Boolean>] Indicates that whenever a virtual NIC is connected to the virtual switch or whenever that virtual NIC's traffic is routed over a different physical NIC in the team because of a failover event, a notification is sent over the network to update the lookup tables on the physical switches.

-FailbackEnabled [<Boolean>] Specifies how a physical adapter is returned to active duty after recovering from a failure. If the value is $true, the adapter is returned to active duty immediately on recovery, displacing the standby adapter that took over its slot, if any. If the value is $false, a failed adapter is left inactive even after recovery until another active adapter fails, requiring its replacement.

-MakeNicActive <PhysicalNic[]> Specifies the adapters you want to continue to use when the network adapter connectivity is available and active.

-MakeNicStandby <PhysicalNic[]> Specifies the adapters you want to use if one of the active adapter's connectivity is unavailable.

-MakeNicUnused <PhysicalNic[]> Specifies the adapters you do not want to use.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-VirtualPortGroupPolicy <NicTeamingVirtualPortGroupPolicy[]> Specifies the virtual port group policy to configure.

-InheritLoadBalancingPolicy [<Boolean>] Indicates that the value of the LoadBalancingPolicy parameter is inherited from the virtual switch.

-InheritNetworkFailoverDetectionPolicy [<Boolean>] Indicates that the value of the NetworkFailoverDetectionPolicy parameter is inherited from the virtual switch.

-InheritNotifySwitches [<Boolean>] Indicates that the value of the NotifySwitches parameter is inherited from the virtual switch.

-InheritFailback [<Boolean>] Indicates that the value of the FailbackEnabled parameter is inherited from the virtual switch.

-InheritFailoverOrder [<Boolean>] Indicates that the value of the MakeNicActive, MakeNicStandBy, and MakeNicUnused parameters are inherited from the virtual switch.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$policy = Get-VirtualSwitch -VMHost (Get-VMHost *.128) -Name vSwitch1 | Get-NicTeamingPolicy

$policy | Set-NicTeamingPolicy -LoadBalancingPolicy LoadBalanceSrcMac

Configures the NicTeaming policy of the vSwitch1 virtual switch.

REMARKS To see the examples, type: "get-help Set-NicTeamingPolicy -examples". For more information, type: "get-help Set-NicTeamingPolicy -detailed". For technical information, type: "get-help Set-NicTeamingPolicy -full". For online help, type: "get-help Set-NicTeamingPolicy -online"

Set-OSCustomizationNicMapping

NAME Set-OSCustomizationNicMapping

SYNOPSIS This cmdlet modifies the provided OS customization NIC mappings.

SYNTAX Set-OSCustomizationNicMapping -OSCustomizationNicMapping <OSCustomizationNicMapping[]> [-Position <Int32>] [-Server <VIServer[]>] [-IpMode <OSCustomizationIPMode>] [-VCApplicationArgument <String>] [[-IpAddress] <String>] [[-SubnetMask] <String>] [[-DefaultGateway] <String>] [-AlternateGateway <String>] [[-Dns] <String[]>] [-Wins <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-OSCustomizationNicMapping -OSCustomizationNicMapping <OSCustomizationNicMapping[]> [-NetworkAdapterMac <String>] [-Server <VIServer[]>] [-IpMode <OSCustomizationIPMode>] [-VCApplicationArgument <String>] [[-IpAddress] <String>] [[-SubnetMask] <String>] [[-DefaultGateway] <String>] [-AlternateGateway <String>] [[-Dns] <String[]>] [-Wins <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the provided OS customization NIC mappings. If the parent spec of the provided NIC mapping is a server-side spec, it is updated on the server. If the parent spec is client-side, the reference that is kept in the memory is updated, but the variable that is passed to the cmdlet is not modified.

PARAMETERS

-OSCustomizationNicMapping <OSCustomizationNicMapping[]> Specifies the OS customization NIC mapping you want to configure.

-Position <Int32> Specifies the position of the mapping you want to modify.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-IpMode <OSCustomizationIPMode> Specifies the IP configuration mode. The valid values are UseDhcp, PromptUser, UseVCApplication, and UseStaticIP.

-VCApplicationArgument <String> Specifies a new argument you want to pass to VCApplication in order to obtain an IP address.

-IpAddress <String> Specifies an IP address. Using this parameter automatically sets the IpMode parameter to UseStaticIp.

-SubnetMask <String> Specifies a subnet mask.

-DefaultGateway <String> Specifies a default gateway.

-AlternateGateway <String> Specifies an alternate gateway.

-Dns <String[]> Specifies a DNS address. This parameter applies only to Windows operating systems.

-Wins <String[]> Specifies WINS servers. This parameter applies only to Windows operating systems.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-NetworkAdapterMac <String> Specifies the MAC address of the network adapter to which you want to map the OS customization specification.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-OSCustomizationSpec Spec | Get-OSCustomizationNicMapping | Set-OSCustomizationNicMapping -IpAddress 10.0.0.2

Modifies the IP address of the specified NIC mapping that uses static IP mode.

———— Example 2 ————

C:PS>Get-OSCustomizationSpec Spec | Get-OSCustomizationNicMapping | Set-OSCustomizationNicMapping -VcApplicationArgument "subnet2"

Modifies the VCApplication argument of the specified NIC mapping.

———— Example 3 ————

C:PS>Get-OSCustomizationSpec Spec | Get-OSCustomizationNicMapping | Set-OSCustomizationNicMapping -IpMode UseStaticIp -IpAddress 10.10.0.1 -SubnetMask 255.255.255.0 -DefaultGateway 10.10.0.1 -AlternateGateway 10.10.0.1 -Dns 10.10.150.1 -PrimaryWins 10.10.150.2

Modifies the attributes of a NIC mapping.

———— Example 4 ————

C:PS>Set-OSCustomizationNicMapping -OSCustomizationNicMapping $nicMapping1, $nicMapping2 -IPMode UseVCApplication -VcApplicationArgument "subnet2"

Modifies the specified NIC mapping using VCApplication.

REMARKS To see the examples, type: "get-help Set-OSCustomizationNicMapping -examples". For more information, type: "get-help Set-OSCustomizationNicMapping -detailed". For technical information, type: "get-help Set-OSCustomizationNicMapping -full". For online help, type: "get-help Set-OSCustomizationNicMapping -online"

Set-OSCustomizationSpec

NAME Set-OSCustomizationSpec

SYNOPSIS This cmdlet modifies the specified OS customization specification.

SYNTAX Set-OSCustomizationSpec [-OSCustomizationSpec] <OSCustomizationSpec[]> [-NewSpec <OSCustomizationSpec>] [-Type <OSCustomizationSpecType>] [-Server <VIServer[]>] [-Name <String>] [-DnsServer <String[]>] [-DnsSuffix <String[]>] [-Domain <String>] [-NamingScheme <String>] [-NamingPrefix <String>] [-Description <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-OSCustomizationSpec [-FullName <String>] [-OrgName <String>] [-ChangeSID <Boolean>] [-DeleteAccounts <Boolean>] [-OSCustomizationSpec] <OSCustomizationSpec[]> [-NewSpec <OSCustomizationSpec>] [-Type <OSCustomizationSpecType>] [-Server <VIServer[]>] [-Name <String>] [-DnsServer <String[]>] [-DnsSuffix <String[]>] [-GuiRunOnce <String[]>] [-AdminPassword <String>] [-TimeZone <String>] [-AutoLogonCount <Int32>] [-Domain <String>] [-Workgroup <String>] [-DomainCredentials

<PSCredential>] [-DomainUsername <String>] [-DomainPassword <String>] [-ProductKey <String>] [-NamingScheme <String>] [-NamingPrefix <String>] [-Description <String>] [-LicenseMode <LicenseMode>] [-LicenseMaxConnections <Int32>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified OS customization specification. The specification to be updated is identified by one or both of the Name and Spec parameters. If a Windows specification is to be updated, one of the Domain and Workgroup parameters must be provided. If a Linux specification is to be updated, the Domain parameter must be provided.

PARAMETERS

-OSCustomizationSpec <OSCustomizationSpec[]> Specifies the specification you want to modify.

-NewSpec <OSCustomizationSpec> If no other parameters are provided, this parameter specifies a specification from which to retrieve information for the updated specification.

-Type <OSCustomizationSpecType> Sets the type of the OS customization specification. The valid values are Persistent and NonPersistent.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Name <String> Specifies a new name for the OS customization specification.

-DnsServer <String[]> Specifies the DNS server. This parameter applies only to Linux operating systems.

-DnsSuffix <String[]> Specifies the DNS suffix. This parameter applies only to Linux operating systems.

-Domain <String> Specifies the domain name.

-NamingScheme <String> Specifies the naming scheme for the virtual machine. The valid values are Custom, Fixed, Prefix, and Vm.

-NamingPrefix <String> The behavior of this parameter is related to the customization scheme. If a Custom customization scheme is specified, NamingPrefix is an optional argument that is passed to the utility for this IP address. The value of this field is defined by the user in the script. If a Fixed customization scheme is specified, NamingPrefix should indicate the name of the virtual machine. If a Prefix customization scheme is set, NamingPrefix indicates the prefix to which a unique number is appended.

-Description <String> Provides a new description for the specification.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-FullName <String> Specifies the administrator's full name. This parameter applies only to Windows operating systems.

-OrgName <String> Specifies the name of the organization to which the administrator belongs.

-ChangeSID [<Boolean>] Indicates that the customization should modify the system security identifier (SID). This parameter applies only to Windows operating systems.

-DeleteAccounts [<Boolean>] Indicates that you want to delete all user accounts. This parameter applies only to Windows operating systems.

-GuiRunOnce <String[]> Provides a list of commands to run after first user login. This parameter applies only to Windows operating systems.

-AdminPassword <String> Specifies the new OS administrator's password. This parameter applies only to Windows operating systems.

-TimeZone <String> Specifies the name or ID of the time zone for a Windows guest OS only. Using wildcards is supported. The following time zones are available:

000 Int'l Dateline 001 Samoa 002 Hawaii 003 Alaskan 004 Pacific 010 Mountain (U.S. and Canada) 015 U.S. Mountain: Arizona 020 Central (U.S. and Canada) 025 Canada Central 030 Mexico 033 Central America 035 Eastern (U.S. and Canada) 040 U.S. Eastern: Indiana (East) 045 S.A. Pacific 050 Atlantic (Canada) 055 S.A. Western 056 Pacific S.A. 060 Newfoundland 065 E. South America 070 S.A. Eastern 073 Greenland 075 Mid-Atlantic 080 Azores 083 Cape Verde Islands 085 GMT (Greenwich Mean Time) 090 GMT Greenwich 095 Central Europe 100 Central European 105 Romance 110 W. Europe 113 W. Central Africa 115 E. Europe 120 Egypt 125 EET (Helsinki, Riga, Tallinn) 130 EET (Athens, Istanbul, Minsk) 135 Israel: Jerusalem 140 S. Africa: Harare, Pretoria 145 Russian 150 Arab 155 E. Africa 160 Iran 165 Arabian 170 Caucasus Pacific (U.S. and Canada) 175 Afghanistan 180 Russia Yekaterinburg 185 W. Asia 190 India 193 Nepal 195 Central Asia 200 Sri Lanka 201 N. Central Asia 203 Myanmar: Rangoon 205 S.E. Asia 207 N. Asia 210 China 215 Singapore 220 Taipei 225 W. Australia 227 N. Asia East 230 Korea: Seoul 235 Tokyo 240 Sakha Yakutsk 245 A.U.S. Central: Darwin 250 Central Australia 255 A.U.S. Eastern 260 E. Australia 265 Tasmania 270 Vladivostok 275 W. Pacific 280 Central Pacific 285 Fiji 290 New Zealand 300 Tonga

-AutoLogonCount <Int32> Specifies the number of times the virtual machine should automatically login as an administrator. The valid values are in the range between 0 and Int32.MaxValue. Specifying 0 disables auto log-on. This parameter applies only to Windows operating systems.

-Workgroup <String> Specifies the workgroup. This parameter applies only to Windows operating systems.

-DomainCredentials <PSCredential> Specifies credentials for authentication with the specified domain. This parameter applies only to Windows operating systems.

-DomainUsername <String> Specifies a username for authentication with the specified domain. This parameter applies only to Windows operating systems.

-DomainPassword <String> Specifies a password for authentication with the specified domain. This parameter applies only to Windows operating systems.

-ProductKey <String> Specifies the MS product key. If the guest OS version is earlier than Vista, this parameter is required in order to make the customization unattended. For Windows Vista and later, the OS customization is unattended no matter if the ProductKey parameter is set.

-LicenseMode <LicenseMode> Specifies the license mode of the Windows 2000/2003 guest operating system. The valid values are Perseat, Perserver, and NotSpecified. If Perserver is set, use the LicenseMaxConnection parameter to define the maximum number of connections. This parameter applies only to Windows operating systems.

-LicenseMaxConnections <Int32> Specifies the maximum connections for server license mode. Use this parameter only if the LicenseMode parameter is set to Perserver. This parameter applies only to Windows operating systems.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-OSCustomizationSpec Spec -Description 'This is a test OS customization specification.'

Updates the description of the Spec OS customization specification.

REMARKS To see the examples, type: “get-help Set-OSCustomizationSpec -examples”. For more information, type: “get-help Set-OSCustomizationSpec -detailed”. For technical information, type: “get-help Set-OSCustomizationSpec -full”. For online help, type: “get-help Set-OSCustomizationSpec -online”

Set-PowerCLIConfiguration

NAME Set-PowerCLIConfiguration

SYNOPSIS This cmdlet modifies the vSphere PowerCLI configuration.

SYNTAX Set-PowerCLIConfiguration [-ProxyPolicy <ProxyPolicy>] [-DefaultVIServerMode <DefaultVIServerMode>] [-InvalidCertificateAction <BadCertificateAction>] [-ParticipateInCeip <Boolean>] [-CEIPDataTransferProxyPolicy <ProxyPolicy>] [-DisplayDeprecationWarnings <Boolean>] [-WebOperationTimeoutSeconds <Int32>] [-VMConsoleWindowBrowser <String>] [-Scope <ConfigurationScope>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the vSphere PowerCLI configuration.

PARAMETERS

-ProxyPolicy <ProxyPolicy> Specifies whether vSphere PowerCLI uses a system proxy server to connect to the vCenter Server system. The valid values are NoProxy and UseSystemProxy.

-DefaultVIServerMode <DefaultVIServerMode> Specifies the server connection mode. The new configuration takes effect immediately after you run the cmdlet. The following values are valid:

- Single - Switching to “single” removes all server connections except the last established one. If no target servers are specified, cmdlets run only on the last connected server.
- Multiple - All servers connected after switching to “multiple” mode are stored together with the current server connection in an array variable. If no target servers are

specified, cmdlets run on the servers in the variable.

For more information on default servers, see the description of Connect-VIServer.

-InvalidCertificateAction <BadCertificateAction> Define the action to take when an attempted connection to a server fails due to a certificate error. The following values are valid:

Unset - this is the default value and it acts as a “Warn” value for Connect-VIServer and as “Prompt” for “Connect-CloudServer”.

Prompt - if the server certificate is not trusted the cmdlet will prompt you for a course of action before it continues. T
Deny - no connection will be established. Accept for once - accept the connection only for the current PowerCLI session. You will be prompted again if you attempt to connect to the same server from other

processes. Accept Permanently - the action will add this certificate as an exception in the “SSL Certificate Exceptions” user list. The Connect-VIServer and Connect-CIServer

cmdlets will never prompt again what action to take for this particulate certificate and server.
Accept For All Users - same as above, however the exception will be added to all user lists, which is common for all Windows accounts on the current machine.

Fail - the cmdlet will not establish connection if the certificate is not valid.

Ignore - the cmdlet will establish the connection without taking into account that the certificate is invalid.

Warn - the cmdlet will display a warning saying that the certificate is not valid, the reason why it is not considered valid and then will print additional information about the certificate.

For more information about invalid certificates, run ‘Get-Help about_invalid_certificates’.

-ParticipateInCeip [<Boolean>] Specifies if PowerCLI should send anonymous usage information to VMware. For more information about the Customer Experience Improvement Program (CEIP), see the PowerCLI User's Guide. Setting this option is valid only for the AllUsers and User configuration scopes. Changing this setting requires a restart of PowerCLI before it takes effect.

-CEIPDataTransferProxyPolicy <ProxyPolicy> Specifies the proxy policy for the connection through which Customer Experience Improvement Program (CEIP) data is sent to VMware. Setting this option is valid only when ParticipateInCEIP option is set to $true. Changing this setting requires a restart of PowerCLI before it takes effect.

-DisplayDeprecationWarnings [<Boolean>] Indicates whether you want to see warnings about deprecated elements.

-WebOperationTimeoutSeconds <Int32> Defines the timeout for Web operations. The default value is 300 sec. To specify an infinite operation timeout, pass a negative integer to this parameter. Changing this setting requires a restart of PowerCLI before it takes effect.

-VMConsoleWindowBrowser <String> Specifies the Web browser to be used for opening virtual machine console windows (by using the Open-VMConsoleWindow cmdlet). The browser must be 32-bit.

-Scope <ConfigurationScope>

Specifies the scope of the setting that you want to modify. The parameter accepts Sesstion, User and All Users values
*Session - the setting is valid for the current vSphere PowerCLI session only and overrides any User and All Users settings. *User - the setting is valid for the current Windows user only, overrides All Users settings, and is applied only if a Session setting cannot be detected. *All Users - the setting is valid for all users and is applied only if Session and User settings cannot be detected.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-PowerCLIConfiguration -ProxyPolicy NoProxy -Scope Session

Modifies the proxy policy of vSphere PowerCLI for the Session scope.

———— Example 2 ————

C:PS>Set-PowerCLIConfiguration -ProxyPolicy NoProxy -DefaultVIServerMode Single

Changes the default server connection mode and the proxy policy of vSphere PowerCLI for the AllUsers scope.

———— Example 3 ————

C:PS>Set-PowerCLIConfiguration -DefaultVIServerMode 'Single' -Scope ([VMware.VimAutomation.ViCore.Types.V1.ConfigurationScope]::User -bor [VMware.VimAutomation.ViCore.Types.V1.ConfigurationScope]::AllUsers)

Changes the default server connection mode of vSphere PowerCLI for the User and AllUsers scopes.

REMARKS To see the examples, type: "get-help Set-PowerCLIConfiguration -examples". For more information, type: "get-help Set-PowerCLIConfiguration -detailed". For technical information, type: "get-help Set-PowerCLIConfiguration -full". For online help, type: "get-help Set-PowerCLIConfiguration -online"

Set-ResourcePool

NAME Set-ResourcePool

SYNOPSIS This cmdlet modifies the properties of the specified resource pool.

SYNTAX Set-ResourcePool [-ResourcePool] <ResourcePool[]> [-Name <String>] [-CpuExpandableReservation <Boolean>] [-CpuLimitMhz <Int64>] [-CpuReservationMhz <Int64>] [-CpuSharesLevel <SharesLevel>] [-MemExpandableReservation <Boolean>] [-MemLimitMB <Int64>] [-MemLimitGB <Decimal>] [-MemReservationMB <Int64>] [-MemReservationGB <Decimal>] [-MemSharesLevel <SharesLevel>] [-NumCpuShares <Int32>] [-NumMemShares <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified resource pool.

PARAMETERS

-ResourcePool <ResourcePool[]> Specifies the resource pool you want to configure.

-Name <String> Specifies a new name for the resource pool.

-CpuExpandableReservation [<Boolean>] Indicates that the CPU reservation can grow beyond the specified value if the parent resource pool has unreserved resources.

-CpuLimitMhz <Int64> Specifies a CPU usage limit in MHz. If this parameter is set, utilization will not exceed this limit even if there are available resources.

-CpuReservationMhz <Int64> Specifies the guaranteed available CPU in MHz.

-CpuSharesLevel <SharesLevel> Specifies the CPU allocation level for this pool. This property is used in relative allocation between resource consumers. This parameter accepts Custom, High, Low, and Normal values.

-MemExpandableReservation [<Boolean>] Indicates that the memory reservation can grow beyond the specified value if the parent resource pool has unreserved resources.

-MemLimitMB <Int64> This parameter is obsolete. Use MemLimitGB instead. Specifies a memory usage limit in megabytes (MB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemLimitGB <Decimal> Specifies a memory usage limit in gigabytes (GB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemReservationMB <Int64> This parameter is obsolete. Use MemReservationGB instead. Specifies the guaranteed available memory in megabytes (MB).

-MemReservationGB <Decimal> Specifies the guaranteed available memory in gigabytes (GB).

-MemSharesLevel <SharesLevel> Specifies the memory allocation level for the resource pool. This property is used in relative allocation between resource consumers. This parameter accepts Custom, High, Low, and Normal values.

-NumCpuShares <Int32> Specifies the CPU allocation level for the resource pool. This property is used in relative allocation between resource consumers. This parameter is ignored unless CpuSharesLevel is set to Custom.

-NumMemShares <Int32> Specifies the memory allocation level for the resource pool. This property is used in relative allocation between resource consumers. This parameter is ignored unless MemSharesLevel is set to Custom.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-ResourcePool -Resourcepool Resourcepool -NumCpuShares 512 -MemLimitGB 4

Sets the CPU allocation level and the limit on memory usage in GB for the resource pool named Resourcepool.

REMARKS To see the examples, type: "get-help Set-ResourcePool -examples". For more information, type: "get-help Set-ResourcePool -detailed". For technical information, type: "get-help Set-ResourcePool -full". For online help, type: "get-help Set-ResourcePool -online"

Set-ScsiController

NAME Set-ScsiController

SYNOPSIS This cmdlet modifies the specified SCSI controllers.

SYNTAX Set-ScsiController [-ScsiController] <ScsiController[]> [-BusSharingMode <ScsiBusSharingMode>] [-Type <ScsiControllerType>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified SCSI controllers. Set-ScsiController cannot set both the Type and BusSharing parameters at the same time. First run the cmdlet to set the type and then run it again to configure the bus sharing mode.

PARAMETERS

-ScsiController <ScsiController[]> Specifies the SCSI controller you want to modify.

-BusSharingMode <ScsiBusSharingMode> Specifies the bus sharing mode of the SCSI controller. The valid values are NoSharing, Physical, and Virtual.

-Type <ScsiControllerType> Specifies the type of the SCSI controller. The valid values are ParaVirtual, VirtualBusLogic, VirtualLsiLogic, and VirtualLsiLogicSAS.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-ScsiController -VM VM | Set-ScsiController -BusSharingMode Physical

Configures the bus sharing mode of the SCSI controllers of a virtual machine to Physical mode.

———— Example 2 ————

C:PS>$scsiController = Get-HardDisk -VM VM | Select -First 1 | Get-ScsiController

Set-ScsiController -ScsiController $scsiController -Type VirtualLsiLogic

Changes the type of the SCSI controller of the first hard disk of the VM virtual machine to VirtualLsiLogic.

REMARKS To see the examples, type: "get-help Set-ScsiController -examples". For more information, type: "get-help Set-ScsiController -detailed". For technical information, type: "get-help Set-ScsiController -full". For online help, type: "get-help Set-ScsiController -online"

Set-ScsiLun

NAME Set-ScsiLun

SYNOPSIS This cmdlet modifies the configuration of a SCSI device.

SYNTAX Set-ScsiLun [[-MultipathPolicy] <ScsiLunMultipathPolicy>] [[-PreferredPath] <ScsiLunPath>] [-ScsiLun] <ScsiLun[]> [-CommandsToSwitchPath <Int32>] [-BlocksToSwitchPath <Int32>] [-NoCommandsSwitch] [-NoBlocksSwitch] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of a SCSI device.

PARAMETERS

-MultipathPolicy <ScsiLunMultipathPolicy> Specifies the policy that the logical unit must use when choosing a path. The following values are valid:

Fixed - uses the preferred path whenever possible. RoundRobin - load balance. MostRecentlyUsed - uses the most recently used path. Unknown - supported only when connected to vCenter Server 4.1/ESX 4.1.

Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-PreferredPath <ScsiLunPath> Specifies the preferred path to access the SCSI logical unit. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release.

-ScsiLun <ScsiLun[]> Specifies the SCSI device you want to configure.

-CommandsToSwitchPath <Int32> Specifies the maximum number of I/O requests to be issued on a given path before the system tries to select a different path. Modifying this setting affects all ScsiLun devices that are connected to the same ESX host. The default value is 50. Setting this parameter to zero (0) disables switching based on commands. This parameter is not supported on vCenter Server 4.x.

-BlocksToSwitchPath <Int32> Specifies the maximum number of I/O blocks to be issued on a given path before the system tries to select a different path. Modifying this setting affects all ScsiLun devices that are connected to the same ESX/ESXi host. The default value is 2048. Setting this parameter to zero (0) disables switching based on blocks.

-NoCommandsSwitch	Indicates that switching based on commands is disabled. Not supported on vCenter Server 4.x.
-NoBlocksSwitch	Indicates that switching based on blocks is disabled. Not supported on vCenter Server 4.x.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$scsilun = Get-ScsiLun -VMHost 10.23.123.100 -LunType disk

Set-ScsiLun -ScsiLun $scsilun -CommandsToSwitchPath 100

Configures the SCSI Lun device of the virtual machine host, so that the maximum number of I/O requests to be issued before the system tries to select a different path is 100.

REMARKS To see the examples, type: “get-help Set-ScsiLun -examples”. For more information, type: “get-help Set-ScsiLun -detailed”. For technical information, type: “get-help Set-ScsiLun -full”. For online help, type: “get-help Set-ScsiLun -online”

Set-ScsiLunPath

NAME Set-ScsiLunPath

SYNOPSIS This cmdlet configures a vmhba path to a SCSI device.

SYNTAX Set-ScsiLunPath [[-Active] <Boolean>] [-ScsiLunPath] <ScsiLunPath[]> [-Preferred] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures a vmhba path to a SCSI device.

PARAMETERS

-Active [<Boolean>] Indicates that the specified path is active.

-ScsiLunPath <ScsiLunPath[]> Specifies a path to the SCSI logical unit you want to configure.

-Preferred	Indicates that the specified path is preferred. Only one path can be preferred, so when a path is made preferred, the preference is removed from the previously preferred path.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$scsilun = Get-ScsiLun -VMHost 10.23.123.100 -LunType disk

$scsipath = Get-ScsiLunPath -ScsiLun $scsilun

Set-ScsiLunPath -ScsiLunPath $scsipath -Preferred $true

Sets the specified SCSI Lun path as preferred.

REMARKS To see the examples, type: “get-help Set-ScsiLunPath -examples”. For more information, type: “get-help Set-ScsiLunPath -detailed”. For technical information, type: “get-help Set-ScsiLunPath -full”. For online help, type: “get-help Set-ScsiLunPath -online”

Set-SecurityPolicy

NAME Set-SecurityPolicy

SYNOPSIS This cmdlet modifies the security policy for virtual port groups or the default port security policy for virtual switches.

SYNTAX Set-SecurityPolicy [-VirtualSwitchPolicy] <VirtualSwitchSecurityPolicy[]> [-AllowPromiscuous <Boolean>] [-ForgedTransmits <Boolean>] [-MacChanges <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-SecurityPolicy [-VirtualPortGroupPolicy] <VirtualPortgroupSecurityPolicy[]> [-AllowPromiscuousInherited <Boolean>] [-ForgedTransmitsInherited <Boolean>] [-MacChangesInherited <Boolean>] [-AllowPromiscuous <Boolean>] [-ForgedTransmits <Boolean>] [-MacChanges <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the security policy for virtual port groups or the default port security policy for virtual switches. Specifying a parameter automatically changes the parameter's Inherited setting to 'false'. Specifying the parameter's Inherited setting as 'true' automatically applies the switch level security policy to the parameter.

PARAMETERS

-VirtualSwitchPolicy <VirtualSwitchSecurityPolicy[]> Specifies the virtual switch security policy that you want to configure.

-AllowPromiscuous [<Boolean>] Specifies whether promiscuous mode is enabled for the corresponding virtual port group or switch.

-ForgedTransmits [<Boolean>] Specifies whether forged transmits are enabled for the corresponding virtual port group or switch.

-MacChanges [<Boolean>] Specifies whether MAC address changes are enabled for the corresponding virtual port group or switch.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-VirtualPortGroupPolicy <VirtualPortgroupSecurityPolicy[]> Specifies the virtual port group security policy that you want to configure.

-AllowPromiscuousInherited [<Boolean>] Specifies whether the AllowPromiscuous setting is inherited from the parent virtual switch.

-ForgedTransmitsInherited [<Boolean>] Specifies whether the ForgedTransmits setting is inherited from the parent virtual switch.

-MacChangesInherited [<Boolean>] Specifies whether the MacChanges setting is inherited from the parent virtual switch.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VirtualSwitch -Name "MyVirtualSwitch" | Get-SecurityPolicy | Set-SecurityPolicy -MacChanges $false

Retrieves a virtual switch named “MyVirtualSwitch” and updates its security policy to forbid MAC address changes.

———— Example 2 ————

C:PS>Get-VirtualPortgroup -Name “MyVirtualPortGroup” | Get-SecurityPolicy | Set-SecurityPolicy -ForgedTransmitsInherited $true

Retrieves a virtual port group named “MyVirtualPortGroup” and updates the security policy to inherit the setting value for controlling outbound frames filtering by MAC address from its parent.

REMARKS To see the examples, type: “get-help Set-SecurityPolicy -examples”. For more information, type: “get-help Set-SecurityPolicy -detailed”. For technical information, type: “get-help Set-SecurityPolicy -full”. For online help, type: “get-help Set-SecurityPolicy -online”

Set-Snapshot

NAME Set-Snapshot

SYNOPSIS This cmdlet modifies the specified virtual machine snapshot.

SYNTAX Set-Snapshot [-Snapshot] <Snapshot[]> [-Name <String>] [-Description <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the name and the description of the specified virtual machine snapshot.

PARAMETERS

-Snapshot <Snapshot[]> Specifies the snapshot whose properties you want to change.

-Name <String> Specifies a new name for the snapshot.

-Description <String> Provides a new description for the snapshot.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-Snapshot -Snapshot $snapshot -Name BeforePatch -Description “Before windows update”

Sets the name and the description of the snapshot in the $snapshot variable.

———— Example 2 ————

C:PS>Get-VM | Get-Snapshot -Name “InitialState” | Set-Snapshot -Description “This snapshot is created right after the OS installation.”

Updates the description of all snapshots with name InitialState, from all virtual machines.

REMARKS To see the examples, type: “get-help Set-Snapshot -examples”. For more information, type: “get-help Set-Snapshot -detailed”. For technical information, type: “get-help Set-Snapshot -full”. For online help, type: “get-help Set-Snapshot -online”

Set-StatInterval

NAME Set-StatInterval

SYNOPSIS This cmdlet changes the statistics interval that is specified by the provided parameters.

SYNTAX Set-StatInterval [[-SamplingPeriodSecs] <Int32>] [[-StorageTimeSecs] <Int32>] [-Interval] <StatInterval[]> [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet changes the statistics interval that is specified by the provided parameters.

PARAMETERS

-SamplingPeriodSecs <Int32> Specifies a new sampling period in seconds.

-StorageTimeSecs <Int32> Specifies a new time period in seconds, for which the statistics information is kept.

-Interval <StatInterval[]> Specifies the statistics interval you want to change.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-StatInterval -Interval “past day” -StorageTimeSecs 700000

Changes the storage time of the “past day” statistics interval.

REMARKS To see the examples, type: “get-help Set-StatInterval -examples”. For more information, type: “get-help Set-StatInterval -detailed”. For technical information, type: “get-help Set-StatInterval -full”. For online help, type: “get-help Set-StatInterval -online”

Set-Tag

NAME Set-Tag

SYNOPSIS This cmdlet modifies the specified tags.

SYNTAX Set-Tag [-Tag] <Tag[]> [-Name <String>] [-Description <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified tags.

PARAMETERS

-Tag <Tag[]> Specifies the tags that you want to configure.

-Name <String> Specifies the name to which the specified tags will be renamed.

-Description <String> Specifies the new description to set to the specified tags.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-Tag -Name “MyTag” | Set-Tag -Name “MyNewTag” -Description “MyNewDescription”

Retrieves a tag named “MyTag” and updates its name and description.

REMARKS To see the examples, type: “get-help Set-Tag -examples”. For more information, type: “get-help Set-Tag -detailed”. For technical information, type: “get-help Set-Tag -full”. For online help, type: “get-help Set-Tag -online”

Set-TagCategory

NAME Set-TagCategory

SYNOPSIS This cmdlet modifies the specified tag categories.

SYNTAX Set-TagCategory [-Category] <TagCategory[]> [-Name <String>] [-Description <String>] [-Cardinality <Cardinality>] [-AddEntityType <String[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified tag categories. The cardinality of a tag category can only be changed from “Single” to “Multiple”.

PARAMETERS

-Category <TagCategory[]> Specifies the tag categories that you want to configure.

-Name <String> Specifies the name to which the specified tag categories will be renamed.

-Description <String> Specifies the new description to set to the tag categories.

-Cardinality <Cardinality> Specifies the cardinality of the tag category. If not specified, the default value is “Single”.

“Single” means that only a single tag from this category can be assigned to a specific object at a time. “Multiple” means that more than one tag from this category can be assigned to a specific object at a time.

The only value that is accepted for this parameter is “Multiple”.

-AddEntityType <String[]> Adds the specified entity types to the list of types that tags in this category are applicable to. If you specify “All” as a value, the tags will be applicable to all entity types.

This parameter accepts both PowerCLI type names and vSphere API type names. The valid PowerCLI type names are: Cluster, Datacenter, Datastore, DatastoreCluster, DistributedPortGroup, DistributedSwitch, Folder, ResourcePool, VApp, VirtualPortGroup, VirtualMachine, VM, VMHost.

For vSphere type names that are not vCenter Server API type names, a namespace prefix is used. The format is: <namespace>/<type> Example: 'vco/WorkflowItem'

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-TagCategory "MyTagCategory" | Set-TagCategory -Name "MyNewCategoryName" -Description "Update MyTagCategory description"

Retrieves a tag category named "MyTagCategory" and updates its name and description.

———— Example 2 ————

C:PS>$myTagCategory = Get-TagCategory "MyTagCategory" Set-TagCategory -Category $myTagCategory -Cardinality Multiple -AddEntityType "VirtualMachine"

Retrieves a tag category named "MyTagCategory" and updates it by allowing more than one of its tags to be assigned to a specific object at a time, as well as adding "VirtualMachine" to the set of applicable entity types.

REMARKS To see the examples, type: "get-help Set-TagCategory -examples". For more information, type: "get-help Set-TagCategory -detailed". For technical information, type: "get-help Set-TagCategory -full". For online help, type: "get-help Set-TagCategory -online"

Set-Template

NAME Set-Template

SYNOPSIS This cmdlet modifies the specified virtual machine template.

SYNTAX Set-Template [-Template] <Template[]> [-Name <String>] [-ToVM] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet changes the name and the description of a virtual machine template according to the provided parameters. The cmdlet can convert the template to a virtual machine if the value of the ToVM parameter is $true.

PARAMETERS

-Template <Template[]> Specifies the template whose properties you want to change.

-Name <String> Specifies a new name for the template.

-ToVM Indicates that the template is to be converted to a virtual machine.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-Template -Template $template -Name Template2

Renames the template saved in the $template variable to Template2.

———— Example 2 ————

C:PS>$vm = Set-Template -Template $template -ToVM

Converts a template to a virtual machine.

REMARKS To see the examples, type: "get-help Set-Template -examples". For more information, type: "get-help Set-Template -detailed". For technical information, type: "get-help Set-Template -full". For online help, type: "get-help Set-Template -online"

Set-VApp

NAME Set-VApp

SYNOPSIS This cmdlet modifies the specified vApp.

SYNTAX Set-VApp -VApp <VApp[]> [-Name <String>] [-CpuExpandableReservation <Boolean>] [-CpuLimitMhz <Int64>] [-CpuReservationMhz <Int64>] [-CpuSharesLevel <SharesLevel>] [-MemExpandableReservation <Boolean>] [-MemLimitMB <Int64>] [-MemLimitGB <Decimal>] [-MemReservationMB <Int64>] [-MemReservationGB <Decimal>] [-MemSharesLevel <SharesLevel>] [-NumCpuShares <Int32>] [-NumMemShares <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified vApp.

PARAMETERS

-VApp <VApp[]> Specifies the vApp that you want to configure.

-Name <String> Modifies the name of the vApp.

-CpuExpandableReservation [<Boolean>] Indicates that the CPU reservation can grow beyond the specified value if there are available resources.

-CpuLimitMhz <Int64> Specifies a CPU usage limit in MHz. If this parameter is set, utilization will not exceed this limit even if there are available resources.

-CpuReservationMhz <Int64> Specifies the guaranteed available CPU in MHz.

-CpuSharesLevel <SharesLevel> Specifies the CPU allocation level for this vApp. This property is used in relative allocation between resource consumers. This parameter accepts Custom, High, Low, and Normal values.

-MemExpandableReservation [<Boolean>] Indicates that the memory reservation can grow beyond the specified value if there are available resources.

-MemLimitMB <Int64> This parameter is obsolete. Use MemLimitGB instead. Specifies a memory usage limit in megabytes (MB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemLimitGB <Decimal> Specifies a memory usage limit in gigabytes (GB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemReservationMB <Int64> This parameter is obsolete. Use MemReservationGB instead. Specifies the guaranteed available memory in megabytes (MB).

-MemReservationGB <Decimal> Specifies the guaranteed available memory in gigabytes (GB).

-MemSharesLevel <SharesLevel> Specifies the memory allocation level for the vApp. This property is used in relative allocation between resource consumers. This cmdlet accepts Custom, High, Low, and Normal values.

-NumCpuShares <Int32> Specifies the CPU allocation level for the vApp. This property is used in relative allocation between resource consumers. This parameter is ignored unless the CpuSharesLevel parameter is set to Custom.

-NumMemShares <Int32> Specifies the memory allocation level for the resource pool. This property is used in relative allocation between resource consumers. This parameter is ignored unless the MemSharesLevel parameter is set to Custom.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VApp -Name MyTestVApp1 | Set-VApp -CpuSharesLevel Low -MemSharesLevel Normal

Modifies the CpuSharesLevel and MemSharesLevel properties of the MyTestVApp1 virtual appliance.

————— Example 2 —————

C:PS>$myvApp = Get-VApp -Location MyDatacenter1 Set-VApp -VApp $myvApp -CpuExpandableReservation:$true -CpuLimitMhz 4000 -MemExpandableReservation:$true -MemLimitGB 2

Modifies the properties of the vApps available on the MyDatacenter1 datacenter.

REMARKS To see the examples, type: "get-help Set-VApp -examples". For more information, type: "get-help Set-VApp -detailed". For technical information, type: "get-help Set-VApp -full". For online help, type: "get-help Set-VApp -online"

Set-VDBlockedPolicy

NAME Set-VDBlockedPolicy

SYNOPSIS This cmdlet modifies the blocking policy for distributed ports.

SYNTAX Set-VDBlockedPolicy [-Policy] <BlockedPolicy[]> [-Blocked <Boolean>] [-BlockedInherited <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the blocking policy for distributed ports at switch, port group, or port level (depending on the input policy).

PARAMETERS

-Policy <BlockedPolicy[]> Specifies the blocking policy that you want to configure.

-Blocked [<Boolean>] Specifies whether packet forwarding is stopped for the corresponding distributed port, port group, or switch.

-BlockedInherited [<Boolean>] Specifies whether the Blocked setting is inherited from a parent object, such as a distributed port group or switch.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDPort -VDSwitch “MyVDSwitch” -ActiveOnly | Get-VDBlockedPolicy | Set-VDBlockedPolicy -Blocked

Retrieves all active distributed ports and updates their blocking policy to stop packet forwarding through them.

————— Example 2 —————

C:PS>Get-VDportgroup “MyVDPortgroup” | Get-VDPort | Get-VDBlockedPolicy | Set-VDBlockedPolicy -BlockedInherited

Retrieves the specified “MyVDPortgroup” distributed port group and updates the blocking policy of all ports inside the group to inherit the setting for packet forwarding from the one set on the distributed port group level.

REMARKS To see the examples, type: “get-help Set-VDBlockedPolicy -examples”. For more information, type: “get-help Set-VDBlockedPolicy -detailed”. For technical information, type: “get-help Set-VDBlockedPolicy -full”. For online help, type: “get-help Set-VDBlockedPolicy -online”

Set-VDPort

NAME Set-VDPort

SYNOPSIS This cmdlet modifies the configuration of virtual distributed ports.

SYNTAX Set-VDPort [-VDPort] <VDPort[]> [-Name <String>] [-Description <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of virtual distributed ports.

PARAMETERS

-VDPort <VDPort[]> Specifies the virtual distributed port that you want to configure.

-Name <String> Specifies a new name for the virtual distributed port that you want to configure.

-Description <String> Specifies a description for the virtual distributed port that you want to configure.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$myVDPort = Get-VDPort -Key "Port0" -VDSwtich "MyVDSwitch" Set-VDPort -VDPort $myVDPort -Name "MyUpdatedPortName" -Description "MyUpdatedVDPortDescription"

Updates the name and the description of a specified virtual distributed port inside a vSphere distributed switch named "MyVDSwitch".

REMARKS To see the examples, type: "get-help Set-VDPort -examples". For more information, type: "get-help Set-VDPort -detailed". For technical information, type: "get-help Set-VDPort -full". For online help, type: "get-help Set-VDPort -online"

Set-VDPortgroup

NAME Set-VDPortgroup

SYNOPSIS This cmdlet modifies the configuration of distributed port groups.

SYNTAX Set-VDPortgroup [-Name <String>] [-Notes <String>] [-NumPorts <Int32>] [-VlanId <Int32>] [-VlanTrunkRange <VlanRangeList>] [-PrivateVlanId <Int32>] [-PortBinding <DistributedPortGroupPortBinding>] [-DisableVlan] [-VDPortgroup] <VDPortgroup[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VDPortgroup -RollbackConfiguration [-VDPortgroup] <VDPortgroup[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VDPortgroup -BackupPath <String> [-VDPortgroup] <VDPortgroup[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of distributed port groups. You can set the properties of the distributed port group manually, provide a backup profile to import the port group configuration, or rollback to the last valid configuration.

Note: You can import or rollback a configuration only on vSphere 5.1 and later.

PARAMETERS

-Name <String> Specifies a new name for the distributed port group that you want to configure.

-Notes <String> Specifies a new description for the distributed port group that you want to configure.

-NumPorts <Int32> Specifies a new number of ports on the distributed port group that you want to configure.

-VlanId <Int32> Specifies a new VLAN ID for the distributed port group that you want to configure. The VLAN IDs of 0 and 4095 are reserved and cannot be used. This parameter is obsolete. Use the corresponding parameter from the Set-VDVlanConfiguration cmdlet instead.

-VlanTrunkRange <VlanRangeList> Specifies a new VLAN trunk range for the distributed port group that you want to configure. Valid values are strings representing ranges of IDs. For example, "1-4, 6, 8-9". This parameter is obsolete. Use the corresponding parameter from the Set-VDVlanConfiguration cmdlet instead.

-PrivateVlanId <Int32> Specifies the secondary VLAN ID of a vSphere distributed switch's private VLAN configuration entry. This parameter is obsolete. Use the corresponding parameter from the Set-VDVlanConfiguration cmdlet instead.

-PortBinding <DistributedPortGroupPortBinding> Specifies a new port binding setting for the distributed port group that you want to configure. This parameter accepts Static, Dynamic, and Ephemeral values.

Note: Dynamic port binding is deprecated. For better performance, static port binding is recommended.

-DisableVlan Sets the VLAN type of the distributed port group to None. This parameter is obsolete. Use the corresponding parameter from the Set-VDVlanConfiguration cmdlet instead.

-VDPortgroup <VDPortgroup[]> Specifies the distributed port group that you want to configure.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-RollbackConfiguration Indicates that you want to rollback the distributed port group to its last valid configuration.

Note: Rollback is available only on vSphere 5.1 and later.

-BackupPath <String> Specifies the full file path to the .zip file containing the backup configuration that you want to import. You can import only .zip files created with the Export-VDPortgroup cmdlet.

Note: This parameter is only supported on vSphere 5.1 and later.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDPortgroup -Name "MyVDPortGroup" | Set-VDPortgroup -Name "MyNewVDPortGroupName" -NumPorts 5 -VlanId 4

Changes the name, number of ports and the VLAN ID of all distributed port groups named "MyVDPortGroup".

———— Example 2 ————

C:PS>Get-VDPortgroup -Name "MyVDPortGroup" | Set-VDPortgroup -VlanTrunkRange "1-5, 8-10"

Changes the VLAN trunk range of all distributed port groups named "MyVDPortGroup".

———— Example 3 ————

C:PS>$myVDPortgroup = Get-VDPortgroup -Name "MyVDPortGroup" -VDSwitch "MyVDSwitch" Set-VDPortgroup -VDPortgroup $myVDPortgroup -DisableVlan

Sets the VLAN type of the specified distributed port group to None.

———— Example 4 ————

C:PS>Get-VDPortgroup -Name "MyVDPortGroup" | Set-VDPortgroup -RollbackConfiguration

Rollbacks the configuration of all distributed port groups named "MyVDPortGroup".

———— Example 5 ————

C:PS>Get-VDPortgroup -Name "MyVDPortGroup" | Set-VDPortgroup -BackupPath 'c:backup.zip'

Reconfigures all distributed port groups named "MyVDPortGroup" by importing the configuration from the specified backup profile.

REMARKS To see the examples, type: "get-help Set-VDPortgroup -examples". For more information, type: "get-help Set-VDPortgroup -detailed". For technical information, type: "get-help Set-VDPortgroup -full". For online help, type: "get-help Set-VDPortgroup -online"

Set-VDPortgroupOverridePolicy

NAME Set-VDPortgroupOverridePolicy

SYNOPSIS This cmdlet modifies the policy for overriding port group settings at port level.

SYNTAX Set-VDPortgroupOverridePolicy [-Policy] <VDPortgroupOverridePolicy[]> [-BlockOverrideAllowed <Boolean>] [-ResetPortConfigAtDisconnect <Boolean>] [-SecurityOverrideAllowed <Boolean>] [-TrafficShapingOverrideAllowed <Boolean>] [-UplinkTeamingOverrideAllowed <Boolean>] [-VlanOverrideAllowed <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the policy for overriding port group settings at port level. At least one of the Bool parameters must be specified.

PARAMETERS

-Policy <VDPortgroupOverridePolicy[]> Specifies the port group overriding policy that you want to configure.

-BlockOverrideAllowed [<Boolean>] Specifies whether overriding port blocking settings is allowed.

-ResetPortConfigAtDisconnect [<Boolean>] Specifies whether the port configuration is reset when the port is disconnected.

-SecurityOverrideAllowed [<Boolean>] Specifies whether overriding security settings is allowed.

-TrafficShapingOverrideAllowed [<Boolean>] Specifies whether overriding traffic shaping settings is allowed.

-UplinkTeamingOverrideAllowed [<Boolean>] Specifies whether overriding uplink teaming settings is allowed.

-VlanOverrideAllowed [<Boolean>] Specifies whether overriding VLAN settings is allowed.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDPortgroup "MyVDPortgroup" | Get-VDPortgroupOverridePolicy | Set-VDPortgroupOverridePolicy -BlockOverrideAllowed $true

Retrieves a distributed port group named "MyVDPortgroup" and updates its overriding policy to allow the port blocking settings to override the default settings at port group level.

————— Example 2 —————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDPortgroup | Get-VDPortgroupOverridePolicy | Set-VDPortgroupOverridePolicy -ResetPortConfigAtDisconnect $true -TrafficShapingOverrideAllowed $true

Retrieves all port groups inside a distributed switch named "MyVDSwitch" and updates their overriding policies with the options to override the traffic shaping setting at port level, and to reset the distributed port network settings back to the port group settings.

REMARKS To see the examples, type: "get-help Set-VDPortgroupOverridePolicy -examples". For more information, type: "get-help Set-VDPortgroupOverridePolicy -detailed". For technical information, type: "get-help Set-VDPortgroupOverridePolicy -full". For online help, type: "get-help Set-VDPortgroupOverridePolicy -online"

Set-VDSecurityPolicy

NAME Set-VDSecurityPolicy

SYNOPSIS This cmdlet modifies the security policy for distributed ports.

SYNTAX Set-VDSecurityPolicy [-Policy] <SecurityPolicy[]> [-AllowPromiscuous <Boolean>] [-AllowPromiscuousInherited <Boolean>] [-ForgedTransmits <Boolean>] [-ForgedTransmitsInherited <Boolean>] [-MacChanges <Boolean>] [-MacChangesInherited <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the security policy for distributed ports or the default port policy at port group or switch level (depending on the input policy).

PARAMETERS

-Policy <SecurityPolicy[]> Specifies the security policy that you want to configure.

-AllowPromiscuous [<Boolean>] Specifies whether promiscuous mode is enabled for the corresponding distributed port, port group, or switch.

-AllowPromiscuousInherited [<Boolean>] Specifies whether the AllowPromiscuous setting is inherited from a parent object, such as a distributed port group or switch.

-ForgedTransmits [<Boolean>] Specifies whether forged transmits are enabled for the corresponding distributed port, port group, or switch.

-ForgedTransmitsInherited [<Boolean>] Specifies whether the ForgedTransmits setting is inherited from a parent object, such as a distributed port group or switch.

-MacChanges [<Boolean>] Specifies whether MAC address changes are enabled for the corresponding distributed port, port group, or switch.

-MacChangesInherited [<Boolean>] Specifies whether the MacChanges setting is inherited from a parent object, such as a distributed port group or switch.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch “MyVDSwitch” | Get-VDSecurityPolicy | Set-VDSecurityPolicy -MacChanges $true

Retrieves a vSphere distributed switch named “MyVDSwitch” and updates its security policy to allow MAC address changes.

————— Example 2 —————

C:PS>Get-VDPortgroup “MyVDPortgroup” | Get-VDSecurityPolicy | Set-VDSecurityPolicy -ForgedTransmitsInherited $true

Retrieves a distributed port group named “MyVDPortgroup” and updates the security policy to inherit the setting value for controlling outbound frames filtering by MAC address from its parent.

REMARKS To see the examples, type: “get-help Set-VDSecurityPolicy -examples”. For more information, type: “get-help Set-VDSecurityPolicy -detailed”. For technical information, type: “get-help Set-VDSecurityPolicy -full”. For online help, type: “get-help Set-VDSecurityPolicy -online”

Set-VDSwitch

NAME Set-VDSwitch

SYNOPSIS This cmdlet modifies the configuration of vSphere distributed switches.

SYNTAX Set-VDSwitch [-Name <String>] [-ContactDetails <String>] [-ContactName <String>] [-LinkDiscoveryProtocol <LinkDiscoveryProtocol>] [-LinkDiscoveryProtocolOperation <LinkDiscoveryOperation>] [-MaxPorts <Int32>] [-Mtu <Int32>] [-Notes <String>] [-NumUplinkPorts <Int32>] [-Version <String>] [-VDSwitch] <VDSwitch[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VDSwitch -BackupPath <String> [-WithoutPortGroups] [-VDSwitch] <VDSwitch[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VDSwitch -RollBackConfiguration [-VDSwitch] <VDSwitch[]> [-RunAsync] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of vSphere distributed switches. You can set the properties of the vSphere distributed switch manually, rollback the configuration to its previous state, or import it from a backup profile.

Note: Rollback and importing are available only on vSphere 5.1 and later.

PARAMETERS

-Name <String> Specifies a new name for the vSphere distributed switch that you want to configure.

-ContactDetails <String> Specifies new contact details of the vSphere distributed switch administrator.

-ContactName <String> Specifies a new name for the vSphere distributed switch administrator.

-LinkDiscoveryProtocol <LinkDiscoveryProtocol> Specifies the link discovery protocol for the vSphere distributed switch that you want to configure. This parameter accepts CDP and LLDP values.

-LinkDiscoveryProtocolOperation <LinkDiscoveryOperation> Specifies the link discovery protocol operation for the vSphere distributed switch that you want to configure. This parameter accepts Advertise, Listen, Both, and Disabled values.

-MaxPorts <Int32> Specifies the maximum number of ports allowed on the vSphere distributed switch that you want to configure.

-Mtu <Int32> Specifies the maximum MTU size for the vSphere distributed switch that you want to configure. Valid values are positive integers only.

-Notes <String> Specifies a new description for the vSphere distributed switch that you want to configure.

-NumUplinkPorts <Int32> Specifies the number of uplink ports on the vSphere distributed switch that you want to configure.

-Version <String> Specifies a new version for the vSphere distributed switch that you want to configure. This parameter accepts 4.0, 4.1.0, 5.0.0, 5.1.0, 5.5.0, and 6.0.0 values. You cannot specify a version that is incompatible with the version of the vCenter Server system you are connected to.

-VDSwitch <VDSwitch[]> Specifies the vSphere distributed switch that you want to configure.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-BackupPath <String> Specifies the full file path to the .zip file containing the backup configuration that you want to import. You can import only .zip files created with the Export-VDSwitch cmdlet.

Note: This parameter is supported only on vSphere 5.1 and later.

-WithoutPortGroups Indicates that the specified backup configuration is imported without its port groups.

Note: This parameter is supported only on vSphere 5.1 and later.

-RollBackConfiguration Indicates that you want to rollback the configuration of the vSphere distributed switch to an earlier state.

Note: This parameter is supported only on vSphere 5.1 and later.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch -Name “MyVDSwitch” | Set-VDSwitch -MaxPorts 1000 -NumUplinkPorts 8 -Mtu 2000

Modifies the maximum number of ports, the number of uplink ports, and the maximum MTU size of the specified vSphere distributed switch.

————— Example 2 —————

C:PS>$myVDSwitches = Get-VDSwitch -Name MyVDSwitch* Set-VDSwitch -VDSwitch $myVDSwitches -Version '5.1.0'

Changes the version of all vSphere distributed switches whose names start with "MyVDSwitch".

————— Example 3 —————

C:PS>$myVDSwitch = Get-VDSwitch -Name "MyVDSwitch" Set-VDSwitch -VDSwitch $myVDSwitch -LinkDiscoveryProtocol LLDP -LinkDiscoveryProtocolOperation Listen

Enables link discovery protocol on the specified vSphere distributed switch, sets it to LLDP and changes the link discovery protocol operation to listen.

————— Example 4 —————

C:PS>Get-VDSwitch -Name "MyVDSwitch" | Set-VDSwitch -BackupPath 'c:MyVDSwitchBackupsMyVDSwitch_12_12_2012.zip' -WithoutPortGroups

Reconfigures the specified vSphere distributed switch by importing the specified backup profile. The original port groups are not recreated.

————— Example 5 —————

C:PS>Get-VDSwitch -Name "MyVDSwitch" | Set-VDSwitch -RollbackConfiguration

Rollbacks the configuration of the specified vSphere distributed switch to its previous state.

REMARKS To see the examples, type: "get-help Set-VDSwitch -examples". For more information, type: "get-help Set-VDSwitch -detailed". For technical information, type: "get-help Set-VDSwitch -full". For online help, type: "get-help Set-VDSwitch -online"

Set-VDTrafficShapingPolicy

NAME Set-VDTrafficShapingPolicy

SYNOPSIS This cmdlet modifies the traffic shaping policy for distributed ports.

SYNTAX Set-VDTrafficShapingPolicy [-Policy] <TrafficShapingPolicy[]> [-Enabled <Boolean>] [-EnabledInherited <Boolean>] [-AverageBandwidth <Int64>] [-AverageBandwidthInherited <Boolean>] [-BurstSize <Int64>] [-BurstSizeInherited <Boolean>] [-PeakBandwidth <Int64>] [-PeakBandwidthInherited <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the traffic shaping policy for distributed ports or the default port policy at port group or switch level (depending on the input policy).

PARAMETERS

-Policy <TrafficShapingPolicy[]> Specifies the traffic shaping policy that you want to configure.

-Enabled [<Boolean>] Specifies whether traffic shaping is enabled for the corresponding distributed port, port group, or switch.

-EnabledInherited [<Boolean>] Specifies whether the Enabled setting is inherited from a parent object, such as a distributed port group or switch.

-AverageBandwidth <Int64> Specifies the average bandwidth of the traffic shaping policy for the corresponding distributed port, port group, or switch. The value is in bits per second.

-AverageBandwidthInherited [<Boolean>] Specifies whether the AverageBandwidth setting is inherited from a parent object, such as a distributed port group or switch.

-BurstSize <Int64> Specifies the burst size of the traffic shaping policy for the corresponding distributed port, port group, or switch. The value is in bits per second.

-BurstSizeInherited [<Boolean>] Specifies whether the BurstSize setting is inherited from a parent object, such as a distributed port group or switch.

-PeakBandwidth <Int64> Specifies the peak bandwidth of the traffic shaping policy for the corresponding distributed port, port group, or switch. The value is in bits per second.

-PeakBandwidthInherited [<Boolean>] Specifies whether the PeakBandwidth setting is inherited from a parent object, such as a distributed port group or switch.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch “MyVDSwitch” | Get-VDPortgroup “MyVDPortgroup” | Get-VDPort -Key 4| Get-VDTrafficShapingPolicy -Direction In | Set-VDTrafficShapingPolicy -Enabled $true -AverageBandwidth 100000

Enables traffic shaping for a specific port in a distributed port group named “MyVDPortgroup” and updates the average bandwidth settings in their traffic shaping policies.

———— Example 2 ————

C:PS>Get-VDPortgroup “MyVDPortgroup” | Get-VDTrafficShapingPolicy | Set-VDTrafficShapingPolicy -BurstSizeInherited

Retrieves a distributed port group named “MyVDPortgroup” and updates its traffic shaping policy by inheriting the burst size from its corresponding parent.

REMARKS To see the examples, type: “get-help Set-VDTrafficShapingPolicy -examples”. For more information, type: “get-help Set-VDTrafficShapingPolicy -detailed”. For technical information, type: “get-help Set-VDTrafficShapingPolicy -full”. For online help, type: “get-help Set-VDTrafficShapingPolicy -online”

Set-VDUplinkLacpPolicy

NAME Set-VDUplinkLacpPolicy

SYNOPSIS This cmdlet modifies the Link Aggregation Control Protocol policy for uplink ports.

SYNTAX Set-VDUplinkLacpPolicy [-Policy] <UplinkLacpPolicy[]> [-Enabled <Boolean>] [-EnabledInherited <Boolean>] [-Mode <UplinkLacpMode>] [-ModeInherited <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the Link Aggregation Control Protocol policy for uplink ports at switch or uplink port group level.

PARAMETERS

-Policy <UplinkLacpPolicy[]> Specifies the Link Aggregation Control Protocol policy that you want to configure.

-Enabled [<Boolean>] Specifies whether the Link Aggregation Control Protocol is enabled for the corresponding uplink port group or vSphere distributed switch.

-EnabledInherited [<Boolean>] Specifies whether the Enabled setting is inherited from a parent object, such as a vSphere distributed switch.

-Mode <UplinkLacpMode> Specifies the mode of the Link Aggregation Control Protocol policy for the corresponding uplink port group or vSphere distributed switch. The value can be Active or Passive.

-ModeInherited [<Boolean>] Specifies whether the Mode setting is inherited from a parent object, such as a vSphere distributed switch.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VDSwitch “MyVDSwitch” | Get-VDUplinkLacpPolicy | Set-VDUplinkLacpPolicy -Enabled $true -Mode Active

Retrieves a vSphere distributed switch named “MyVDSwitch” and updates its policy to enable the Link Aggregation Control Protocol, and sets the corresponding policy mode to active.

————— Example 2 —————

C:PS>Get-VDPortgroup “MyVDPortgroup” | Get-VDUplinkLacpPolicy | Set-VDUplinkLacpPolicy -ModeInherited $true

Retrieves a distributed port group named “MyVDPortgroup” and updates its Link Aggregation Control Protocol mode based on its parent object’s setting values.

REMARKS To see the examples, type: “get-help Set-VDUplinkLacpPolicy -examples”. For more information, type: “get-help Set-VDUplinkLacpPolicy -detailed”. For technical information, type: “get-help Set-VDUplinkLacpPolicy -full”. For online help, type: “get-help Set-VDUplinkLacpPolicy -online”

Set-VDUplinkTeamingPolicy

NAME Set-VDUplinkTeamingPolicy

SYNOPSIS This cmdlet modifies the uplink teaming policy for distributed ports.

SYNTAX Set-VDUplinkTeamingPolicy [-ActiveUplinkPort <String[]>] [-ObsoleteParameterDisableFailback <Boolean>] [-EnableFailback <Boolean>] [-FailbackInherited <Boolean>] [-FailoverDetectionPolicy <NetworkFailoverDetectionPolicy>] [-FailoverDetectionPolicyInherited <Boolean>] [-LoadBalancingPolicy <LoadBalancingPolicy>] [-LoadBalancingPolicyInherited <Boolean>] [-NotifySwitches <Boolean>] [-NotifySwitchesInherited <Boolean>] [-Policy] <UplinkTeamingPolicy[]> [-StandbyUplinkPort <String[]>] [-UnusedUplinkPort <String[]>] [-UplinkPortOrderInherited <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the uplink teaming policy for distributed ports at switch, port group, or port level.

PARAMETERS

-ActiveUplinkPort <String[]> Specifies the active uplink ports used for load balancing for a corresponding vSphere distributed switch.

-ObsoleteParameterDisableFailback [<Boolean>] This parameter and its alias, Failback, are obsolete and should not be used. They exist for compatibility with earlier PowerCLI versions. Use the EnableFailback parameter instead. Note that passing $false to the obsolete Failback parameter is equivalent to passing $true to the new EnableFailback parameter and the reverse.

-EnableFailback [<Boolean>] Specifies whether to use failback when restoring links. If, for example, the explicit link order is (vmnic9, vmnic0), and vmnic9 goes down, vmnic0 is activated. However, when vmnic9 comes back up, if EnableFailback is set to $true, vmnic9 is restored as specified in the explicit order. If EnableFailback is set to $false, vmnic0 continues to be used and vmnic9 remains on standby.

This parameter replaces the obsolete ObsoleteParameterDisableFailback parameter, which is an alias fort the Failback parameter. Note that passing $false to the obsolete Failback parameter is equivalent to passing $true to the new EnableFailback parameter and the reverse.

-FailbackInherited [<Boolean>] Specifies whether the Failback setting is inherited from a parent object, such as a distributed port group or switch.

-FailoverDetectionPolicy <NetworkFailoverDetectionPolicy> Specifies the method to use for failover detection for the corresponding distributed port, port group, or switch. The value can be LinkStatus or BeaconProbing.

-FailoverDetectionPolicyInherited [<Boolean>] Specifies whether the FailoverDetectionPolicy setting is inherited from a parent object, such as a distributed port group or switch.

-LoadBalancingPolicy <LoadBalancingPolicy> Specifies the load balancing policy for the corresponding distributed port, port group, or switch. The value can be LoadBalanceIP, LoadBalanceSrcMac, LoadBalanceSrcId, or ExplicitFailover.

-LoadBalancingPolicyInherited [<Boolean>] Specifies whether the LoadBalancingPolicy setting is inherited from a parent object, such as a distributed port group or switch.

-NotifySwitches [<Boolean>] Specifies whether to notify switches in the case of failover.

-NotifySwitchesInherited [<Boolean>] Specifies whether the NotifySwitches setting is inherited from a parent object, such as a distributed port group or switch.

-Policy <UplinkTeamingPolicy[]> Specifies the uplink teaming policy that you want to configure.

-StandbyUplinkPort <String[]> Specifies the standby uplink ports for the corresponding vSphere distributed switch.

-UnusedUplinkPort <String[]> Specifies the unused uplink ports for the corresponding vSphere distributed switch.

-UplinkPortOrderInherited [<Boolean>] Specifies whether the UplinkPortOrder setting is inherited from a parent object, such as a distributed port group or switch.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$activePortsList = "Port0" $standbyPortsList = "Port1", "Port2" Get-VDSwitch "MyVDSwitch" | Get-VDPortgroup | Get-VDUplinkTeamingPolicy | Set-VDUplinkTeamingPolicy -ActiveUplinkPort $activePortsList -StandbyUplinkPort $standbyPortsList

Retrieves all distributed port groups from a vSphere distributed switch named "MyVDSwitch" and updates their teaming policies with information about the active and standby uplinks that will be used when the adapter connectivity is up or down.

———— Example 2 ————

C:PS>Get-VDPortgroup "MyVDPortgroup" | Get-VDUplinkTeamingPolicy | Set-VDUplinkTeamingPolicy -NotifySwitches $true -FailoverDetectionPolicy LinkStatus

Updates the teaming policy of a distributed port group named "MyVDPortgroup" with a link status method for failover detection and with the ability to notify the corresponding distributed switch when any adapter failover occurs.

———— Example 3 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDPort "Port2" | Get-VDUplinkTeamingPolicy | Set-VDUplinkTeamingPolicy -LoadBalancingPolicy LoadBalanceSrcId -FailBack $true

Updates the uplink teaming policy of a distributed port named "Port2" inside the "MyVDSwitch" vSphere distributed switch to enable failback and to choose an uplink based on the current loads of physical NICs.

REMARKS To see the examples, type: "get-help Set-VDUplinkTeamingPolicy -examples". For more information, type: "get-help Set-VDUplinkTeamingPolicy -detailed". For technical information, type: "get-help Set-VDUplinkTeamingPolicy -full". For online help, type: "get-help Set-VDUplinkTeamingPolicy -online"

Set-VDVlanConfiguration

NAME Set-VDVlanConfiguration

SYNOPSIS This cmdlet modifies the virtual distributed port's VLAN configuration.

SYNTAX Set-VDVlanConfiguration -VDPortgroup <VDPortgroup[]> [-DisableVlan] [-VlanId <Int32>] [-VlanTrunkRange <VlanRangeList>] [-PrivateVlanId <Int32>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VDVlanConfiguration -VDSwitch <VDSwitch[]> [-DisableVlan] [-VlanId <Int32>] [-VlanTrunkRange <VlanRangeList>] [-PrivateVlanId <Int32>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VDVlanConfiguration -VDPort <VDPort[]> [-DisableVlan] [-VlanId <Int32>] [-VlanTrunkRange <VlanRangeList>] [-PrivateVlanId <Int32>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the virtual distributed port's VLAN configuration. For vSphere distributed switch and port group parameter sets, the cmdlet modifies the respective default port configuration.

PARAMETERS

-VDPortgroup <VDPortgroup[]> Specifies the port group whose default VLAN port configuration you want to modify.

-DisableVlan	Sets the VLAN type to None.

-VlanId <Int32> Specifies a new VLAN ID. The VLAN IDs of 0 and 4095 are reserved and cannot be used.

-VlanTrunkRange <VlanRangeList> Specifies a new VLAN trunk range. Valid values are strings representing ranges of IDs. For example, "1-4, 6, 8-9".

-PrivateVlanId <Int32> Specifies the secondary VLAN ID of a vSphere distributed switch's private VLAN configuration entry. The VLAN IDs of 0 and 4095 are reserved and cannot be used.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-VDSwitch <VDSwitch[]> Specifies the vSphere distributed switch whose default VLAN port configuration you want to modify.

-VDPort <VDPort[]> Specifies the port whose VLAN configuration you want to modify.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VDSwitch "MyVDSwitch" | Get-VDPort -ActiveOnly | Set-VDVlanConfiguration -PrivateVlanId 4

Assigns all active ports of a specific vSphere distributed switch named "MyVDSwitch" to a private VLAN with ID "4".

———— Example 2 ————

C:PS>Get-VDPortgroup "MyVDPorgroup" | Get-VDPort | Set-VDVlanConfiguration -VlanId 3

Assigns all ports of a specific distributed port group named "MyVDPorgroup" to a VLAN with ID "3".

REMARKS To see the examples, type: "get-help Set-VDVlanConfiguration -examples". For more information, type: "get-help Set-VDVlanConfiguration -detailed". For technical information, type: "get-help Set-VDVlanConfiguration -full". For online help, type: "get-help Set-VDVlanConfiguration -online"

Set-VIPermission

NAME Set-VIPermission

SYNOPSIS This cmdlet modifies the properties of the specified permissions.

SYNTAX Set-VIPermission [-Permission] <Permission[]> [-Role <Role>] [-Propagate <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified permissions. The cmdlet can change the role and define whether the permission propagates down the hierarchy to child inventory objects.

PARAMETERS

-Permission <Permission[]> Specifies the permissions you want to modify.

-Role <Role> Specifies a new role for the permissions.

-Propagate [<Boolean>] Indicates that you want to propagate the new permissions to the child inventory objects.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VIPermission -Permission $permission -Role Role -Propagate:$false

Changes the Propagate property of the $permission permission to $false.

REMARKS To see the examples, type: "get-help Set-VIPermission -examples". For more information, type: "get-help Set-VIPermission -detailed". For technical information, type: "get-help Set-VIPermission -full". For online help, type: "get-help Set-VIPermission -online"

Set-VIRole

NAME Set-VIRole

SYNOPSIS This cmdlet modifies the privileges of the provided roles.

SYNTAX Set-VIRole [-Role] <Role[]> [-Name <String>] [-AddPrivilege <Privilege[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VIRole [-Role] <Role[]> [-Name <String>] [-RemovePrivilege <Privilege[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the privileges of the provided roles.

PARAMETERS

-Role <Role[]> Specifies the roles you want to modify.

-Name <String> Provides a new name for the provided role.

-AddPrivilege <Privilege[]> Specifies privileges and privilege groups you want to add to the provided roles.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-RemovePrivilege <Privilege[]> Specifies privileges or privilege groups you want to remove from the provided roles.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Set-VIRole -Role Role -AddPrivilege (Get-VIPrivilege -Name ‘Create Datacenter’)

Adds the “Create Datacenter” privilege to the Role role.

REMARKS To see the examples, type: “get-help Set-VIRole -examples”. For more information, type: “get-help Set-VIRole -detailed”. For technical information, type: “get-help Set-VIRole -full”. For online help, type: “get-help Set-VIRole -online”

Set-VirtualPortGroup

NAME Set-VirtualPortGroup

SYNOPSIS This cmdlet modifies the properties of the specified virtual port group.

SYNTAX Set-VirtualPortGroup [-Name <String>] [-VLanId <Int32>] [-VirtualPortGroup] <VirtualPortGroup[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified virtual port group.

PARAMETERS

-Name <String> Specifies a new name for the virtual port group.

-VLanId <Int32> Specifies the VLAN ID for ports using this port group. The following values are valid:

0 - specifies that you do not want to associate the port group with a VLAN.

1 to 4094 - specifies a VLAN ID for the port group.

4095 - specifies that the port group should use trunk mode, which allows the guest operating system to manage its own VLAN tags.

-VirtualPortGroup <VirtualPortGroup[]> Specifies the virtual port group whose properties you want to change.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.112.36 -Name VSwitch

$vportgroup1 = New-VirtualPortGroup -VirtualSwitch $vswitch -Name VPortGroup1

$vportgroup2 = Set-VirtualPortGroup -VirtualPortGroup $vportgroup1 -VLanId 1

Creates a new virtual switch named VSwitch on the virtual machine host with IP address 10.23.112.36. Creates a new virtual port group for the new switch named VPortGroup1. Sets the VLAN ID for the ports using the VPortGroup1 group.

REMARKS To see the examples, type: “get-help Set-VirtualPortGroup -examples”. For more information, type: “get-help Set-VirtualPortGroup -detailed”. For technical information, type: “get-help Set-VirtualPortGroup -full”. For online help, type: “get-help Set-VirtualPortGroup -online”

Set-VirtualSwitch

NAME Set-VirtualSwitch

SYNOPSIS This cmdlet modifies the properties of the specified virtual switch.

SYNTAX Set-VirtualSwitch [-VirtualSwitch] <VirtualSwitch[]> [[-NumPorts] <Int32>] [[-Nic] <String[]>] [[-Mtu] <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the properties of the specified virtual switch. The server rounds the value of the NumPorts parameter up to the closest exact power of two, greater than the provided number. When updating NumPorts, the user needs to restart the ESX/ESXi host for the change to take effect.

PARAMETERS

-VirtualSwitch <VirtualSwitch[]> Specifies the virtual switch you want to configure.

-NumPorts <Int32> Specifies the VirtualSwitch port number. The value is rounded to the closest exact power of two, greater than the provided number (for example, if the user specifies 67, this number is rounded to 128). The ESX host to which the virtual switch belongs, must be restarted for the change to take effect. Note that the port number displayed in the vSphere Client might differ from the value that you specified for the NumPorts parameter.

Note: In ESX 5.5 or later, standard virtual switches are always elastic, so the NumPorts parameter is no longer applicable and its value is ignored.

-Nic <String[]> Specifies new network interface cards for the virtual switch. The old NICs are replaced by the specified ones.

-Mtu <Int32> Specifies the maximum transmission unit (MTU) associated with the specified virtual switch (in bytes). The MTU value must be greater than 0.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vswitch = New-VirtualSwitch -Host 10.23.115.67 -Name VSwitch

Set-VirtualSwitch -VirtualSwitch $vswitch -MTU 500

Creates a new virtual switch named VSwitch on the virtual machine host on IP address 10.23.115.67. Then sets the virtual switch MTU to 500.

————— Example 2 —————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.115.67

$networkAdapters = Get-VMHostNetworkAdapter -VMHost 10.23.115.67 -Physical

$phNic = $networkAdapters[0]

$vSwitch | Set-VirtualSwitch -Nic $phNic

Creates a new virtual switch named VSwitch on the virtual machine host on IP address 10.23.115.67. Then assigns to it a network adapter.

————— Example 3 —————

C:PS>Get-VMHost *.128 | Get-VirtualSwitch | Select-Object -First 1 | Set-VirtualSwitch -Nic vmnic5

Add a physical network adapter named 'vmnic5' to the first switch of the host. Note that the 'vmnic5' adapter must not be assigned to other virtual switches.

REMARKS To see the examples, type: "get-help Set-VirtualSwitch -examples". For more information, type: "get-help Set-VirtualSwitch -detailed". For technical information, type: "get-help Set-VirtualSwitch -full". For online help, type: "get-help Set-VirtualSwitch -online"

Set-VM

NAME Set-VM

SYNOPSIS This cmdlet modifies the configuration of the virtual machine.

SYNTAX Set-VM [-VM] <VirtualMachine[]> [-Name <String>] [-Version <VMVersion>] [-MemoryMB <Int64>] [-MemoryGB <Decimal>] [-NumCpu <Int32>] [-GuestId <String>] [-AlternateGuestName <String>] [-OSCustomizationSpec <OSCustomizationSpec>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] [-Server <VIServer[]>] [-RunAsync] [-VMSwapFilePolicy <VMSwapfilePolicy>] [-Notes <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VM [-VM] <VirtualMachine[]> [-Name <String>] [-Snapshot <Snapshot>] [-OSCustomizationSpec <OSCustomizationSpec>] [-HARestartPriority <HARestartPriority>] [-HAIsolationResponse <HAIsolationResponse>] [-DrsAutomationLevel <DrsAutomationLevel>] [-Server <VIServer[]>] [-RunAsync] [-VMSwapFilePolicy <VMSwapfilePolicy>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VM [-VM] <VirtualMachine[]> [-Name <String>] [-Server <VIServer[]>] [-RunAsync] [-ToTemplate] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of the virtual machine. If the OSCustomizationSpec parameter is used, the cmdlet customizes the virtual machine according to the specification. In addition, the cmdlet allows you to revert a virtual machine to a snapshot and convert a virtual machine to a template.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machine you want to configure.

-Name <String> Specifies a new name for the virtual machine.

-Version <VMVersion> Specifies the version to which you want to upgrade the virtual machine. The valid values are v4, v7, v8, v9, v10, and v11. You cannot downgrade to an earlier version.

-MemoryMB <Int64> This parameter is obsolete. Use MemoryGB instead. Specifies the memory size in megabytes (MB).

-MemoryGB <Decimal> Specifies the memory size in gigabytes (GB).

-NumCpu <Int32> Specifies the number of virtual CPUs.

-GuestId <String> Specifies the guest operating system of the virtual machine. The valid values for specific ESX versions are listed in the description of the VirtualMachineGuestOsIdentifier enumeration type in the vSphere API Reference available at http://www.vmware.com/support/developer/vc-sdk/. Depending on the hardware configuration of the host, some of the guest operating systems might be inapplicable.

-AlternateGuestName <String> Specifies the full name of the guest OS for the virtual machine if the value of the GuestID parameter is set to otherGuest or otherGuest64.

-OSCustomizationSpec <OSCustomizationSpec> Specifies a customization specification you want to apply to the virtual machine. This works only in 32-bit mode.

-HARestartPriority <HARestartPriority> Specifies the virtual machine HA restart priority. The valid values are Disabled, Low, Medium, High, and ClusterRestartPriority. VMware HA is a feature that detects failed virtual machines and automatically restarts them on alternative ESX hosts. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. Specifying this parameter is only supported when the virtual machine is inside a cluster. Otherwise, an error is generated.

-HAIsolationResponse <HAIsolationResponse> Indicates whether the virtual machine should be powered off if a host determines that it is isolated from the rest of the compute resource. The valid values are AsSpecifiedByCluster, PowerOff, and DoNothing. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. Specifying this parameter is only supported when the virtual machine is inside a cluster. Otherwise, an error is generated.

-DrsAutomationLevel <DrsAutomationLevel> Specifies a DRS (Distributed Resource Scheduler) automation level. The valid values are FullyAutomated, Manual, PartiallyAutomated, AsSpecifiedByCluster, and Disabled. Passing values to this parameter through a pipeline is deprecated and will be disabled in a future release. Specifying this parameter is only supported when the virtual machine is inside a cluster. Otherwise, an error is generated.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-VMSwapFilePolicy <VMSwapfilePolicy> Specifies the swapfile placement policy. The following values are valid:

InHostDataStore - Stores the swapfile in the datastore specified by the VMSwapfileDatastoreID property of the virtual machine host. If the VMSwapfileDatastoreID property is not set or indicates a datastore with insufficient free space, the swapfile is stored in the same directory as the virtual machine. This setting might degrade the VMotion performance.

WithVM - Stores the swapfile in the same directory as the virtual machine.

-Notes <String> Provide a description for the virtual machine. The alias of this parameter is Description.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Snapshot <Snapshot> Specifies a snapshot whose state you want to apply to the virtual machine.

-ToTemplate	Indicates that you want to convert the virtual machine to a template.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$template = Get-VM VM | Set-VM -ToTemplate -Name VMTemplate

Converts the VM virtual machine to a template and stores the template in the $template variable.

———— Example 2 ————

C:PS>Get-VM -Location ResourcePool01 | Set-VM -MemoryGB 2 -NumCPU 2

Upgrades the memory and CPU count of the virtual machines in ResourcePool01.

———— Example 3 ————

C:PS>Set-VM -VM VM -Version v7

Upgrades the virtual hardware version of the VM machine.

———— Example 4 ————

C:PS>$snapshot = Get-Snapshot -VM $vm -Name "Initial state"

Set-VM -VM $vm -Snapshot $snapshot

Revert the VM virtual machine to the "Initial state" snapshot.

———— Example 5 ————

C:PS>$spec = Get-OSCustomizationSpec -Name FinanceDepartmentSpec;

Set-VM -VM $vm -OSCustomizationSpec $spec

Apply a customization specification on the specified virtual machines.

———— Example 6 ————

C:PS>Set-VM $vm -Name "Web Server" -GuestID winNetStandardGuest -Description "Company's web server"

Changes the name, description, and guest ID of the specified virtual machine.

REMARKS To see the examples, type: "get-help Set-VM -examples". For more information, type: "get-help Set-VM -detailed". For technical information, type: "get-help Set-VM -full". For online help, type: "get-help Set-VM -online"

Set-VMGuestNetworkInterface

NAME Set-VMGuestNetworkInterface

SYNOPSIS This cmdlet configures the network settings of a virtual machine using VMware Tools.

SYNTAX Set-VMGuestNetworkInterface -VmGuestNetworkInterface <VMGuestNetworkInterface[]> [-WinsPolicy <DhcpPolicy>] [-Wins <String[]>] [-DnsPolicy <DhcpPolicy>] [-Dns <String[]>] [-IPPolicy <DhcpPolicy>] [[-Gateway] <Object>] [[-Netmask] <String>] [[-Ip] <IPAddress>] [-ToolsWaitSecs <Int32>] [-GuestPassword <SecureString>] [-GuestUser <String>] [-GuestCredential <PSCredential>] [-HostPassword <SecureString>] [-HostUser <String>] [-HostCredential <PSCredential>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use Invoke-VMScript instead.

This cmdlet configures the network settings of a virtual machine using VMware Tools. The cmdlet allows IP and routing configuration. You can modify Wins settings only for Windows virtual machines. The cmdlet sends a remote script which executes inside the virtual machine in the context of the specified user account. For a list of supported operating systems, see the PowerCLI User's Guide.

To run this cmdlet against vCenter Server/ESX/ESXi versions earlier than 5.0, you need to meet the following requirements: *You must run the cmdlet on the 32-bit version of Windows PowerShell. *You must have access to the ESX that hosts the virtual machine over TCP port 902. *For vCenter Server/ESX/ESXi versions earlier than 4.1, you need VirtualMachine.Interact.ConsoleInteract privilege. For vCenter Server/ESX/ESXi 4.1 and later, you need VirtualMachine.Interact.GuestControl privilege.

To run this cmdlet against vCenter Server/ESXi 5.0 and later, you need VirtualMachine.GuestOperations.Execute and VirtualMachine.GuestOperations.Modify privileges.

PARAMETERS

-VmGuestNetworkInterface <VMGuestNetworkInterface[]> Specifies the guest network interface you want to configure.

-WinsPolicy <DhcpPolicy> Specifies the Wins policy. The valid values are Static and Dhcp.

-Wins <String[]> Specifies WINS servers. Use this parameter only if the WinsPolicy parameter is set to Static.

-DnsPolicy <DhcpPolicy> Specifies the DNS policy. The valid values are Static and Dhcp.

-Dns <String[]> Specifies DNS addresses. Use this parameter only if The DnsPolicy parameter is set to Static.

-IPPolicy <DhcpPolicy> Specifies the IP policy. The valid values are Static and Dhcp.

-Gateway <Object> Specifies a gateway.

-Netmask <String> Specifies a network mask.

-Ip <IPAddress> Specifies an IP address. Use this parameter only if The IpPolicy parameter is set to Static.

-ToolsWaitSecs <Int32> Specifies the time in seconds to wait for a response from VMware Tools. If a non-positive value is provided, the system waits indefinitely.

-GuestPassword <SecureString> Specifies the password you want to use for authenticating with the guest OS.

-GuestUser <String> Specifies the user name you want to use for authenticating with the guest OS.

-GuestCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the guest OS. Do not use this parameter if the GuestUser and GuestPassword parameters are used.

-HostPassword <SecureString> Specifies the password you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostUser <String> Specifies the user name you want to use for authenticating with the host. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-HostCredential <PSCredential> Specifies a PSCredential object that contains credentials for authenticating with the host. Do not use this parameter if the HostUser and HostPassword parameters are used. You need to specify host credentials only if the version of the vCenter Server or ESX you are authenticating with is earlier than 4.0, or the VIX version you have installed is earlier than 1.10.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VMGuestNetworkInterface -VMGuestNetworkInterface $vmGuestNetworkInterface -GuestUser User -GuestPassword Pass02 -Netmask 255.255.255.255 -Gateway 10.23.112.58

Changes the net mask and the gateway of the specified guest network interface.

REMARKS To see the examples, type: "get-help Set-VMGuestNetworkInterface -examples". For more information, type: "get-help Set-VMGuestNetworkInterface -detailed". For technical information, type: "get-help Set-VMGuestNetworkInterface -full". For online help, type: "get-help Set-VMGuestNetworkInterface -online"

Set-VMHost

NAME Set-VMHost

SYNOPSIS This cmdlet modifies the configuration of the host.

SYNTAX Set-VMHost [-VMHost] <VMHost[]> [[-State] <VMHostState>] [-VMSwapfilePolicy <VMSwapfilePolicy>] [-VMSwapfileDatastore <Datastore>] [-Profile <VMHostProfile>] [-Evacuate] [-TimeZone <VMHostTimeZone>] [-LicenseKey <String>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the configuration of the host.

The State parameter is set to Connected. - If the host is currently connected and is not in a maintenance mode, it does nothing. - If the host is in a maintenance mode, it exits the maintenance mode. - If the host is not connected or is not responding, it tries to reconnect.

The State parameter is set to Disconnected. - If the host is currently connected, it attempts to disconnect. - If the host is not connected or not responding, it does nothing.

The State parameter is set to Maintenance. - If the host is currently connected and not in a maintenance mode, it enters a maintenance mode. - If the host is currently connected and in a maintenance mode, it does nothing. - If the host is not connected or not responding, it attempts to reconnect and enter maintenance mode.

PARAMETERS

-VMHost <VMHost[]> Specifies the host you want to configure.

-State <VMHostState> Specifies the state of the host. The valid values are Connected, Disconnected, and Maintenance. If there are powered on virtual machines on the host, you can set the host into a maintenance mode, only if it is part of a DRS-enabled cluster. Before entering maintenance mode, if the host is fully automated, the cmdlet first relocates all powered on virtual machines. If the host is not automated or partially automated, you must first generate a DRS recommendation and wait until all powered on virtual machines are relocated or powered off. In this case, you must specify the RunAsync parameter, otherwise an error is thrown.

-VMSwapfilePolicy <VMSwapfilePolicy> Specifies the swapfile placement policy. The following values are valid:

InHostDataStore - Stores the swapfile in the datastore specified by the VMSwapfileDatastoreID property of the virtual machine host. If the VMSwapfileDatastoreID property is not set or indicates a datastore with insufficient free space, the swapfile is stored in the same directory as the virtual machine. This setting might degrade the VMotion performance.

WithVM - Stores the swapfile in the same directory as the virtual machine.

-VMSwapfileDatastore <Datastore> Specifies a datastore that is visible to the host and can be used for storing swapfiles for the virtual machines that run on this host. Using a host-specific swap location might degrade the VMotion performance.

-Profile <VMHostProfile> Specifies a host profile you want to associate with the host. If the value of this parameter is $null, the current profile association is removed.

-Evacuate	If the value is $true, vCenter automatically reregisters the virtual machines that are compatible for reregistration. If they are not compatible, they remain on the host. If there are powered-on virtual machines that cannot be reregistered the operation waits until they are powered off manually. The Evacuate parameter is valid only when connected to a vCenter Server system and the State parameter is set to Maintenance. Also, the virtual machine host must be in a DRS-enabled cluster.

-TimeZone <VMHostTimeZone> Specifies the time zone for the host by using its name or by providing the corresponding time zone object. Time zone names support wildcards. If the wildcards match more than one time zones, an error is reported. Time zone objects can only be applied to the hosts they originate from.

-LicenseKey <String> Specifies the license key to be used by the host. You can set the host to evaluation mode by passing the 00000-00000-00000-00000-00000 evaluation key.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VMHost -VMHost Host -State "Disconnected"

Resets the state of the Host virtual host to disconnected.

————— Example 2 —————

C:PS>$cluster = Get-Cluster -VMHost Host

$task = Set-VMHost -VMHost Host -State "Maintenance" -RunAsync

Get-DrsRecommendation -Cluster $cluster | where {$_.Reason -eq "Host is entering maintenance mode"} | Apply-DrsRecommendation

$vmhost = Wait-Task $task

Activate a maintenance mode for a not automated host that is part of a DRS-enabled cluster and has powered on virtual machines on it.

REMARKS To see the examples, type: "get-help Set-VMHost -examples". For more information, type: "get-help Set-VMHost -detailed". For technical information, type: "get-help Set-VMHost -full". For online help, type: "get-help Set-VMHost -online"

Set-VMHostAccount

NAME Set-VMHostAccount

SYNOPSIS This cmdlet configures a host account.

SYNTAX Set-VMHostAccount [-GroupAccount] <HostGroupAccount[]> [-AssignUsers <String[]>] [-UnassignUsers <String[]>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostAccount [-UserAccount] <HostUserAccount[]> [-Password <String>] [-Description <String>] [-AssignGroups <String[]>] [-UnassignGroups <String[]>] [-GrantShellAccess <Boolean>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures a host account. When configuring a host user account, you can include or exclude the user from the specified groups. When configuring a host group account, you can include or exclude the specified users from this group.

PARAMETERS

-GroupAccount <HostGroupAccount[]> Specifies the host group account you want to configure. Starting with ESXi 5.1, you cannot configure group host accounts.

-AssignUsers <String[]> If a group host account is configured, specify the users you want to add to the account. Starting with ESXi 5.1, you cannot configure group host accounts.

-UnassignUsers <String[]> If a group host account is to be configured, specifies the users you want to remove from the account.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-UserAccount <HostUserAccount[]> Specifies the host user account you want to configure.

-Password <String> Specifies a new password for the account.

-Description <String> Provides a description of the specified account. The maximum length of the text is 255 symbols.

-AssignGroups <String[]> If a user host account is to be configured, specifies the group to which you want to add the account. Starting with ESXi 5.1, you cannot configure group host accounts.

-UnassignGroups <String[]> If a user host account is to be configured, specifies a group from which you want to remove the account. Starting with ESXi 5.1, you cannot configure group host accounts.

-GrantShellAccess [<Boolean>] Indicates that the account is allowed to access the ESX shell.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$myUserAccount = New-VMHostAccount -ID MyUser1 -Password MyPassword1 -UserAccount $myGroupAccount = New-VMHostAccount -ID MyGroup1 -GroupAccount -AssignUsers $myUserAccount Set-VMHostAccount -UserAccount $myUserAccount -UnassignGroups $myGroupAccount

Creates a user account with an ID MyUser1. Then creates a group account with an ID MyGroup1 and assigns the user account MyUser1 to the group account MyGroup1. Finally, excludes the MyUser1 account from the MyGroup1 account. Starting with ESXi 5.1, you cannot configure group host accounts.

REMARKS To see the examples, type: "get-help Set-VMHostAccount -examples". For more information, type: "get-help Set-VMHostAccount -detailed". For technical information, type: "get-help Set-VMHostAccount -full". For online help, type: "get-help Set-VMHostAccount -online"

Set-VMHostAdvancedConfiguration

NAME Set-VMHostAdvancedConfiguration

SYNOPSIS This cmdlet modifies the advanced configuration settings of a host.

SYNTAX Set-VMHostAdvancedConfiguration [[-Name] <String>] [[-Value] <Object>] [-VMHost] <VMHost[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostAdvancedConfiguration [[-NameValue] <Hashtable>] [-VMHost] <VMHost[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet is deprecated. Use New-AdvancedSetting, Set-AdvancedSetting, or Remove-AdvancedSetting instead.

This cmdlet modifies the advanced configuration settings of a host.

PARAMETERS

-Name <String> Specifies the name of the host configuration setting you want to change.

-Value <Object> Specifies a new value of the host configuration setting that you want to modify.

-VMHost <VMHost[]> Specifies the host whose advanced configuration settings you want to change.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-NameValue <Hashtable> Provides a hash table that maps values to settings.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHost 10.23.123.144 | Set-VmHostAdvancedConfiguration -Name Migrate.NetTimeout -Value ([system.int32] 10)

Change the migration timeout for the virtual machine host with an IP address 10.23.123.144.

————— Example 2 —————

C:PS>$migrationSettings = Get-VMHost 10.23.123.144| Get-VmHostAdvancedConfiguration -Name Migrate.*

Set-VmHostAdvancedConfiguration -VMHost 10.23.123.122 -NameValue $migrationSettings

Gets the advanced settings concerning migration from the host with an IP address 10.23.123.144 and applies them to the virtual machine host with an IP address 10.23.123.122.

———— Example 3 ————

C:PS>Set-VMHostAdvancedConfiguration -VMHost 10.23.112.120 -Name Migrate.Enabled -Value 1

Enable VMotion on a host using Set-VMHostAdvancedConfiguration.

REMARKS To see the examples, type: “get-help Set-VMHostAdvancedConfiguration -examples”. For more information, type: “get-help Set-VMHostAdvancedConfiguration -detailed”. For technical information, type: “get-help Set-VMHostAdvancedConfiguration -full”. For online help, type: “get-help Set-VMHostAdvancedConfiguration -online”

Set-VMHostAuthentication

NAME Set-VMHostAuthentication

SYNOPSIS This cmdlet modifies the host authentication information.

SYNTAX Set-VMHostAuthentication [-Domain] <String> [[-Username] <String>] [[-Password] <SecureString>] [-Credential <PSCredential>] -JoinDomain -VMHostAuthentication <VMHostAuthentication[]> [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostAuthentication [-Force] -LeaveDomain -VMHostAuthentication <VMHostAuthentication[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the host authentication information.

PARAMETERS

-Domain <String> Specifies a domain you want to join.

-Username <String> Specifies a user name for authentication.

-Password <SecureString> Specifies a password for authentication.

-Credential <PSCredential> Specifies a credential object for authentication.

-JoinDomain	Indicates whether you want to join the specified domain.

-VMHostAuthentication <VMHostAuthentication[]> Specifies the VMHostAuthentication object you want to modify.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.
-Force	If the value is $true, any existing permissions on the managed objects for Active Directory users are deleted and the cmdlet completes. If the value is $false, the cmdlet cannot run if there are any existing permissions on managed objects for Active Directory users.
-LeaveDomain	Indicates whether you want to leave the currently joined domain.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmhost | Get-VMHostAuthentication | Set-VMHostAuthentication -JoinDomain -Domain "DomainName.com" -User "Username1" -Password "Password1"

Include an ESX host in a domain.

————— Example 2 —————

C:PS>$vmhost | Get-VMHostAuthentication | Set-VMHostAuthentication -LeaveDomain

Exclude an ESX host from a domain.

————— Example 3 —————

C:PS>$vmhost | Get-VMHostAuthentication | Set-VMHostAuthentication -LeaveDomain -Force

Exclude an ESX host from a domain. If AD account permissions are defined on the host, they will be removed.

REMARKS To see the examples, type: "get-help Set-VMHostAuthentication -examples". For more information, type: "get-help Set-VMHostAuthentication -detailed". For technical information, type: "get-help Set-VMHostAuthentication -full". For online help, type: "get-help Set-VMHostAuthentication -online"

Set-VMHostDiagnosticPartition

NAME Set-VMHostDiagnosticPartition

SYNOPSIS This cmdlet activates or deactivates the diagnostic partitions of hosts.

SYNTAX Set-VMHostDiagnosticPartition [-Active] <Boolean> [-VMHostDiagnosticPartition] <VMHostDiagnosticPartition[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet activates or deactivates the diagnostic partitions of hosts.

PARAMETERS

-Active [<Boolean>] If the value of this parameter is $true, the partition state is changed to active. If the value is $false, the partition state is set to inactive.

-VMHostDiagnosticPartition <VMHostDiagnosticPartition[]> Specifies the host diagnostic partition you want to set.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$diagPartition = Get-VMHostDiagnosticPartition -VMHost $vmhost

$diagPartition | Set-VMHostDiagnosticPartition -Active $false -Confirm

Deactivates the active diagnostic partition of the specified host.

REMARKS To see the examples, type: "get-help Set-VMHostDiagnosticPartition -examples". For more information, type: "get-help Set-VMHostDiagnosticPartition -detailed". For technical information, type: "get-help Set-VMHostDiagnosticPartition -full". For online help, type: "get-help Set-VMHostDiagnosticPartition -online"

Set-VMHostFirewallDefaultPolicy

NAME Set-VMHostFirewallDefaultPolicy

SYNOPSIS This cmdlet sets the default policy for the specified host firewall.

SYNTAX Set-VMHostFirewallDefaultPolicy [[-AllowIncoming] <Boolean>] [[-AllowOutgoing] <Boolean>] [-Policy] <VMHostFirewallDefaultPolicy[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet sets the default policy for the specified host firewall. This policy specifies whether outgoing or incoming connections are allowed. At least one of the AllowIncoming and AllowOutgoing parameters must be set. When you configure the default firewall policy of an ESX/ESXi host version 5.0, you must provide the same value for the AllowIncoming and AllowOutgoing parameters.

PARAMETERS

-AllowIncoming [<Boolean>] If the value of this parameter is $true, all incoming connections are allowed. If the value is $false, all incoming connections are disallowed.

-AllowOutgoing [<Boolean>] If the value of this parameter is $true, all outcoming connections are allowed. If the value is $false, all outcoming connections are disallowed.

-Policy <VMHostFirewallDefaultPolicy[]> Specifies the host firewall default policy you want to apply.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$firewallpolicy = Get-VMHostFirewallDefaultPolicy -VMHost 10.23.123.100

Set-VMHostFirewallDefaultPolicy -Policy $firewallpolicy -AllowOutGoing $true | fl

Changes the default firewall policy of the virtual machine host with IP address 10.23.123.100, so that the outgoing connections are allowed.

VMHostId : HostSystem-host-8 IncomingEnabled : False OutgoingEnabled : True Client : VMware.VimAutomation.Client20.VimClient

REMARKS To see the examples, type: "get-help Set-VMHostFirewallDefaultPolicy -examples". For more information, type: "get-help Set-VMHostFirewallDefaultPolicy -detailed". For technical information, type: "get-help Set-VMHostFirewallDefaultPolicy -full". For online help, type: "get-help Set-VMHostFirewallDefaultPolicy -online"

Set-VMHostFirewallException

NAME Set-VMHostFirewallException

SYNOPSIS This cmdlet enables or disables host firewall exceptions.

SYNTAX Set-VMHostFirewallException [-Enabled] <Boolean> [-Exception] <VMHostFirewallException[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet enables or disables host firewall exceptions.

PARAMETERS

-Enabled [<Boolean>] If the value is $true, the specified firewall exceptions are enabled. If the value is $false, the firewall exceptions are disabled.

-Exception <VMHostFirewallException[]> Specifies the firewall exceptions you want to enable or disable.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$ftpFirewallExceptions = Get-VMHostFirewallException -VMHost $vmhost | where {$_.Name.StartsWith('FTP')}

$ftpFirewallExceptions | Set-VMHostFirewallException -Enabled $true

Enables the firewall exceptions for the FTP services on the specified host.

REMARKS To see the examples, type: "get-help Set-VMHostFirewallException -examples". For more information, type: "get-help Set-VMHostFirewallException -detailed". For technical information, type: "get-help Set-VMHostFirewallException -full". For online help, type: "get-help Set-VMHostFirewallException -online"

Set-VMHostFirmware

NAME Set-VMHostFirmware

SYNOPSIS This cmdlet configures hosts firmware settings.

SYNTAX Set-VMHostFirmware [-VMHost] <VMHost[]> [-BackupConfiguration] -DestinationPath <String> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostFirmware [-VMHost] <VMHost[]> [-ResetToDefaults] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostFirmware [-VMHost] <VMHost[]> [-Restore] [-SourcePath <String>] [-Force] [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures hosts firmware settings. If the BackupConfiguration parameter is set, backups the host configuration and downloads the bundle to the specified DestinationPath. In order to use the Restore and ResetToDefaults parameters, the host needs to be in maintenance mode. The Backup functionality of Set-VMHostFirmware is deprecated and scheduled for removal. For making backups, use the Get-VMHostFirmware cmdlet instead.

PARAMETERS

-VMHost <VMHost[]> Specifies the host whose firmware you want to configure (it must be an ESX visor).

-BackupConfiguration The Backup functionality of Set-VMHostFirmware is deprecated and scheduled for removal. For making backups, use the Get-VMHostFirmware cmdlet instead.

Indicates that you want to backup the host firmware configuration and download the bundle to the path specified by the DestinationPath parameter.

-DestinationPath <String> Specifies a destination path where to download the host configuration backup bundle if the BackupConfiguration parameter is set.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-ResetToDefaults Indicates that you want to reset all configuration settings, including the "admin" password, to the factory defaults. The host is rebooted immediately. The host needs to be in a maintenance in order to perform this operation.

-Restore Indicates that you want to restore the configuration of the host to the one that is specified in the provided bundle. The bundle is uploaded to the URL retrieved via Get-VMHostFirmware. This method resets all configuration options, including the "admin" password, to the values in the bundle. The host is rebooted immediately. The host needs to be in maintenance mode in order to perform this operation.

-SourcePath <String> Specifies the path to the host configuration backup bundle you want to restore. The bundle is uploaded to an URL address which you can retrieve by using the Get-VMHostFirmware cmdlet.

-Force Indicates that you want to apply the configuration even if the bundle is mismatched. Use this parameter in combination with the Restore parameter.

-HostCredential <PSCredential> Specifies the credential object you want to use for authenticating with the host when uploading a firmware configuration bundle. Do not use this parameter if the HostUser and HostPassword parameters are specified.

-HostUser <String> Specifies a username for authenticating with the host when uploading a firmware configuration bundle.

-HostPassword <SecureString> Specifies a password for the authenticating with the host when uploading a firmware configuration bundle.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VMHost -VMHost Host -State 'Maintenance'

Set-VMHostFirmware -VMHost Host -Restore

Restore the host firmware by using the default path for the firmware bundle. You can store the bundle to the default path through HTTP by using the upload URL specified in the firmware bundle object:

$bundle = Get-VMHostFirmware

$uploadUrl = $bundle.UploadUrl

————— Example 2 —————

C:PS>Set-VMHost -VMHost Host -State ‘Maintenance’

Set-VMHostFirmware -VMHost Host -Restore -SourcePath c:bundleToRestore.tgz -HostUser user -HostPassword pass

Restore the host firmware by specifying a firmware bundle as a source path.

————— Example 3 —————

C:PS>Set-VMHost -VMHost Host -State ‘Maintenance’

Set-VMHostFirmware -VMHost Host -ResetToDefaults

Reset the host configuration to the factory default settings.

————— Example 4 —————

C:PS>Get-VMHostFirmware -VMHost Host1, Host2 -BackupConfiguration -DestinationPath c:StoredBundles

Set-VMHost -VMHost Host1, Host2 -State ‘Maintenance’

Get-VMHost -Name Host1, Host2 | Set-VMHostFirmware -Restore -SourcePath c:StoredBundles -HostUser user -HostPassword pass

Restore multiple hosts firmware by specifying the firmware bundle as a source path directory. The command determines which bundle is needed for each host by the bundle name.

REMARKS To see the examples, type: “get-help Set-VMHostFirmware -examples”. For more information, type: “get-help Set-VMHostFirmware -detailed”. For technical information, type: “get-help Set-VMHostFirmware -full”. For online help, type: “get-help Set-VMHostFirmware -online”

Set-VMHostHba

NAME Set-VMHostHba

SYNOPSIS This cmdlet configures the CHAP properties of the specified iSCSI HBAs.

SYNTAX Set-VMHostHba -IScsiHba <IScsiHba[]> [-IScsiName <String>] [-ChapType <ChapType>] [-ChapName <String>] [-ChapPassword <String>] [-MutualChapEnabled <Boolean>] [-MutualChapName <String>] [-MutualChapPassword <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures the CHAP properties of the specified iSCSI HBAs. If (Mutual)ChapType is set to a value different than “Prohibited”, (Mutual)ChapPassword must be set. ChapType, MutualChapType, MutualChapName, MutualChapPassword - these are only available on 4.1 or later. Note: Run Set-VmHostHba directly against ESX. When Set-VmHostHba is run against vCenter Server, changing the iScsiName property of an iSCSI adapter modifies its AuthenticationCapabilities property.

PARAMETERS

-IScsiHba <IScsiHba[]> Specifies the iSCSI HBA device you want to configure.

-IScsiName <String> Specifies a new name for the host HBA device.

-ChapType <ChapType> Specifies the type of the CHAP authorization. The valid values are Prohibited, Discouraged, Preferred, and Required.

-ChapName <String> Specifies the CHAP initiator name if CHAP is enabled.

-ChapPassword <String> Specifies the CHAP password if CHAP is enabled.

-MutualChapEnabled [<Boolean>] Indicates that Mutual CHAP authorization is enabled.

-MutualChapName <String> Specifies the Mutual CHAP initiator name if Mutual CHAP is enabled.

-MutualChapPassword <String> Specifies the Mutual CHAP password if Mutual CHAP is enabled.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostHba -Type iScsi | Set-VMHostHba -ChapType Required -ChapName Admin -ChapPassword pass

Changes the CHAP type of the available iScsi to Required.

————— Example 2 —————

C:PS>Set-VMHostHba -IScsiHba $iscsi -MutualChapEnabled $true -ChapType Required -ChapName Admin -ChapPassword pass -MutualChapName Administrator -MutualChapPassword Pass

Enables Mutual CHAP for the iScsi devices stored in the $iscsi variable and changes the CHAP type to Required.

REMARKS To see the examples, type: "get-help Set-VMHostHba -examples". For more information, type: "get-help Set-VMHostHba -detailed". For technical information, type: "get-help Set-VMHostHba -full". For online help, type: "get-help Set-VMHostHba -online"

Set-VMHostModule

NAME Set-VMHostModule

SYNOPSIS This cmdlet overrides the host module options with the given ones.

SYNTAX Set-VMHostModule [-HostModule] <VmHostModule[]> [-Options] <String> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet overrides the host module options with the given ones.

PARAMETERS

-HostModule <VmHostModule[]> Specifies the host module you want to configure.

-Options <String> Specifies the new options of the host module.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$module = Get-VMHostModule -Name Shaper

Set-VMHostModule -HostModule $module -Options "New options text"

Overrides the options of the Shaper host module with the provided ones.

————— Example 2 —————

C:PS>Get-VMHostModule Shaper | Set-VMHostModule -Options "New options text" -Confirm

Overrides the options of the Shaper host module with the provided ones.

REMARKS To see the examples, type: "get-help Set-VMHostModule -examples". For more information, type: "get-help Set-VMHostModule -detailed". For technical information, type: "get-help Set-VMHostModule -full". For online help, type: "get-help Set-VMHostModule -online"

Set-VMHostNetwork

NAME Set-VMHostNetwork

SYNOPSIS This cmdlet updates the specified virtual network.

SYNTAX Set-VMHostNetwork [-Network] <VMHostNetworkInfo[]> [-ConsoleGateway <String>] [-VMKernelGateway <String>] [-VMKernelGatewayDevice <String>] [-ConsoleGatewayDevice <String>] [-DomainName <String>] [-HostName <String>] [-DnsFromDhcp <Boolean>] [-DnsDhcpDevice <Object>] [-DnsAddress <String[]>] [-SearchDomain <String[]>] [-IPv6Enabled <Boolean>] [-ConsoleV6Gateway <String>] [-ConsoleV6GatewayDevice <String>] [-VMKernelV6Gateway <String>] [-VMKernelV6GatewayDevice <String>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet updates the specified virtual network. The service console and the VMkernel are often not connected to the same network, and therefore each needs its own gateway information. A gateway is needed for connectivity to machines not on the same IP subnet as the service console or VMkernel.

PARAMETERS

-Network <VMHostNetworkInfo[]> Specifies the host network you want to configure.

-ConsoleGateway <String> Specifies a new console gateway.

-VMKernelGateway <String> Specifies a new kernel gateway.

-VMKernelGatewayDevice <String> Specifies a new kernel gateway device.

-ConsoleGatewayDevice <String> Specifies a new console gateway device.

-DomainName <String> Specifies a new domain name.

-HostName <String> Specifies a new host name.

-DnsFromDhcp [<Boolean>] Indicates that you want to obtain the network settings from a Dhcp server.

-DnsDhcpDevice <Object> This parameter is mandatory if the value of the DnsFromDhcp parameter is 'true'. Otherwise, it is disregarded. If the DnsDhcpDevice parameter is set, the Dhcp DNS of the service console or VMKernel network adapter will override the system DNS. The parameter takes a ServiceConsoleNIC object, a VMKernelNIC object in case of an ESX visor, or the NIC name as a string.

-DnsAddress <String[]> Specifies a new DNS address.

-SearchDomain <String[]> Specifies a new search domain.

-IPv6Enabled [<Boolean>] Indicates that IPv6 configuration is enabled. Setting this parameter to $false disables the ConsoleV6Gateway, ConsoleV6GatewayDevice, and VMKernelV6Gateway parameters. IPv6 is supported only on vCenter 4.1 and ESX 4.1 or later. To use IPv6 on ESX, you must restart the host after enabling IPv6.

-ConsoleV6Gateway <String> Specifies a console V6 gateway address. Not supported on ESXi.

-ConsoleV6GatewayDevice <String> Specifies a console V6 gateway device. Not supported on ESXi.

-VMKernelV6Gateway <String> Specifies a VMKernel V6 gateway address. This parameter is supported only on ESX hosts.

-VMKernelV6GatewayDevice <String> Specifies a VMKernel V6 gateway device. This parameter is supported only on ESX hosts.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmHostNetworkInfo = Get-VmHostNetwork -Host Host

Set-VmHostNetwork -Network $vmHostNetworkInfo -VMKernelGateway 10.23.11.11 -DomainName eng.vmware.com -HostName Host1 -DnsFromDhcp $false

Gets the network configuration of the virtual machine host named Host. Sets the virtual machine kernel gateway, the domain name, the host name, and the Dhcp of the network.

————— Example 2 —————

C:PS>Get-VMHost Host | Get-VMHostNetwork | Set-VMHostNetwork -IPv6Enabled $true

Get-VMHost Host | Restart-VMHost -Force -Confirm:$false

Enables IPv6 support on the Host host and restarts the host.

————— Example 3 —————

C:PS>Get-VMHost Host | Get-VMHostNetwork | Set-VMHostNetwork -ConsoleV6Gateway $ipv6GatewayAddress -ConsoleV6GatewayDevice “vswif0”

Configures the IPv6 console default gateway on the Host host.

————— Example 4 —————

C:PS>Get-VMHost Host | Get-VMHostNetwork | Set-VMHostNetwork -VMKernelV6Gateway $ipv6GatewayAddress

Configures the IPv6 VMKernel default gateway on the Host host.

REMARKS To see the examples, type: “get-help Set-VMHostNetwork -examples”. For more information, type: “get-help Set-VMHostNetwork -detailed”. For technical information, type: “get-help Set-VMHostNetwork -full”. For online help, type: “get-help Set-VMHostNetwork -online”

Set-VMHostNetworkAdapter

NAME Set-VMHostNetworkAdapter

SYNOPSIS This cmdlet configures the specified host network adapter.

SYNTAX Set-VMHostNetworkAdapter -PhysicalNic <PhysicalNic[]> [-Duplex <String>] [-BitRatePerSecMb <Int32>] [-AutoNegotiate] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostNetworkAdapter -VirtualNic <HostVirtualNic[]> [-Dhcp] [-IP <String>] [-SubnetMask <String>] [-Mac <String>] [-Mtu <Int32>] [-VMotionEnabled <Boolean>] [-FaultToleranceLoggingEnabled <Boolean>] [-ManagementTrafficEnabled <Boolean>] [-VsanTrafficEnabled <Boolean>] [-IPv6ThroughDhcp <Boolean>] [-AutomaticIPv6 <Boolean>] [-IPv6 <String[]>] [-IPv6Enabled <Boolean>] [-WhatIf] [-Confirm] [<Common-Parameters>]

Set-VMHostNetworkAdapter -VirtualNic <HostVirtualNic[]> -PortGroup <DistributedPortGroup> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures the specified host network adapter. For a physical NIC, you can change the duplex and the bit rate settings (10, 100, 1000, 10000). For a regular virtual NIC, you can change the IP address and the subnet mask. For a console virtual NIC, you can modify the IP and the subnet mask, or choose DHCP mode.

PARAMETERS

-PhysicalNic <PhysicalNic[]> Specifies the PhysicalNIC objects you want to update.

-Duplex <String> Indicates whether the link is capable of full-duplex. The valid values are Full and Half. You can set this parameter only when updating a PhysicalNIC. Use this parameter only if the AutoNegotiate parameter is not set.

-BitRatePerSecMb <Int32> Specifies the bit rate of the link. Only valid when configuring a physical NIC. Use this parameter only if the AutoNegotiate parameter is not set. Note that updating the speed (BitRatePerSec) of a physical NIC might take some time due to the hardware configuration being performed, and the returned object might still contain the current configuration instead of the updated one.

-AutoNegotiate	Indicates that the host network adapter speed/duplex settings are configured automatically. Use this parameter only if the Duplex and BitRatePerSecMb parameters are not set.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-VirtualNic <HostVirtualNic[]> Provide a list of the host network adapters you want to configure.

-Dhcp	Indicates whether the host network adapter uses a Dhcp server. This parameter works only on ESXi hosts. For VMKernel adapters, Dhcp is supported only on ESX 4.1 and later.

-IP <String> Specifies an IP address for the network adapter using an IPv4 dot notation. If the NIC has no subnet mask previously defined, you must also set the SubnetMask parameter. If the IP parameter is not specified, DHCP mode is enabled. Only valid when configuring a virtual NIC.

-SubnetMask <String> Specifies a subnet mask for the NIC. If the NIC has a subnet mask previously defined, specifying the SubnetMask parameter when configuring the IP address is not mandatory unless you want to modify the mask. Only valid when configuring a virtual NIC.

-Mac <String> Specifies the media access control (MAC) address of the virtual network adapter. Only valid when configuring a virtual NIC.

-Mtu <Int32> Specifies the MTU size.

-VMotionEnabled [<Boolean>] Indicates that you want to use the virtual host/VMKernel network adapter for VMotion.

-FaultToleranceLoggingEnabled [<Boolean>] Indicates that the network adapter is enabled for Fault Tolerance (FT) logging. This parameter is supported only on ESX/vCenter Server 4.1 and later.

-ManagementTrafficEnabled [<Boolean>] Indicates that you want to enable the network adapter for management traffic. This parameter is supported only on ESX/ESXi/vCenter Server 4.1 and later.

-VsanTrafficEnabled [<Boolean>] Specifies whether Virtual SAN traffic is enabled on this network adapter.

-IPv6ThroughDhcp [<Boolean>] Indicates that the IPv6 address is obtained through DHCP.

-AutomaticIPv6 [<Boolean>] Indicates that the IPv6 address is obtained through a router advertisement.

-IPv6 <String[]> Specifies static addresses using the following format: <IPv6>/<subnet_prefix_length> or <IPv6>. If you skip <subnet_prefix_length>, the default value of 64 is used. Specifying a value for IPv6 parameter overrides the current configuration. To clear all configured static IP addresses, pass an empty array to the IPv6 parameter.

-IPv6Enabled [<Boolean>] Indicates that IPv6 configuration is enabled. Setting this parameter to $false disables all IPv6-related parameters. If the value is $true", you need to provide values for at least one of the IPv6ThroughDhcp, AutomaticIPv6, and IPv6 parameters.

-PortGroup <DistributedPortGroup> Specifies a distributed port group to which you want to connect the host network adapter. You can use this parameter only to migrate a virtual network adapter from a standard port group to a distributed port group.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vswitch = New-VirtualSwitch -VMHost 10.23.112.234 -Name VSwitch

$nic = New-VMHostNetworkAdapter -VMHost 10.23.112.234 -PortGroup PortGroup -VirtualSwitch $vswitch -IP 10.23.123.234 -SubnetMask 255.255.254.0

Set-VMHostNetworkAdapter -VirtualNIC $nic -IP 10.23.112.245 -SubnetMask 255.255.255.0 -Mtu 4000

Updates the network adapter IP address, Subnet mask, and MTU size.

———— Example 2 ————

C:PS>Get-VMHost Host | Get-VMHostNetworkAdapter -VMKernel | Set-VMHostNetworkAdapter -VMotionEnabled $true

Enable VMotion on all VMKernel network adapters on the specified host.

———— Example 3 ————

C:PS>Get-VMHostNetworkAdapter | where { $_.PortGroupName -eq "Service Console 1" } | Set-VMHostNetworkAdapter -IPv6Enabled $false

Disables the IPv6 support on a network adapter.

———— Example 4 ————

C:PS>Get-VMHostNetworkAdapter | where { $_.PortGroupName -eq "Service Console 1" } | Set-VMHostNetworkAdapter -IPv6ThroughDhcp $true

Configures a network adapter to obtain IPv6 through DHCP.

———— Example 5 ————

C:PS>Get-VMHostNetworkAdapter | where { $_.PortGroupName -eq "Service Console 1" } | Set-VMHostNetworkAdapter -AutomaticIPv6 $true

Configures a network adapter to obtain IPv6 by a router advertisement.

———— Example 6 ————

C:PS>Get-VMHostNetworkAdapter | where { $_.PortGroupName -eq "Service Console 1" } | Set-VMHostNetworkAdapter -IPv6 $ipv6Address

Changes the IPv6 address of a network adapter.

REMARKS To see the examples, type: "get-help Set-VMHostNetworkAdapter -examples". For more information, type: "get-help Set-VMHostNetworkAdapter -detailed". For technical information, type: "get-help Set-VMHostNetworkAdapter -full". For online help, type: "get-help Set-VMHostNetworkAdapter -online"

Set-VMHostProfile

NAME Set-VMHostProfile

SYNOPSIS This cmdlet modifies the specified host profile.

SYNTAX Set-VMHostProfile [[-Name] <String>] [[-ReferenceHost] <VMHost>] [-Profile] <VMHostProfile[]> [-Description <String>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the specified host profile.

PARAMETERS

-Name <String> Specifies a new name for the host profile.

-ReferenceHost <VMHost> Specifies a reference host for the host profile.

-Profile <VMHostProfile[]> Specifies the host profile you want to modify.

-Description <String> Specifies a new description for the host profile.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$profile = (Get-VMHostProfile -Name Profile)[0]

Set-VMHostProfile -Profile $profile -Description "New description."

Changes the description of the Profile host profile.

REMARKS To see the examples, type: "get-help Set-VMHostProfile -examples". For more information, type: "get-help Set-VMHostProfile -detailed". For technical information, type: "get-help Set-VMHostProfile -full". For online help, type: "get-help Set-VMHostProfile -online"

Set-VMHostRoute

NAME Set-VMHostRoute

SYNOPSIS This cmdlet modifies a route in the host routing table.

SYNTAX Set-VMHostRoute [-VMHostRoute] <VMHostRoute[]> [-Destination <IPAddress>] [-Gateway <IPAddress>] [-PrefixLength <Int32>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies a route in the host routing table.

PARAMETERS

-VMHostRoute <VMHostRoute[]> Specifies the route you want to modify.

-Destination <IPAddress> Changes the destination IP address of the route.

-Gateway <IPAddress> Changes the gateway IP address of the route.

-PrefixLength <Int32> Modifies the prefix length of the destination IP address. For IPv4, the valid values are from 0 to 32, and for IPv6 - from 0 to 128.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vmhostroute = New-VMHostRoute -VMHost 10.23.114.189 -Destination 192.168.104.101 -Gateway 10.23.84.69 -PrefixLength 32

$vmhostroute | Set-VMHostRoute -Gateway 10.23.84.70

Creates a new host route and modifies its gateway.

———— Example 2 ————

C:PS>$vmhostroute1 = New-VMHostRoute -VMHost 10.23.114.189 -Destination 192.168.104.101 -Gateway 10.23.84.69 -PrefixLength 32

$vmhostroute2 = New-VMHostRoute -VMHost 10.23.114.190 -Destination 192.168.104.101 -Gateway 10.23.84.70 -PrefixLength 32

Set-VMHostRoute -VMHostRoute ($vmhostroute1, $vmhostroute2) -Destination 192.168.104.0 -PrefixLength 24

Modifies the destination and the prefix length of two host routes.

REMARKS To see the examples, type: "get-help Set-VMHostRoute -examples". For more information, type: "get-help Set-VMHostRoute -detailed". For technical information, type: "get-help Set-VMHostRoute -full". For online help, type: "get-help Set-VMHostRoute -online"

Set-VMHostService

NAME Set-VMHostService

SYNOPSIS This cmdlet modifies a host service.

SYNTAX Set-VMHostService [-HostService] <HostService[]> [-Policy] <HostServicePolicy> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies a host service.

PARAMETERS

-HostService <HostService[]> Specifies the host service you want to update.

-Policy <HostServicePolicy> Specifies an activation policy for the host service.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VMHostService -VMHost Host | Set-VMHostService -Policy "Automatic"

Sets the policy of all services the Host virtual machine host to "Automatic".

REMARKS To see the examples, type: "get-help Set-VMHostService -examples". For more information, type: "get-help Set-VMHostService -detailed". For technical information, type: "get-help Set-VMHostService -full". For online help, type: "get-help Set-VMHostService -online"

Set-VMHostSnmp

NAME Set-VMHostSnmp

SYNOPSIS This cmdlet modifies the host SNMP configuration.

SYNTAX Set-VMHostSnmp [-HostSnmp] <VmHostSnmp[]> [-Enabled <Boolean>] [-Port <Int32>] [-ReadOnlyCommunity <String[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostSnmp [-HostSnmp] <VmHostSnmp[]> [-Enabled <Boolean>] [-Port <Int32>] [-ReadOnlyCommunity <String[]>] -TargetCommunity <String> [-TargetPort <Int32>] -TargetHost <String> -AddTarget [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostSnmp [-HostSnmp] <VmHostSnmp[]> [-Enabled <Boolean>] [-Port <Int32>] [-ReadOnlyCommunity <String[]>] [-TargetCommunity <String>] [-TargetPort <Int32>] [-TargetHost <String>] -RemoveTarget [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMHostSnmp [-HostSnmp] <VmHostSnmp[]> [-Enabled <Boolean>] [-Port <Int32>] [-ReadOnlyCommunity <String[]>] [-RemoveTarget] -TrapTargetToRemove <TrapTarget> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the host SNMP configuration. If specified, adds or removes a trap target (removing can be specified by either TrapTargetToRemove parameter or by any of the following parameters (or combination of them): TargetCommunity, TargetHost, TargetPort). If the user passes $null, an empty array or string to the ReadOnlyCommunities parameter, the old values of this property are erased. This results in a NULL value of this property of the output object.

PARAMETERS

-HostSnmp <VmHostSnmp[]> Specifies the host Snmp object you want to modify.

-Enabled [<Boolean>] Indicates that the SNMP feature is enabled on the specified host.

-Port <Int32> Specifies the port on which the host listens to SNMP messages.

-ReadOnlyCommunity <String[]> Provide a list of communities, identifying who is able to send SNMP requests to that host. If $null, an empty array or string are passed to this parameter, its old values are erased and the output object for the ReadOnlyCommunity property is an empty array. In PowerShell an empty array is defined by @().

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-TargetCommunity <String> Specifies the community identifier of the trap target.

-TargetPort <Int32> Specifies the port on which the target host listens to SNMP messages.

-TargetHost <String> Specifies the identifier of the target host - a host name or an IP address.

-AddTarget	Indicates that you want to add a new trap target to the host SNMP configuration. A trap target consists of three elements - Community (mandatory), HostName (mandatory), Port (optional - defaults to 162), specified by the TargetCommunity, TargetHost, and TargetPort parameters.
-RemoveTarget	Indicates that you want to remove a trap target from the host SNMP configuration. There are two ways to specify a trap target: * Pass the trap target to the TrapTargetToRemove parameter. * Use a combination of the TargetCommunity, TargetHost, and TargetPort parameters to specify a criteria (for example, remove all trap targets that are using port 162).

-TrapTargetToRemove <TrapTarget> Specifies the trap target you want to remove. The trap target can be obtained from the "TrapTargets" property of the HostSNMP object (an array of TrapTarget objects).

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmhostSNMP = Get-VMHostSNMP

Set-VMHostSNMP $vmhostSNMP -Enabled:$true -ReadOnlyCommunity 'example-community'

Enables SNMP on a virtual machine host.

————— Example 2 —————

C:PS>Get-VMHostSnmp | Set-VMHostSnmp -ReadonlyCommunity @()

Sets the virtual machine host SNMP by erasing the old value of the ReadOnlyCommunity parameter.

REMARKS To see the examples, type: "get-help Set-VMHostSnmp -examples". For more information, type: "get-help Set-VMHostSnmp -detailed". For technical information, type: "get-help Set-VMHostSnmp -full". For online help, type: "get-help Set-VMHostSnmp -online"

Set-VMHostStartPolicy

NAME Set-VMHostStartPolicy

SYNOPSIS This cmdlet modifies the host default start policy.

SYNTAX Set-VMHostStartPolicy [-VMHostStartPolicy] <VMHostStartPolicy[]> [-Enabled <Boolean>] [-StartDelay <Int32>] [-StopAction <VmStopAction>] [-StopDelay <Int32>] [-WaitForHeartBeat <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the host default start policy. Start policy defines what happens to virtual machines when the server starts up or stops.

PARAMETERS

-VMHostStartPolicy <VMHostStartPolicy[]> Specifies the host start policy you want to modify.

-Enabled [<Boolean>] Indicates that the service that controls the host start policies is enabled. If it is enabled, the default start policies and the start policies of the specified hosts are applied. If disabled, no start policy is applied.

-StartDelay <Int32> Specifies a default start delay of the virtual machines in seconds.

-StopAction <VmStopAction> Specifies the default action that is applied to the virtual machines when the server stops. The valid values are None, Suspend, PowerOff, or GuestShutDown.

-StopDelay <Int32> Specifies a default stop delay of the virtual machines in seconds.

-WaitForHeartBeat [<Boolean>] Specifies whether the virtual machines should start after receiving a heartbeat from the host, ignore heartbeats, and start after the StartDelay has elapsed ($true), or follow the system default before powering on ($false). When a virtual machine is next in the start order, the system either waits a specified period of time for a host to power on or it waits until it receives a successful heartbeat from a powered-on host.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

---------- Example 1 ----------

C:PS>Get-VMHost Host | Get-VMHostStartPolicy | Set-VMHostStartPolicy -Enabled:$true -StartOrder 2 -StartDelay 300 -StopAction GuestShutDown -StopDelay 300

Retrieves the start policy of the Host host and modifies its configuration settings.

---------- Example 2 ----------

C:PS>Get-VMHost Host | Get-VMHostStartPolicy | Set-VMHostStartPolicy -WaitForHeartbeat

Retrieves the start policy of the Host host and modifies its configuration settings, so that virtual machines on the specified host wait for the host heartbeat.

REMARKS To see the examples, type: "get-help Set-VMHostStartPolicy -examples". For more information, type: "get-help Set-VMHostStartPolicy -detailed". For technical information, type: "get-help Set-VMHostStartPolicy -full". For online help, type: "get-help Set-VMHostStartPolicy -online"

Set-VMHostStorage

NAME Set-VMHostStorage

SYNOPSIS This cmdlet configures a host storage.

SYNTAX Set-VMHostStorage -VMHostStorage <VMHostStorageInfo[]> -SoftwareIScsiEnabled <Boolean> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures a host storage. The cmdlet enables or disables the software iSCSI support for the specified VMHostStorage objects.

PARAMETERS

-VMHostStorage <VMHostStorageInfo[]> Specifies the host storage you want to configure.

-SoftwareIScsiEnabled [<Boolean>] Indicates that on this storage, software iSCSI is enabled.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHostStorage 10.23.112.234 | Set-VMHostStorage -SoftwareIScsiEnabled $true

Enables the iSCSI on the specified storage.

REMARKS To see the examples, type: "get-help Set-VMHostStorage -examples". For more information, type: "get-help Set-VMHostStorage -detailed". For technical information, type: "get-help Set-VMHostStorage -full". For online help, type: "get-help Set-VMHostStorage -online"

Set-VMHostSysLogServer

NAME Set-VMHostSysLogServer

SYNOPSIS This cmdlet configures the remote syslog server of the specified hosts.

SYNTAX Set-VMHostSysLogServer [[-SysLogServer] <NamedIPEndPoint[]>] [-VMHost] <VMHost[]> [-SysLogServerPort <Int32>] [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures the remote syslog server of the specified hosts.

PARAMETERS

-SysLogServer <NamedIPEndPoint[]> Specifies the sys log servers you want to configure. The parameter accepts objects of the NamedIPEndPoint, IPEndPoint, IPAddress, and String types. The accepted formats, if string is used, are DNS names and the standard IPv6/IPv4 format: FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:192.168.0.1:80, where the FFFF:FFFF:FFFF:FFFF:FFFF:FFFF can be replaced by any hex value with the same structure (both upper or lower case). It is only meaningful in case of IPv6 address and is omitted for IPv4. The 192.168.0.1 part is mandatory and can be any address in the IPv4 format. The :80 part is optional. If omitted, the port must be specified through the SysLogServerPort parameter. If Syslog is set to $null, the configured syslog server, if any, is removed.

-VMHost <VMHost[]> Specifies the host whose syslog servers you want to configure.

-SysLogServerPort <Int32> Specifies the sys log server port. Must be specified if the string that is passed to the SysLogServer parameter does not contain the port value, or the argument of the SysLogServer is an IP address.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VMHostSysLogServer -SysLogServer '192.168.0.1:133' -VMHost Host

Sets a SysLog server on the Host virtual machine host.

————— Example 2 —————

C:PS>Set-VMHostSysLogServer -SysLogServer $null -VMHost Host

Removes the SysLog server from the Host virtual machine host.

REMARKS To see the examples, type: "get-help Set-VMHostSysLogServer -examples". For more information, type: "get-help Set-VMHostSysLogServer -detailed". For technical information, type: "get-help Set-VMHostSysLogServer -full". For online help, type: "get-help Set-VMHostSysLogServer -online"

Set-VMQuestion

NAME Set-VMQuestion

SYNOPSIS This cmdlet answers the specified virtual machine question.

SYNTAX Set-VMQuestion -VMQuestion <VMQuestion[]> [-Option] <Object> [-WhatIf] [-Confirm] [<CommonParameters>]

Set-VMQuestion -VMQuestion <VMQuestion[]> -DefaultOption [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet answers the specified virtual machine question using the value of the Option parameter. If the DefaultOption parameter is set to $true, the cmdlet answers the question with a default option, if any.

PARAMETERS

-VMQuestion <VMQuestion[]> Specifies the virtual machine question you want to answer.

-Option <Object> Specifies an object or string to provide an option to the virtual machine question. Wildcards are supported for string values. The string can be used to specify an option ID or label. If the string does not match a valid option ID or label, or if there are multiple matches, an error is generated.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-DefaultOption Indicates that you want to answer the virtual machine question using a default option. If no default option exists for the question, an error is generated.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VMQuestion -VMQuestion $question -DefaultOption

Answers the question stored in the $question with a default option.

————— Example 2 —————

C:PS>Set-VMQuestion -VMQuestion $question -Option “Cancel”

Answers the question stored in the $question variable with “cancel”.

————— Example 3 —————

C:PS>Get-VM VM | Get-VMQuestion | Set-VMQuestion -DefaultOption

Answers the question of VM virtual machine with a default option.

REMARKS To see the examples, type: “get-help Set-VMQuestion -examples”. For more information, type: “get-help Set-VMQuestion -detailed”. For technical information, type: “get-help Set-VMQuestion -full”. For online help, type: “get-help Set-VMQuestion -online”

Set-VMResourceConfiguration

NAME Set-VMResourceConfiguration

SYNOPSIS This cmdlet configures resource allocation between the virtual machines.

SYNTAX Set-VMResourceConfiguration [-Configuration] <VMResourceConfiguration[]> [-HtCoreSharing <HTCoreSharing>] [-CpuAffinity <CpuAffinity>] [-CpuAffinityList <Int32[]>] [-CpuReservationMhz <Int64>] [-CpuLimitMhz <Int64>] [-CpuSharesLevel <SharesLevel>] [-NumCpuShares <Int32>] [-MemReservationMB <Int64>] [-MemReservationGB <Decimal>] [-MemLimitMB <Int64>] [-MemLimitGB <Decimal>] [-MemSharesLevel <SharesLevel>] [-NumMemShares <Int32>] [-Disk <HardDisk[]>] [-NumDiskShares <Int32>] [-DiskSharesLevel <SharesLevel>] [-DiskLimitIOPerSecond <Int64>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet configures resource allocation between the virtual machines. To retain the current value of a setting, omit the corresponding parameter. To disable a setting (only applicable to the nullable limit settings), pass $null.

PARAMETERS

-Configuration <VMResourceConfiguration[]> Specifies the configuration object you want to modify.

-HtCoreSharing <HTCoreSharing> Specifies whether a virtual machine is scheduled to share a physical processor core (assuming hyperthreading is enabled on the host at all). The following values are valid:

Any - (default) the virtual CPUs of this virtual machine can freely share cores with other virtual CPUs of this or other virtual machines.

None - the virtual CPUs of this virtual machine have exclusive use of a processor core whenever they are scheduled to it. The other hyperthread of the core is "halted" while this virtual machine is using the core.

Internal - on a virtual machine with exactly two virtual processors, the two virtual processors are allowed to share one physical core (at the discretion of the ESX scheduler), but this virtual machine never shares a core with any other virtual machine. If this virtual machine has any other number of processors than two, this setting is the same as the None setting.

-CpuAffinity <CpuAffinity> The use of this parameter is deprecated. Use CpuAffinityList instead.

Specifies the distribution of virtual machine CPUs across the physical cores or hyperthreads of the host. You must pass exactly as many arguments as the number of CPUs of the virtual machine. Each argument specifies the physical core or hyperthread that the virtual machine will use. Valid arguments are NoAffinity, Cpu1, . . . , Cpu63.

When the virtual machine resides in a DRS cluster, you cannot use CpuAffinity.

-CpuAffinityList <Int32[]> Specifies the distribution of virtual machine CPUs across the physical cores or hyperthreads of the host. You must pass exactly as many arguments as the number of CPUs of the virtual machine. Each argument specifies the physical core or hyperthread that the virtual machine will use. Valid arguments are positive integers. To clear formerly specified arguments, pass an empty array.

When the virtual machine resides in a DRS cluster, you cannot use CpuAffinityList.

-CpuReservationMhz <Int64> Specifies the number of CPU MHz that are guaranteed to be available.

-CpuLimitMhz <Int64> Specifies the limit on CPU usage in MHz. Utilization will not exceed this limit even if there are available resources.

-CpuSharesLevel <SharesLevel> Specifies the CPU allocation level. Used in relative allocation between virtual machines. The valid values are Custom, High, Low, and Normal.

-NumCpuShares <Int32> Specifies the CPU allocation level for this pool. Used in relative allocation between resource consumers. This parameter is ignored unless CpuSharesLevel is set to Custom.

-MemReservationMB <Int64> This parameter is obsolete. Use MemReservationGB instead. Specifies the guaranteed available memory in megabytes (MB).

-MemReservationGB <Decimal> Specifies the guaranteed available memory in gigabytes (GB).

-MemLimitMB <Int64> This parameter is obsolete. Use MemLimitGB instead. Specifies a memory usage limit in megabytes (MB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemLimitGB <Decimal> Specifies a memory usage limit in gigabytes (GB). If this parameter is set, utilization will not exceed the specified limit even if there are available resources.

-MemSharesLevel <SharesLevel> Specifies the memory allocation level for this pool. Used in relative allocation between resource consumers. The valid values are Custom, High, Low, and Normal.

-NumMemShares <Int32> Specifies the number of memory shares allocated. Used to determine resource allocation in case of resource contention.

-Disk <HardDisk[]> Specifies the virtual hard disk you want to configure.

-NumDiskShares <Int32> Specifies the number of shares allocated. Used to determine resource allocation in case of resource contention.

-DiskSharesLevel <SharesLevel> Specifies the allocation level. The level is a simplified view of shares. Levels map to a pre-determined set of numeric values for shares. If the shares value does not map to a predefined size, then the level is set as custom.

-DiskLimitIOPerSecond <Int64> Specifies the disk limit IO per second. The valid values are in the range between 16 and 2147483647. -1 means unlimited.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Set-VMResourceConfiguration -Configuration $conf -CpuAffinity ([CpuAffinity]::Cpu1 -bor [CpuAffinity]::Cpu2)

Specifies two affinities for the virtual machine resource configuration in the $conf variable. Bit Or is used.

REMARKS To see the examples, type: "get-help Set-VMResourceConfiguration -examples". For more information, type: "get-help Set-VMResourceConfiguration -detailed". For technical information, type: "get-help Set-VMResourceConfiguration -full". For online help, type: "get-help Set-VMResourceConfiguration -online"

Set-VMStartPolicy

NAME Set-VMStartPolicy

SYNOPSIS This cmdlet modifies the virtual machine start policy.

SYNTAX Set-VMStartPolicy [-StartPolicy] <VMStartPolicy[]> [-StartAction <VmStartAction>] [-StartOrder <Int32>] [-InheritStopActionFromHost] [-InheritStopDelayFromHost] [-InheritWaitForHeartbeatFromHost] [-InheritStartDelayFromHost] [-UnspecifiedStartOrder] [-StartDelay <Int32>] [-StopAction <VmStopAction>] [-StopDelay <Int32>] [-WaitForHeartBeat <Boolean>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet modifies the virtual machine start policy. Start policy defines what happens to virtual machines when the server starts up or stops.

PARAMETERS

-StartPolicy <VMStartPolicy[]> Specifies the virtual machine start policy you want to modify.

-StartAction <VmStartAction> Specifies a start action for virtual machines. It can be None or PowerOn.

-StartOrder <Int32> Specifies a number to define the virtual machines start order.

-InheritStopActionFromHost Indicates that the virtual machine uses the value of the StopAction parameter of the host.

-InheritStopDelayFromHost Indicates that the virtual machine uses the value of the StopDelay parameter of the host.

-InheritWaitForHeartbeatFromHost Indicates that the virtual machine uses the value of the WaitforHeartbeat parameter of the host.

-InheritStartDelayFromHost Indicates that the virtual machine uses the value of the StartDelay parameter of the host.

-UnspecifiedStartOrder Indicates that no order is defined for starting the virtual machines.

-StartDelay <Int32> Specifies a default start delay in seconds.

-StopAction <VmStopAction> Specifies the default action of the virtual machine when the server stops. The valid values are None, Suspend, PowerOff, and GuestShutDown.

-StopDelay <Int32> Specifies the default stop delay in seconds.

-WaitForHeartBeat [<Boolean>] Indicates whether the virtual machine should start after receiving a heartbeat, ignore heartbeats and start after the StartDelay has elapsed ($true), or follow the system default before powering on ($false). When a virtual machine is next in the start order, the system either waits a specified period of time for a virtual machine to power on or it waits until it receives a successful heartbeat from a powered on virtual machine.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vmstartpolicy = Get-VMStartPolicy -VM VM

Set-VMStartPolicy -StartPolicy $vmstartpolicy -StartAction PowerOn

Retrieves the start policy of the VM virtual machine and defines that when the server starts, the virtual machine is powered on.

————— Example 2 —————

C:PS>Get-VM VM | Get-VMStartPolicy | Set-VMStartpolicy -InheritStopActionFromHost -InheritStopDelayFromHost

Reconfigures the start policy of the VM virtual machine to inherit the values of the StopAction and StopDelay from the host.

————— Example 3 —————

C:PS>Get-VM VM | Get-VMStartPolicy | Set-VMStartpolicy -StartAction PowerOn -StartOrder 2 -StartDelay 300 -StopAction GuestShutDown -StopDelay 300

Retrieve the start policy of the specified virtual machine and modify its configuration settings.

REMARKS To see the examples, type: "get-help Set-VMStartPolicy -examples". For more information, type: "get-help Set-VMStartPolicy -detailed". For technical information, type: "get-help Set-VMStartPolicy -full". For online help, type: "get-help Set-VMStartPolicy -online"

CHAPTER 26

Start Commands

This page contains details on **Start** commands.

Start-VApp

NAME Start-VApp

SYNOPSIS This cmdlet starts vApps.

SYNTAX Start-VApp [-VApp] <VApp[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet starts vApps.

PARAMETERS

-VApp <VApp[]> Specifies the vApp that you want to start.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost MyVMHost1 | Get-VApp | Start-VApp

Starts all vApps on the specified host.

REMARKS To see the examples, type: "get-help Start-VApp -examples". For more information, type: "get-help Start-VApp -detailed". For technical information, type: "get-help Start-VApp -full". For online help, type: "get-help Start-VApp -online"

Start-VM

NAME Start-VM

SYNOPSIS This cmdlet powers on virtual machines.

SYNTAX Start-VM [-RunAsync] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet powers on the virtual machines specified by the VM parameter.

PARAMETERS

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-VM <VirtualMachine[]> Specifies the virtual machines you want to power on.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Start-VM -VM VM -Confirm -RunAsync

Asynchronously starts the virtual machine named VM. Before initializing the task, asks for confirmation.

REMARKS To see the examples, type: "get-help Start-VM -examples". For more information, type: "get-help Start-VM -detailed". For technical information, type: "get-help Start-VM -full". For online help, type: "get-help Start-VM -online"

Start-VMHost

NAME Start-VMHost

SYNOPSIS This cmdlet starts the specified hosts.

SYNTAX Start-VMHost [-VMHost] <VMHost[]> [-TimeoutSeconds <Int32>] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet starts the specified hosts. The task completes when the host successfully exits standby state and sends a heartbeat signal. If nothing is received from the host for the time defined by the TimeoutSeconds parameter, the host is declared timed out, and the task is assumed failed.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to start.

-TimeoutSeconds <Int32> Specifies a time period in seconds to wait for a heartbeat signal from the host. If nothing is received from the host for the specified time, the host is declared timed out, and the task is assumed failed. The default value is 300.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Start-VMHost 10.23.112.235 -RunAsync

Starts the specified host. The command returns without waiting for the task to complete.

REMARKS To see the examples, type: “get-help Start-VMHost -examples”. For more information, type: “get-help Start-VMHost -detailed”. For technical information, type: “get-help Start-VMHost -full”. For online help, type: “get-help Start-VMHost -online”

Start-VMHostService

NAME Start-VMHostService

SYNOPSIS This cmdlet starts the specified host services.

SYNTAX Start-VMHostService [-HostService] <HostService[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet starts the specified host services.

PARAMETERS

-HostService <HostService[]> Specifies the host services you want to start.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Start-VMHostService -Service $vmHostService

Starts a host service.

REMARKS To see the examples, type: "get-help Start-VMHostService -examples". For more information, type: "get-help Start-VMHostService -detailed". For technical information, type: "get-help Start-VMHostService -full". For online help, type: "get-help Start-VMHostService -online"

CHAPTER 27

Stop Commands

This page contains details on **Stop** commands.

Stop-Task

NAME Stop-Task

SYNOPSIS This cmdlet stops the specified tasks.

SYNTAX Stop-Task [-Task] <Task[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet stops the tasks specified by the Task parameter.

PARAMETERS

-Task <Task[]> Specifies the tasks you want to stop.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$vm = Get-VM -Name “VM1”

$datacenter = Get-Datacenter -Name “Datacenter1”

$task = New-Template -Name “Template1” -Location $datacenter -VM $vm -RunAsync

Stop-Task -Task $task

Stops the process of creating a new template from a virtual machine.

REMARKS To see the examples, type: “get-help Stop-Task -examples”. For more information, type: “get-help Stop-Task -detailed”. For technical information, type: “get-help Stop-Task -full”. For online help, type: “get-help Stop-Task -online”

Stop-VApp

NAME Stop-VApp

SYNOPSIS This cmdlet stops vApps.

SYNTAX Stop-VApp [-Force] [-VApp] <VApp[]> [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet stops vApps.

PARAMETERS

-Force Indicates that the virtual machines are powered off regardless of the auto-start configuration of the vApps.

-VApp <VApp[]> Specifies the vApp that you want to stop.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Get-VMHost MyVMHost1 | Get-VApp | Stop-VApp

Stops all virtual appliances on the specified host.

REMARKS To see the examples, type: “get-help Stop-VApp -examples”. For more information, type: “get-help Stop-VApp -detailed”. For technical information, type: “get-help Stop-VApp -full”. For online help, type: “get-help Stop-VApp -online”

Stop-VM

NAME Stop-VM

SYNOPSIS This cmdlet powers off virtual machines.

SYNTAX Stop-VM [-Kill] [-RunAsync] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet stops the virtual machines specified by the VM parameter.

PARAMETERS

-Kill	Indicates that you want to stop the specified virtual machines by terminating their processes running on the ESX. You can use this parameter to stop a virtual machine that is not responding and cannot be stopped or restarted in other ways. To use the Kill parameter, you need to have a direct connection to ESX 4.1 or later.
-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-VM <VirtualMachine[]> Specifies the virtual machines you want to power off.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Stop-VM -VM VM -Confirm

Stops the virtual machine named VM after confirmation by the user.

————— Example 2 —————

C:PS>Stop-VM -VM VM -Kill -Confirm:$false

Stops the virtual machine VM by terminating its process running on the ESX.

REMARKS To see the examples, type: "get-help Stop-VM -examples". For more information, type: "get-help Stop-VM -detailed". For technical information, type: "get-help Stop-VM -full". For online help, type: "get-help Stop-VM -online"

Stop-VMGuest

NAME Stop-VMGuest

SYNOPSIS This cmdlet shuts down the specified virtual machine guest OS.

SYNTAX Stop-VMGuest [[-VM] <VirtualMachine[]>] [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Stop-VMGuest [[-Guest] <VMGuest[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet issues a command to the guest operating system asking it to prepare for a shutdown operation. Returns immediately and does not wait for the guest operating system to complete the operation.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines whose operating systems you want to shut down. The virtual machines must have VMware Tools installed.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Guest <VMGuest[]> Specifies the virtual machine guests you want to shut down.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VM VM | Stop-VMGuest

Shuts down the guest OS of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Stop-VMGuest -examples". For more information, type: "get-help Stop-VMGuest -detailed". For technical information, type: "get-help Stop-VMGuest -full". For online help, type: "get-help Stop-VMGuest -online"

Stop-VMHost

NAME Stop-VMHost

SYNOPSIS This cmdlet powers off the specified hosts.

SYNTAX Stop-VMHost [-VMHost] <VMHost[]> [-Force] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet powers off the specified hosts. When the cmdlet runs asynchronously (with the RunAsync parameter) and you are connected directly to the host, the returned task object contains no indicator of success.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to power off.

-Force	Indicates that you want to stop the hosts even if they are not in a maintenance mode.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Stop-VMHost 10.23.112.235 -Confirm

Shutdowns the specified host after user confirmation.

REMARKS To see the examples, type: "get-help Stop-VMHost -examples". For more information, type: "get-help Stop-VMHost -detailed". For technical information, type: "get-help Stop-VMHost -full". For online help, type: "get-help Stop-VMHost -online"

Stop-VMHostService

NAME Stop-VMHostService

SYNOPSIS This cmdlet stops the specified host services.

SYNTAX Stop-VMHostService [-HostService] <HostService[]> [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet stops the host service specified by the Service parameter.

PARAMETERS

-HostService <HostService[]> Specifies the host services you want to stop.

-WhatIf Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.

-Confirm If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Start-VMHostService -Service $vmHostService

Stops a host service.

REMARKS To see the examples, type: "get-help Stop-VMHostService -examples". For more information, type: "get-help Stop-VMHostService -detailed". For technical information, type: "get-help Stop-VMHostService -full". For online help, type: "get-help Stop-VMHostService -online"

CHAPTER 28

Suspend Commands

This page contains details on **Suspend** commands.

Suspend-VM

NAME Suspend-VM

SYNOPSIS This cmdlet suspends virtual machines.

SYNTAX Suspend-VM [-RunAsync] [-VM] <VirtualMachine[]> [-Server <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet suspends the virtual machines specified by the VM parameter. You can use the suspend feature to make resources available on a short-term basis or for other situations in which you want to put a virtual machine on hold without powering it down. Using wildcards is supported with virtual machine names.

PARAMETERS

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.

-VM <VirtualMachine[]> Specifies the virtual machines you want to suspend.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VM VM | Suspend-VM

Suspends the virtual machine named VM.

REMARKS To see the examples, type: "get-help Suspend-VM -examples". For more information, type: "get-help Suspend-VM -detailed". For technical information, type: "get-help Suspend-VM -full". For online help, type: "get-help Suspend-VM -online"

Suspend-VMGuest

NAME Suspend-VMGuest

SYNOPSIS This cmdlet suspends the specified guest operating systems.

SYNTAX Suspend-VMGuest [[-VM] <VirtualMachine[]>] [[-Server] <VIServer[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

Suspend-VMGuest [[-Guest] <VMGuest[]>] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet issues a command to the specified guest operating system asking it to prepare for a suspend operation.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines whose operating systems you want to suspend. The virtual machines must have VMware Tools installed.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

-Guest <VMGuest[]> Specifies the guest operating systems you want to suspend.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Get-VM VM| Suspend-VMGuest

Suspends the guest OS of the virtual machine named VM.

REMARKS To see the examples, type: "get-help Suspend-VMGuest -examples". For more information, type: "get-help Suspend-VMGuest -detailed". For technical information, type: "get-help Suspend-VMGuest -full". For online help, type: "get-help Suspend-VMGuest -online"

Suspend-VMHost

NAME Suspend-VMHost

SYNOPSIS This cmdlet suspends hosts.

SYNTAX Suspend-VMHost [-VMHost] <VMHost[]> [-TimeoutSeconds <Int32>] [-Evacuate] [-Server <VIServer[]>] [-RunAsync] [-WhatIf] [-Confirm] [<CommonParameters>]

DESCRIPTION This cmdlet puts the specified host machines in standby mode. You can use the suspend feature to make resources available on a short-term basis or for other situations in which you want to put a host on hold without powering it off.

PARAMETERS

-VMHost <VMHost[]> Specifies the hosts you want to suspend.

-TimeoutSeconds <Int32> Specifies a time period in seconds to wait for the host to enter standby mode. If the host is not suspended for the specified time, the host is declared timed out, and the task is assumed failed. The default value is 300.

-Evacuate	If the value is $true, vCenter Server automatically reregisters the virtual machines that are compatible for reregistration. If they are not compatible, they remain on the suspended host. If there are powered-on virtual machines that cannot be reregistered, the operation waits until they are powered off manually. The Evacuate parameter is valid only when connected to a vCenter Server system and the virtual machine host is part of a DRS-enabled cluster.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-RunAsync	Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run “help About_RunAsync” in the vSphere PowerCLI console.
-WhatIf	Indicates that the cmdlet is run only to display the changes that would be made and actually no objects are modified.
-Confirm	If the value is $true, indicates that the cmdlet asks for confirmation before running. If the value is $false, the cmdlet runs without asking for user confirmation.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>Suspend-VMHost 10.23.112.54 -TimeOutSeconds 60 -Confirm

Suspends the specified host after user confirmation. If the host is not suspended within 60 seconds, the task is reported failed.

REMARKS To see the examples, type: “get-help Suspend-VMHost -examples”. For more information, type: “get-help Suspend-VMHost -detailed”. For technical information, type: “get-help Suspend-VMHost -full”. For online help, type: “get-help Suspend-VMHost -online”

CHAPTER 29

Test Commands

This page contains details on **Test** commands.

Test-VMHostProfileCompliance

NAME Test-VMHostProfileCompliance

SYNOPSIS This cmdlet tests hosts for profile compliance.

SYNTAX Test-VMHostProfileCompliance [-VMHost] <VMHost[]> [-UseCache] [[-Server] <VIServer[]>] [<CommonParameters>]

Test-VMHostProfileCompliance [-Profile] <VMHostProfile[]> [-UseCache] [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet tests hosts for profile compliance. The Profile and VMHost parameters cannot be set at the same time. If the Profile parameter is set, the specified host profile is tested for compliance with the hosts, to which it is associated. If the VMHost parameter is specified, the host is tested for compliance with the profiles associated with it. If no profiles are associated with the host, then the profile associated with the cluster is applied.

PARAMETERS

-VMHost <VMHost[]> Specifies the host you want to test for profile compliance with the profile associated with it. If no profile is associated with it, the host is tested for compliance with the profile associated with the cluster, to which the host belongs. Do not set this parameter if the Profile parameter is set.

-UseCache	Indicates that you want the vCenter Server to return cached information. If vCenter Server does not have cached information, a compliance scanning is performed.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Profile <VMHostProfile[]> Specifies a host profile against which to test the specified host for compliance with the host to which it is associated. Do not set this parameter if the VMHost parameter is set.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 2 —————

C:PS>Test-VMHostProfileCompliance -VMHost Host

Tests the specified host for compliance with the profiles associated with it.

————— Example 3 —————

C:PS>$profile = Get-VMHostProfile -Name Profile

Apply-VMHostProfile -AssociateOnly -Profile $profile -Entity 10.0.0.126

Test-VMHostProfileCompliance -VMHost 10.0.0.126 | fl *

Test the profile compliance of a non-compliant virtual machine host associated with the profile.

————— Example 4 —————

C:PS>Test-VMHostProfileCompliance -Profile $profile | fl *

Test the profile compliance of a virtual machine host profile with the hosts it is associated with.

REMARKS To see the examples, type: "get-help Test-VMHostProfileCompliance -examples". For more information, type: "get-help Test-VMHostProfileCompliance -detailed". For technical information, type: "get-help Test-VMHostProfileCompliance -full". For online help, type: "get-help Test-VMHostProfileCompliance -online"

Test-VMHostSnmp

NAME Test-VMHostSnmp

SYNOPSIS This cmdlet tests the host SNMP.

SYNTAX Test-VMHostSnmp [-HostSnmp] <VmHostSnmp[]> [<CommonParameters>]

DESCRIPTION This cmdlet tests the host SNMP specified by the HostSNMP parameter.

PARAMETERS

-HostSnmp <VmHostSnmp[]> Specifies the host SNMP you want to test.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Test-VMHostSNMP -HostSNMP (Get-VMHostSNMP)

Retrieves and tests the SNMP of the default server.

REMARKS To see the examples, type: "get-help Test-VMHostSnmp -examples". For more information, type: "get-help Test-VMHostSnmp -detailed". For technical information, type: "get-help Test-VMHostSnmp -full". For online help, type: "get-help Test-VMHostSnmp -online"

CHAPTER 30

Update Commands

This page contains details on **Update** commands.

Update-Tools

NAME Update-Tools

SYNOPSIS This cmdlet upgrades VMware Tools on the specified virtual machine guest OS.

SYNTAX Update-Tools [-NoReboot] [-RunAsync] [[-Guest] <VMGuest[]>] [<CommonParameters>]

Update-Tools [-NoReboot] [-RunAsync] [[-VM] <VirtualMachine[]>] [[-Server] <VIServer[]>] [<CommonParameters>]

DESCRIPTION This cmdlet upgrades the VMware Tools on the specified virtual machine guest OS. VMware Tools must be installed prior to updating it. After VMware Tools is updated, the virtual machine is restarted unless the NoReboot parameter is specified.

PARAMETERS

-NoReboot Indicates that you do not want to reboot the system after updating VMware Tools. This parameter is supported only for Windows operating systems. NoReboot passes the following set of options to the VMware Tools installer on the guest OS:

/s /v"/qn REBOOT=ReallySuppress"

However, the virtual machine might still reboot after updating VMware Tools, depending on the currently installed VMware Tools version, the VMware Tools version to which you want to upgrade, and the vCenter Center/ESX versions.

-RunAsync Indicates that the command returns immediately without waiting for the task to complete. In this mode, the output of the cmdlet is a Task object. For more information about the RunAsync parameter run "help About_RunAsync" in the vSphere PowerCLI console.

-Guest <VMGuest[]> Specifies the guest operating systems on which you want to update VMware Tools.

-VM <VirtualMachine[]> Specifies a list of the virtual machines whose VMware Tools you want to upgrade.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>Update-Tools VM

Updates the VMware Tools on the specified virtual machine. The virtual machine must be powered on.

————— Example 2 —————

C:PS>Get-VMGuest VM | Update-Tools

Updates the VMware Tools on the virtual machine specified by its guest operating system. The virtual machine must be powered on.

REMARKS To see the examples, type: “get-help Update-Tools -examples”. For more information, type: “get-help Update-Tools -detailed”. For technical information, type: “get-help Update-Tools -full”. For online help, type: “get-help Update-Tools -online”

CHAPTER 31

Wait Commands

This page contains details on **Wait** commands.

Wait-Task

NAME Wait-Task

SYNOPSIS This cmdlet waits for the completion of the specified tasks.

SYNTAX Wait-Task [-Task] <Task[]> [<CommonParameters>]

DESCRIPTION This cmdlet waits for the specified tasks to complete or fail before allowing the next command input. The task progress is observed in real time on the console screen.

PARAMETERS

-Task <Task[]> Specifies the tasks you want to wait to complete.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

————— Example 1 —————

C:PS>$task = Remove-VM -VM VM -Confirm -RunAsync

Wait-Task -Task $task

Waits for the virtual machine to be removed before allowing the next command input.

REMARKS To see the examples, type: “get-help Wait-Task -examples”. For more information, type: “get-help Wait-Task -detailed”. For technical information, type: “get-help Wait-Task -full”. For online help, type: “get-help Wait-Task -online”

Wait-Tools

NAME Wait-Tools

SYNOPSIS This cmdlet waits for VMware Tools on the specified virtual machines to load.

SYNTAX Wait-Tools [-VM] <VirtualMachine[]> [[-TimeoutSeconds] <Int32>] [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-Server <VIServer[]>] [<CommonParameters>]

Wait-Tools [[-TimeoutSeconds] <Int32>] [-HostCredential <PSCredential>] [-HostUser <String>] [-HostPassword <SecureString>] [-Guest] <VMGuest[]> [<CommonParameters>]

DESCRIPTION This cmdlet waits for VMware Tools of the specified virtual machines to load. The cmdlet returns the virtual machines or guests on which VMware Tools have loaded successfully within the specified time limits. You can cancel the operation before completion using Ctrl+C. The successful completion of Wait-Tools means that VMware Tools have loaded, but it does not guarantee for the start of other services. Updating the returned VMGuest objects requires additional communication with VMware Tools and some of their properties (OSFullName, IPAddress, HostName, and other) might be still empty right after the completion of Wait-Tools.

PARAMETERS

-VM <VirtualMachine[]> Specifies the virtual machines for which you want to wait VMware Tools to load.

-TimeoutSeconds <Int32> Specifies the time period in seconds to wait for VMware Tools to start before cancelling the operation.

-HostCredential <PSCredential> Specifies credentials for authenticating with the ESX/ESXi host of the specified virtual machine. This parameter is needed only if you have authenticated with vCenter Server via SSPI. If SSPI is not used, the credentials for authentication with vCenter Server are used.

-HostUser <String> Specifies a username for authenticating with the ESX/ESXi host of the specified virtual machine. This parameter is needed only if you have authenticated with vCenter Server via SSPI. If SSPI is not used, the username for authentication with vCenter Server is used.

-HostPassword <SecureString> Specifies a password for authenticating with the ESX host of the specified virtual machine. This parameter is needed only if you have authenticated with the vCenter Server via SSPI. If no SSPI is used, the password for authentication with vCenter Server is used.

-Server <VIServer[]> Specifies the vCenter Server systems on which you want to run the cmdlet. If no value is given to this parameter, the command runs on the default servers. For more information about default servers, see the description of Connect-VIServer.

-Guest <VMGuest[]> Specifies the guest operating systems for which you want to wait VMware Tools to load.

<CommonParameters> This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https://go.microsoft.com/fwlink/?LinkID=113216).

———— Example 1 ————

C:PS>$vm = Start-VM VM* | Wait-Tools

Starts the virtual machines with names starting with VM and Waits for their VMware Tools to load.

———— Example 2 ————

C:PS>Wait-Tools -VM $vm -TimeoutSeconds 180

Waits for the VMware Tools of the virtual machines in the $vm variable to start. If VMware Tools do not load after 180 seconds, the operation is aborted.

———— Example 3 ————

C:PS>Wait-Tools -VM VM* -TimeoutSeconds 120 -HostCredential $vmhostCredential

Waits for the VMware Tools of the virtual machines in the $vm variable to start. If VMware Tools do not load after 120 seconds, the operation is aborted. Host credentials are required when you run the cmdlet on environments older than vSphere 4.0.

————— Example 4 —————

C:PS>Restart-VMGuest WindowsXP | Wait-Tools

Restart the guest operating system WindowsXP and waits for the VMware Tools to load.

REMARKS To see the examples, type: “get-help Wait-Tools -examples”. For more information, type: “get-help Wait-Tools -detailed”. For technical information, type: “get-help Wait-Tools -full”. For online help, type: “get-help Wait-Tools -online”