f:\dsbuildroot\wswemdmain\1033\Art\wss_logo\wss_logo.gif

System Center Virtual Machine Manager 2008 Cmdlet Reference

Microsoft Corporation

Published: March 2009

Abstract

System Center Virtual Machine Manager (VMM) 2008 is a server application that you can use to manage virtual machines on a variety of virtualization platforms. As an alternative to using the VMM Administrator Console to administer your Virtual Machine Manager environment, you can use the cmdlets in the Windows PowerShell - Virtual Machine Manager command shell, which is an administrator-focused command-line shell. This document provides the Help topics for the VMM 2008 cmdlets.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This cmdlet reference is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, Active Directory, Excel, Hyper-V, SQL Server, Windows, Windows PowerShell, and Windows Vista are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

[Introduction 10](#_Toc225244407)

[Cmdlets Grouped by Noun 10](#_Toc225244408)

[Certificate 11](#_Toc225244409)

[Get-Certificate 11](#_Toc225244410)

[Cluster 13](#_Toc225244411)

[Discover-Cluster 13](#_Toc225244412)

[Computer 16](#_Toc225244413)

[Discover-Computer 16](#_Toc225244414)

[CPUType 19](#_Toc225244415)

[Get-CPUType 19](#_Toc225244416)

[DependentLibraryObject 20](#_Toc225244417)

[Get-DependentLibraryObject 20](#_Toc225244418)

[DirectoryChildItem 22](#_Toc225244419)

[Get-DirectoryChildItem 22](#_Toc225244420)

[GuestOSProfile 24](#_Toc225244421)

[Get-GuestOSProfile 24](#_Toc225244422)

[New-GuestOSProfile 25](#_Toc225244423)

[Remove-GuestOSProfile 29](#_Toc225244424)

[Set-GuestOSProfile 31](#_Toc225244425)

[HardDisk 35](#_Toc225244426)

[Copy-HardDisk 35](#_Toc225244427)

[HardwareProfile 39](#_Toc225244428)

[Get-HardwareProfile 39](#_Toc225244429)

[New-HardwareProfile 40](#_Toc225244430)

[Remove-HardwareProfile 46](#_Toc225244431)

[Set-HardwareProfile 47](#_Toc225244432)

[ISO 52](#_Toc225244433)

[Get-ISO 52](#_Toc225244434)

[Remove-ISO 53](#_Toc225244435)

[Set-ISO 55](#_Toc225244436)

[Job 58](#_Toc225244437)

[Get-Job 58](#_Toc225244438)

[Restart-Job 59](#_Toc225244439)

[Stop-Job 61](#_Toc225244440)

[LibraryRating 63](#_Toc225244441)

[Get-LibraryRating 63](#_Toc225244442)

[LibraryServer 65](#_Toc225244443)

[Add-LibraryServer 65](#_Toc225244444)

[Get-LibraryServer 68](#_Toc225244445)

[Remove-LibraryServer 69](#_Toc225244446)

[Set-LibraryServer 71](#_Toc225244447)

[Add-LibraryShare 73](#_Toc225244448)

[Discover-LibraryShare 75](#_Toc225244449)

[Get-LibraryShare 76](#_Toc225244450)

[Refresh-LibraryShare 78](#_Toc225244451)

[Remove-LibraryShare 79](#_Toc225244452)

[Set-LibraryShare 80](#_Toc225244453)

[MachineConfig 82](#_Toc225244454)

[Get-MachineConfig 82](#_Toc225244455)

[New-MachineConfig 83](#_Toc225244456)

[Remove-MachineConfig 85](#_Toc225244457)

[NetworkLocation 88](#_Toc225244458)

[Get-NetworkLocation 88](#_Toc225244459)

[OperatingSystem 89](#_Toc225244460)

[Get-OperatingSystem 89](#_Toc225244461)

[P2V 90](#_Toc225244462)

[New-P2V 90](#_Toc225244463)

[Patch 101](#_Toc225244464)

[Add-Patch 101](#_Toc225244465)

[PhysicalAddress 103](#_Toc225244466)

[New-PhysicalAddress 103](#_Toc225244467)

[PROTip 106](#_Toc225244468)

[Dismiss-PROTip 106](#_Toc225244469)

[Get-PROTip 107](#_Toc225244470)

[Invoke-PROTip 109](#_Toc225244471)

[Set-PROTip 110](#_Toc225244472)

[Script 113](#_Toc225244473)

[Get-Script 113](#_Toc225244474)

[Remove-Script 115](#_Toc225244475)

[Set-Script 117](#_Toc225244476)

[SshPublicKey 120](#_Toc225244477)

[Get-SshPublicKey 120](#_Toc225244478)

[Step 122](#_Toc225244479)

[Get-Step 122](#_Toc225244480)

[Template 124](#_Toc225244481)

[Get-Template 124](#_Toc225244482)

[New-Template 125](#_Toc225244483)

[Remove-Template 134](#_Toc225244484)

[Set-Template 135](#_Toc225244485)

[V2V 142](#_Toc225244486)

[New-V2V 142](#_Toc225244487)

[VirtualCOMPort 151](#_Toc225244488)

[Get-VirtualCOMPort 151](#_Toc225244489)

[Set-VirtualCOMPort 152](#_Toc225244490)

[VirtualDiskDrive 159](#_Toc225244491)

[Compress-VirtualDiskDrive 159](#_Toc225244492)

[Convert-VirtualDiskDrive 160](#_Toc225244493)

[Expand-VirtualDiskDrive 163](#_Toc225244494)

[Get-VirtualDiskDrive 165](#_Toc225244495)

[New-VirtualDiskDrive 167](#_Toc225244496)

[Remove-VirtualDiskDrive 174](#_Toc225244497)

[Set-VirtualDiskDrive 177](#_Toc225244498)

[VirtualDVDDrive 180](#_Toc225244499)

[Get-VirtualDVDDrive 180](#_Toc225244500)

[New-VirtualDVDDrive 181](#_Toc225244501)

[Remove-VirtualDVDDrive 185](#_Toc225244502)

[Set-VirtualDVDDrive 187](#_Toc225244503)

[VirtualFloppyDisk 192](#_Toc225244504)

[Get-VirtualFloppyDisk 192](#_Toc225244505)

[Remove-VirtualFloppyDisk 193](#_Toc225244506)

[Set-VirtualFloppyDisk 195](#_Toc225244507)

[VirtualFloppyDrive 198](#_Toc225244508)

[Get-VirtualFloppyDrive 198](#_Toc225244509)

[Set-VirtualFloppyDrive 200](#_Toc225244510)

[VirtualHardDisk 204](#_Toc225244511)

[Get-VirtualHardDisk 204](#_Toc225244512)

[Move-VirtualHardDisk 205](#_Toc225244513)

[Remove-VirtualHardDisk 208](#_Toc225244514)

[Set-VirtualHardDisk 209](#_Toc225244515)

[VirtualizationManager 213](#_Toc225244516)

[Add-VirtualizationManager 213](#_Toc225244517)

[Get-VirtualizationManager 215](#_Toc225244518)

[Refresh-VirtualizationManager 216](#_Toc225244519)

[Remove-VirtualizationManager 218](#_Toc225244520)

[Set-VirtualizationManager 220](#_Toc225244521)

[VirtualNetwork 223](#_Toc225244522)

[Get-VirtualNetwork 223](#_Toc225244523)

[New-VirtualNetwork 225](#_Toc225244524)

[Remove-VirtualNetwork 231](#_Toc225244525)

[Set-VirtualNetwork 233](#_Toc225244526)

[VirtualNetworkAdapter 238](#_Toc225244527)

[Get-VirtualNetworkAdapter 238](#_Toc225244528)

[New-VirtualNetworkAdapter 239](#_Toc225244529)

[Remove-VirtualNetworkAdapter 245](#_Toc225244530)

[Set-VirtualNetworkAdapter 248](#_Toc225244531)

[VirtualSCSIAdapter 252](#_Toc225244532)

[Get-VirtualSCSIAdapter 252](#_Toc225244533)

[New-VirtualSCSIAdapter 253](#_Toc225244534)

[Remove-VirtualSCSIAdapter 257](#_Toc225244535)

[Set-VirtualSCSIAdapter 258](#_Toc225244536)

[VM 261](#_Toc225244537)

[DisableUndoDisk-VM 261](#_Toc225244538)

[DiscardSavedState-VM 262](#_Toc225244539)

[Get-VM 264](#_Toc225244540)

[Move-VM 265](#_Toc225244541)

[New-VM 270](#_Toc225244542)

[Refresh-VM 280](#_Toc225244543)

[Register-VM 281](#_Toc225244544)

[Remove-VM 284](#_Toc225244545)

[Repair-VM 286](#_Toc225244546)

[Resume-VM 288](#_Toc225244547)

[SaveState-VM 290](#_Toc225244548)

[Set-VM 291](#_Toc225244549)

[Shutdown-VM 300](#_Toc225244550)

[Start-VM 302](#_Toc225244551)

[Stop-VM 303](#_Toc225244552)

[Store-VM 305](#_Toc225244553)

[Suspend-VM 308](#_Toc225244554)

[VMCheckpoint 310](#_Toc225244555)

[Get-VMCheckpoint 310](#_Toc225244556)

[Merge-VMCheckpoint 312](#_Toc225244557)

[New-VMCheckpoint 314](#_Toc225244558)

[Remove-VMCheckpoint 317](#_Toc225244559)

[Restore-VMCheckpoint 319](#_Toc225244560)

[Set-VMCheckpoint 321](#_Toc225244561)

[VMDK 323](#_Toc225244562)

[Copy-VMDK 323](#_Toc225244563)

[VMHost 327](#_Toc225244564)

[Add-VMHost 327](#_Toc225244565)

[Associate-VMHost 335](#_Toc225244566)

[Get-VMHost 337](#_Toc225244567)

[Move-VMHost 339](#_Toc225244568)

[Refresh-VMHost 341](#_Toc225244569)

[Remove-VMHost 343](#_Toc225244570)

[Set-VMHost 346](#_Toc225244571)

[Update-VMHost 352](#_Toc225244572)

[VMHostCluster 354](#_Toc225244573)

[Add-VMHostCluster 354](#_Toc225244574)

[Get-VMHostCluster 356](#_Toc225244575)

[Move-VMHostCluster 358](#_Toc225244576)

[Refresh-VMHostCluster 359](#_Toc225244577)

[Remove-VMHostCluster 360](#_Toc225244578)

[Set-VMHostCluster 362](#_Toc225244579)

[VMHostDisk 366](#_Toc225244580)

[Get-VMHostDisk 366](#_Toc225244581)

[VMHostGroup 368](#_Toc225244582)

[Get-VMHostGroup 368](#_Toc225244583)

[Move-VMHostGroup 369](#_Toc225244584)

[New-VMHostGroup 371](#_Toc225244585)

[Remove-VMHostGroup 373](#_Toc225244586)

[Set-VMHostGroup 374](#_Toc225244587)

[VMHostNetworkAdapter 378](#_Toc225244588)

[Add-VMHostNetworkAdapter 378](#_Toc225244589)

[Get-VMHostNetworkAdapter 382](#_Toc225244590)

[Remove-VMHostNetworkAdapter 384](#_Toc225244591)

[Set-VMHostNetworkAdapter 385](#_Toc225244592)

[VMHostRating 390](#_Toc225244593)

[Get-VMHostRating 390](#_Toc225244594)

[VMHostVolume 399](#_Toc225244595)

[Get-VMHostVolume 399](#_Toc225244596)

[Set-VMHostVolume 401](#_Toc225244597)

[VMMManagedComputer 404](#_Toc225244598)

[Get-VMMManagedComputer 404](#_Toc225244599)

[Reassociate-VMMManagedComputer 406](#_Toc225244600)

[Update-VMMManagedComputer 408](#_Toc225244601)

[VMMServer 410](#_Toc225244602)

[Backup-VMMServer 410](#_Toc225244603)

[Get-VMMServer 412](#_Toc225244604)

[Set-VMMServer 414](#_Toc225244605)

[VMMUserRole 421](#_Toc225244606)

[Get-VMMUserRole 421](#_Toc225244607)

[New-VMMUserRole 423](#_Toc225244608)

[Remove-VMMUserRole 425](#_Toc225244609)

[Set-VMMUserRole 427](#_Toc225244610)

[VMPerformance 432](#_Toc225244611)

[Get-VMPerformance 432](#_Toc225244612)

[VMRCCertificateRequest 434](#_Toc225244613)

[New-VMRCCertificateRequest 434](#_Toc225244614)

[VMwareResourcePool 437](#_Toc225244615)

[Get-VMwareResourcePool 437](#_Toc225244616)

[VMXMachineConfig 439](#_Toc225244617)

[Get-VMXMachineConfig 439](#_Toc225244618)

[New-VMXMachineConfig 440](#_Toc225244619)

[Remove-VMXMachineConfig 441](#_Toc225244620)

# Introduction

This document contains the output that you receive after you type the following command in the Windows PowerShell – Virtual Machine Manager command shell:

Get-Command -PSSnapin Microsoft.SystemCenter.VirtualMachineManager | Sort‑Object Noun, Verb | Get-Help -detailed > VMM2008CmdletHelpSortedNounVerb.txt

This command generates a list of cmdlet Help topics that is sorted by noun. This list contains the same Help for each cmdlet that you can view in the Windows PowerShell – Virtual Machine Manager command shell by typing the following command:

Get-Help *<Cmdlet-Name>* -detailed

For more information about the Windows PowerShell – Virtual Machine Manager command shell, see [Introduction to Scripting in Virtual Machine Manager 2008](http://go.microsoft.com/fwlink/?LinkId=146907) (http://go.microsoft.com/fwlink/?LinkId=146907).

# Cmdlets Grouped by Noun

In the names of Windows PowerShell cmdlets, including Virtual Machine Manager cmdlets, the word that precedes the hyphen is the verb, and the word that follows the hyphen is the noun (*Verb-Noun*). This command reference groups the cmdlets into sets that are alphabetized first by the noun and then by the verb. To view the entire list of cmdlets in the left margin of this document, click **View** and then select the **Document Map** check box.

## Certificate

### Get-Certificate

SYNOPSIS

Gets a security certificate object from a VMware VirtualCenter server or fr

om an ESX Server.

SYNTAX

Get-Certificate [-ComputerName] <String> [-TCPPort <Int32>] [-VMMServer [<S

tring ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets an object that represents a security certificate from a VMware Virtual

Center Server or from an ESX Server. You can use this cmdlet to import a no

n-trusted certificate into Virtual Machine Manager so that you can use the

certificate with the Add-VirtualizationManager cmdlet or the Set-Virtualiza

tionManager cmdlet.

The certificate is required in order to establish an SSL connection between

the Virtual Machine Manager server and the VirtualCenter Server or ESX Ser

ver.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Retrieve the security certificate for the specified VMware VirtualCenter

Server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-Certificate -Computername "VirtMgrServer01.Contoso.com"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets the object that represents the security certificate

from the VMware VirtualCenter server named VirtMgrServer01, located in the

Contoso.com domain, and displays the security certficate string.

2: Retrieve the security certificate for the specified VMware ESX Server ho

st.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $ESXCert = Get-Certificate -Computername "nnn.nnn.nnn.nnn"

PS C:\> $ESXCert | Get-Member

The first command connects to VMMServer1.

The second command gets the object that represents the security certificate

from the ESX server (whose IP address is represented in this example by "n

nn.nnn.nnn.nnn") and stores the certificate object in $ESXCert.

The last command passes the contents of $ESXCert to the Get-Member cmdlet,

which displays the .NET type for the certificate object:

TypeName:

Microsoft.SystemCenter.VirtualMachineManager.Remoting.ClientCertificate

The Get-Member cmdlet also displays a list of methods and properties for th

e certificate object.

REMARKS

For more information, type: "get-help Get-Certificate -detailed".

For technical information, type: "get-help Get-Certificate -full".

## Cluster

### Discover-Cluster

SYNOPSIS

Discovers the specified failover cluster in a Virtual Machine Manager envir

onment.

SYNTAX

Discover-Cluster [-ComputerName] <String> -Credential <PSCredential> [-JobV

ariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<Str

ing ServerConnection>]] [<CommonParameters>]

Discover-Cluster -LibraryServer [<String LibraryServer>] [-JobVariable <Str

ing>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String ServerCo

nnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Discovers whether the specified name represents a Windows Server 2008 failo

ver cluster or one of its nodes, and, if so, returns an object that contain

s more information about the failover cluster. The information returned by

the Discover-Cluster cmdlet includes cluster name, nodes of the cluster, an

d highly available file servers hosted by the cluster.

You cannot use Virtual Machine Manager to create a failover cluster. You ca

n, however, use the Discover-Cluster cmdlet to discover an existing failove

r cluster configured earlier by using the Failover Cluster Management conso

le.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Discover all nodes of a failover cluster from the cluster name.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Cluster = Discover-Cluster -ComputerName "VMHostCluster01.Contoso.

com" -Credential $Credential

PS C:\> $ClusterNodes = $Cluster.ClusterNodes

PS C:\> Write-Host $ClusterNodes

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are a domain account with administrator r

ights on all nodes of the failover cluster that you want to find more infor

mation about.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands use this

server by default.

The third command uses Discover-Cluster to create a cluster object after qu

erying the failover cluster named VMHostCluster01 and stores the cluster ob

ject in $Cluster, using $Credential to provide your credentials to Discover

-Cluster. Notice that the -ComputerName parameter treats the name of the cl

uster as if it were the name of a computer.

The fourth command gets the names of all nodes of the failover cluster in $

Cluster.

The last command displays the FQDN names of the cluster nodes to the user.

2: Discover all nodes of a failover cluster from one of the node names.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Cluster = Discover-Cluster -ComputerName "VMHostNode02.Contoso.com

" -Credential $Credential

PS C:\> $ClusterName = $Cluster.Name

PS C:\> $ClusterNodes = $Cluster.ClusterNodes

PS C:\> Write-Host $ClusterNodes

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password and stores your credentials in $Credential.

The second command connects to VMMServer1.

The third command uses Discover-Cluster to create a cluster object after qu

erying a failover cluster node named VMHostNode02 and stores the cluster ob

ject in $Cluster.

The next two commands retrieve the cluster name and the name of each node t

hat belongs to this failover cluster from $Cluster.

The last command displays the FQDN name of the nodes in the cluster to the

user.

3: Discovers, from the cluster name, all highly available file servers host

ed by that failover cluster.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Cluster = Discover-Cluster -ComputerName "VMHostCluster03.Contoso.

com" -Credential $Credential

PS C:\> $HAFileServers = $Cluster.HAFileServers

PS C:\> Write-Host $HAFileServers

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password and stores your credentials in $Credential.

The second command connects to VMMServer1.

The third command uses Discover-Cluster to create a cluster object after qu

erying the failover cluster named VMHostCluster03 and stores the cluster ob

ject in $Cluster.

The fourth command gets the names of all highly available file servers host

ed by this failover cluster from $Cluster. NOTE: This example assumes that

the failover cluster is hosting at least one highly available file server.

The last command displays the FQDN name of the highly available file server

s in $Cluster to the user.

REMARKS

For more information, type: "get-help Discover-Cluster -detailed".

For technical information, type: "get-help Discover-Cluster -full".

## Computer

### Discover-Computer

SYNOPSIS

Discovers computers by querying Active Directory, and returns the computer

objects.

SYNTAX

Discover-Computer -Domain <String> [-ComputerNameFilter <String>] [-Credent

ial <PSCredential>] [-DiscoveryID <Guid>] [-ExcludeVMMHost] [-ExcludeVMMLib

rary] [-FindHyperVHosts] [-FindVSHosts] [-JobVariable <String>] [-RunAsynch

ronously] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

Discover-Computer [-ComputerName] <String> [-Credential <PSCredential>] [-E

xcludeVMMHost] [-ExcludeVMMLibrary] [-FindHyperVHosts] [-FindVSHosts] [-VMM

Server [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Discovers one or more computers by querying Active Directory, and returns t

he computer objects.

You can use this cmdlet to query Active Directory for computers based on sp

ecified criteria, or a combination of criteria, including:

\* The fully qualified domain name (FQDN) of a computer.

\* All or part of the computer name.

\* The name of a domain.

\* All computers except hosts managed by Virtual Machine Manager.

\* All computers except library servers managed by Virtual Machine Manager.

\* Only Hyper-V hosts.

\* Only Virtual Server hosts.

NOTE: If you add a new computer (such as a host or library server) located

in an Active Directory domain to Virtual Machine Manager and then immediate

ly run the Discover-Computer cmdlet, the cmdlet might not immediately disco

ver the new computer when it searches Active Directory. This delay is becau

se data about the new computer might not have replicated yet across the Act

ive Directory domain. If you are a Domain Administrator, you can use the Ac

tive Directory Sites and Services console to force the data to replicate im

mediately.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Domain <String>

Specifies a fully qualified domain name (FQDN) for an Active Directory

domain.

Example format: "ThisDomain.Corp.Contoso.com"

-ComputerNameFilter <String>

Specifies the partial or full name of a computer that the cmdlet will t

ry to discover in Active Directory.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-DiscoveryID <Guid>

For internal use only (not for use in your code).

-ExcludeVMMHost

Excludes virtual machine host servers currently managed by VMM.

-ExcludeVMMLibrary

Excludes library servers currently managed by VMM.

-FindHyperVHosts

Searches for computers running a Windows Server 2008 operating system o

n which the Hyper-V server role is enabled.

-FindVSHosts

Searches for computers running a Windows server operating system on whi

ch Virtual Server 2005 R2 SP1 is installed.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1. Discover computers in a specific domain that meet the specified criteria

.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Discover-Computer -ComputerNameFilter "host" -Domain "Contoso.com"

-FindHyperVHosts -ExcludeVMMHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command queries Active Directory and returns a list of Hyper-V c

omputer objects for computers that are located in the Contoso.com domain, t

hat have a name starting with "host", and that are not managed by VMM.

NOTE: When you use Discover-Computer with the -Domain parameter, you must s

pecify the fully qualified domain name.

2. Discover a specific computer by name and validate its properties in Acti

ve Directory.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Discover-Computer -ComputerName "VMHost02.Contoso.com"

The first command connects to VMMServer1.

The second command uses the fully qualified domain name (FQDN) of the compu

ter named VMHost02 to discover this computer in Active Directory, returns t

he computer object, and displays the computer object name and its propertie

s to the user.

3: Discover all computers in the specified domain except for VMM library se

rvers.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Discover-Computer -ComputerNameFilter "vmm" -Domain "Contoso.com" -

ExcludeVMMLibrary

The first command connects to VMMServer1.

The second command queries Active Directory for all computers in the Contos

o.com domain that include "vmm" in the computer name except for Virtual Mac

hine Manager library servers.

REMARKS

For more information, type: "get-help Discover-Computer -detailed".

For technical information, type: "get-help Discover-Computer -full".

## CPUType

### Get-CPUType

SYNOPSIS

Gets objects that represent CPU types for use in virtual machines, or for u

se in templates or hardware profiles used to create virtual machines, in a

Virtual Machine Manager environment.

NOTE: In VMM 2007, this cmdlet was named Get-ProcessorType.

SYNTAX

Get-CPUType [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [<CommonP

arameters>]

DETAILED DESCRIPTION

Gets objects that represent the types of CPU that are available for use in

virtual machines, or for use in templates or hardware profiles used to crea

te virtual machines, in a Virtual Machine Manager environment. The type of

CPU is one of the factors that the Virtual Machine Manager placement proces

s uses to determine which computers (among all available computers) are sui

table hosts for a specific virtual machine.

PARAMETERS

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all available processor types.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-CPUType

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets all objects that represent any available CPU type f

rom the VMM database, formats the information about each CPU type in a list

, and displays this information to the user.

REMARKS

For more information, type: "get-help Get-CPUType -detailed".

For technical information, type: "get-help Get-CPUType -full".

## DependentLibraryObject

### Get-DependentLibraryObject

SYNOPSIS

Identifies dependencies between Virtual Machine Manager objects.

SYNTAX

Get-DependentLibraryObject [-LibraryObject] [<VirtualHardDisk ISO VM Script

Template VirtualFloppyDisk>] [-VMMServer [<String ServerConnection>]] [<Co

mmonParameters>]

Get-DependentLibraryObject [-LibraryServer] [<String LibraryServer>] [-VMMS

erver [<String ServerConnection>]] [<CommonParameters>]

Get-DependentLibraryObject [-LibraryShare] <LibraryShare> [-VMMServer [<Str

ing ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Identifies dependencies between Virtual Machine Manager objects.

You can use the Get-DependentLibraryObject cmdlet to identify objects that

are dependent on the existence of:

\* The specified library object

\* Any object on the specified library share

\* Any object on the specified library server

PARAMETERS

-LibraryShare <LibraryShare>

Specifies a VMM library share object.

-LibraryObject [<VirtualHardDisk ISO VM Script Template VirtualFloppyDisk>]

Specifies a VMM library object to test to determine whether other objec

ts are dependent on this object.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Find all of the objects (if any exist) that depend on a particular virtu

al hard disk.

PS C:\> $VHD = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | wher

e { $\_.Name -eq “VHD01” -and $\_.LibraryServer.Name -eq "FileServer01.Conto

so.com” }

PS C:\> Get-DependentLibraryObject -LibraryObject $VHD

The first command selects from all virtual hard disk objects in the library

on VMMServer1 the object that represents the virtual hard disk named VHD01

(stored on the library server called FileServer01) and stores this virtual

hard disk object in variable $VHD. This example assumes that only one virt

ual hard disk named VHD01 exists.

The second command returns all of the library objects that are dependent on

VHD01.

If dependent objects exist, removing this virtual hard disk will modify tho

se dependent objects so that they no longer reference the removed virtual h

ard disk. Thus, if VHD01 is associated with a specific virtual machine or w

ith a specific template, that virtual machine or template is modified so th

at it no longer references VHD01 after it is removed.

REMARKS

For more information, type: "get-help Get-DependentLibraryObject -detailed"

.

For technical information, type: "get-help Get-DependentLibraryObject -full

".

## DirectoryChildItem

### Get-DirectoryChildItem

SYNOPSIS

Gets all files and subdirectories in the specified directory on a virtual m

achine host or on a library server managed by Virtual Machine Manager.

SYNTAX

Get-DirectoryChildItem -LibraryServer [<String LibraryServer>] -Path <Strin

g> [<CommonParameters>]

Get-DirectoryChildItem -Path <String> -VMHost [<String Host>] [<CommonParam

eters>]

DETAILED DESCRIPTION

Gets all files and subdirectories immediately under the specified directory

(folder) on a virtual machine host or on a library server managed by Virtu

al Machine Manager. If you specify a share path (such as \\ServerName\Share

Name\Directory\FileName), the subdirectories of the share path are returned

.

If you use the Get-DirectoryChildItem cmdlet to retrieve files and subdirec

tories on a library server, you must specify a path to a valid library shar

e. For example, the share path to the default library share installed by Se

tup when you first install the Virtual Machine Manager service is:

\\VMMServerName.DomainName.com\MSSCVMMLibrary

PARAMETERS

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get the subdirectories for the specified path on a host server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> Get-DirectoryChildItem -VMHost $VMHost -Path "C:\"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host named VMHost01

from the VMM database and stores the host object in variable $VMHost.

The last command displays the name and other information about each file an

d subdirectory immediately under the C:\ drive on VMHost01.

2: Get the subdirectories for the specified path on a library server.

PS C:\> $LibServ = Get-LibraryServer -VMMServer VMMServer1.Contoso.com -Com

puterName "FileServer02.Contoso.com"

PS C:\> Get-DirectoryChildItem -LibraryServer $LibServ -Path "\\FileServer0

2.Contoso.com\MSSCVMMLibrary"

The first command gets the object that represents the library server named

FileServer02 from VMMServer1 and stores the library server object in $LibSe

rv.

The second command displays the name, parent directory, and other informati

on about each file stored in the directory for the default library share on

FileServer02. You must specify the complete path to the library share.

NOTE: This example assumes that the default VMM library share (MSSCVMMLibra

ry) is used in your environment. To determine the names of library shares,

type: Get-LibraryShare | select Name

REMARKS

For more information, type: "get-help Get-DirectoryChildItem -detailed".

For technical information, type: "get-help Get-DirectoryChildItem -full".

## GuestOSProfile

### Get-GuestOSProfile

SYNOPSIS

Gets guest operating system profile objects from the Virtual Machine Manage

r library.

SYNTAX

Get-GuestOSProfile [[-Name] <String>] [-VMMServer [<String ServerConnection

>]] [<CommonParameters>]

Get-GuestOSProfile [[-Name] <String>] -AnswerFile <Script> [-OperatingSyste

m <OperatingSystem>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets from the Virtual Machine Manager library one or more objects that repr

esent guest operating system profiles.

For more information about guest operating system profiles, type:

Get-Help New-GuestOSProfile -detailed

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-AnswerFile <Script>

Specifies a script object stored in the VMM library to use as an answer

file. The name of the answer file script depends on the operating syst

em that you want to install on a virtual machine:

ANSWER FILE GUEST OS TO INSTALL ON VM

----------- -------------------------

Sysprep.inf Windows XP, Windows Server 2000, or Windows Server 2003

Unattend.xml Windows Vista or Windows Server 2008

Note: In VMM 2007, this parameter was named SysPrepFile.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1. Get all guest operating system profiles from the library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-GuestOSProfile

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets all objects that represent guest operating system p

rofiles from the VMM library and displays information about these profiles

to the user.

2. Get a specific guest operating system profile from the library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-GuestOSProfile | where { $\_.Name -eq "Windows Server 2003 Stand

ard" }

The first command connects to VMMServer1.

The second command gets the object that represents the guest operating syst

em profile named "Windows Server 2003 Standard" and displays information ab

out this profile to the user.

REMARKS

For more information, type: "get-help Get-GuestOSProfile -detailed".

For technical information, type: "get-help Get-GuestOSProfile -full".

### New-GuestOSProfile

SYNOPSIS

Creates a guest operating system profile for use in Virtual Machine Manager

.

SYNTAX

New-GuestOSProfile [-Name] <String> [-AdminPasswordCredential <PSCredential

>] [-AnswerFile <Script>] [-ComputerName <String>] [-Description <String>]

[-FullName <String>] [-GuestOSProfile [<GuestOSProfile String>]] [-GuiRunOn

ceCommands <String[]>] [-JobVariable <String>] [-JoinWorkgroup <String>] [-

OperatingSystem <OperatingSystem>] [-OrgName <String>] [-Owner <String>] [-

ProductKey <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-TimeZone <In

t32>] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-GuestOSProfile [-Name] <String> -JoinDomain <String> -JoinDomainCredent

ial <PSCredential> [-AdminPasswordCredential <PSCredential>] [-AnswerFile <

Script>] [-ComputerName <String>] [-Description <String>] [-FullName <Strin

g>] [-GuestOSProfile [<GuestOSProfile String>]] [-GuiRunOnceCommands <Strin

g[]>] [-JobVariable <String>] [-OperatingSystem <OperatingSystem>] [-OrgNam

e <String>] [-Owner <String>] [-ProductKey <String>] [-PROTipID <Guid>] [-R

unAsynchronously] [-TimeZone <Int32>] [-VMMServer [<String ServerConnection

>]] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a guest operating system profile for use in Virtual Machine Manager

. A guest operating system is the operating system on a virtual machine, in

contrast to a host operating system on the physical host server on which o

ne or more virtual machines are deployed.

A guest operating system profile stores operating system configuration info

rmation. A profile consists of a set of properties containing the most comm

on settings specified in an unattended answer file (such as a Sysprep.inf o

r an Unattend.xml file).

You can create a standalone guest operating system profile or customize a t

emplate or virtual machine to include guest operating system profile settin

gs. The New-GuestOSProfile cmdlet stores the new guest operating system pro

file object in the Virtual Machine Manager library.

You can create a guest operating system profile based on default settings,

based on an existing guest operating system profile, or you can customize s

ettings as you create the profile. If you specify no parameters except the

Name parameter, the New-GuestOSProfile cmdlet creates a default guest opera

ting system profile object.

Guest operating system settings for a virtual machine include:

-AdminPasswordCredential

-AnswerFile

-ComputerName

-Description

-FullName

-GuiRunOnceCommands

-JoinDomain

-JoinDomainCredential

-JoinWorkgroup

-Name

-OperatingSystem

-OrgName

-Owner

-ProductKey

-TimeZone

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-JoinDomain <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the do

main to which you want to join a virtual machine. You can use this para

meter to override the existing value on a template or on a guest operat

ing system profile. You can join a VM to a domain only if a virtual net

work adapter is configured for the VM.

-JoinDomainCredential <PSCredential>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the user name and

password of an account with permission to join a virtual machine to the

domain. A limited rights account should be used for joining machines t

o the domain.

-AdminPasswordCredential <PSCredential>

Specifies the password for the local Administrator account. Specifying

a password (on a new or existing template, on a new or existing guest

operating system profile, or on a new virtual machine) overrides any ex

isting Administrator password.

-AnswerFile <Script>

Specifies a script object stored in the VMM library to use as an answer

file. The name of the answer file script depends on the operating syst

em that you want to install on a virtual machine:

ANSWER FILE GUEST OS TO INSTALL ON VM

----------- -------------------------

Sysprep.inf Windows XP, Windows Server 2000, or Windows Server 2003

Unattend.xml Windows Vista or Windows Server 2008

Note: In VMM 2007, this parameter was named SysPrepFile.

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Description <String>

Specifies a description for the specified object.

-FullName <String>

Specifies the name of the person in whose name a virtual machine is reg

istered.

-GuestOSProfile [<GuestOSProfile String>]

Specifies a guest operating system profile object.

-GuiRunOnceCommands <String[]>

Specifies one or more commands to add to the [GuiRunOnce] section of an

unattended answer file (such as SysPrep.inf or Unattend.xml). Use sing

le quotes around each string enclosed in double quotes.

Example:

-GuiRunOnceCommands '"C:\APF\APFPostSysPrepCopy.cmd PARAMS1"', '"C:\APF

\APFPostSysPrepCopy.cmd PARAMS1"'

For information about how Windows PowerShell uses quotes, type: Get-Hel

p about\_Quoting\_Rules

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-JoinWorkgroup <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the wo

rkgroup to which you want to join a virtual machine. You can use this p

arameter to override the existing value on a template or on a guest ope

rating system profile.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-OrgName <String>

Specifies the name of the organization of the person in whose name a vi

rtual machine is registered.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-ProductKey <String>

Specifies the product key to use for the operating system to be install

ed on a virtual machine. The product key is a 25-digit number that iden

tifies the product license.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-TimeZone <Int32>

Specifies a number (an "index") that identifies a geographical region t

hat shares the same standard time. For a list of time zone indexes, see

"Microsoft Time Zone Index Values" at: http://go.microsoft.com/fwlink/

?LinkId=120935. If no time zone is specified, the default time zone use

d for a virtual machine is the same time zone setting that is on the vi

rtual machine host.

Example: To specify the GMT Standard Time zone, type: -TimeZone 085

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a default guest operating system profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $OS = Get-OperatingSystem -VMMServer "VMMServer1.contoso.com" | whe

re {$\_.Name -eq "64-bit edition of Windows Server 2008 Datacenter"}

PS C:\> New-GuestOSProfile -Name "NewProfile1" -OperatingSystem $OS

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following command uses t

his server by default.

The second command gets from the VMM database the object that represents a

specific operating system (64-bit edition of Windows Server 2008 Datacenter

) and stores the operating system object in variable $OS.

The second command creates a default guest operating system profile named N

ewProfile1 and specifies the name of the operating system.

2: Create a guest operating system profile to use to join a virtual machine

to a domain.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $JoinDomainCredential = Get-Credential

PS C:\> $OS = Get-OperatingSystem -VMMServer "VMMServer1.contoso.com" | whe

re {$\_.Name -eq "Red Hat Enterprise Linux 5 (64 bit)"}

PS C:\> New-GuestOSProfile -Name "NewProfile2" -JoinDomain "Contoso.com" -J

oinDomainCredential $JoinDomainCredential -OperatingSystem $OS

The first command connects to VMMServer1.

The second command creates a Windows PowerShell credential object (PSCreden

tial object) by prompting you for a user name and password with permissions

to join a computer to the domain and stores the credentials in variable $J

oinDomainCredential.

The third command gets the object that represents a specific operating syst

em (Red Hat Enterprise Linux 5 (64 bit) and stores the operating system obj

ect in $OS.

The last command creates a guest operating system profile called NewProfile

2 and specifies the name of the operating system. When you use NewProfile2

to create a virtual machine, the profile will use the specified domain acco

unt and account password to join the virtual machine to the Contoso.com dom

ain.

NOTE: The recommended practice is to use only an account with limited right

s for automatically joining a virtual machine to a domain because the domai

n account name and password appear briefly in plain text on the host server

.

3: Clone an existing guest operating system profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $OSProfile = New-GuestOSProfile -Name "NewProfile3" -ComputerName "

Contoso3" -FullName "Contoso"

PS C:\> $OS = Get-OperatingSystem -VMMServer "VMMServer1.contoso.com" | whe

re {$\_.Name -eq "Windows Small Business Server 2003"}

PS C:\> New-GuestOSProfile -Name "NewProfile4" -GuestOSProfile $OSProfile -

ComputerName "Contoso4"

The first command connects to VMMServer1.

The second command creates a guest operating system profile called NewProfi

le3, specifies that the computer name is Contoso3, that the full name is Co

ntoso, and stores the profile object in $OSProfile.

The third command gets the object that represents a specific operating syst

em (Windows Small Business Server 2003) and stores the operating system obj

ect in $OS.

The last command creates another guest operating system profile, called New

Profile4, which is based on NewProfile3 but modifies the computer name and

specifies the name of the operating system. All other settings in NewProfil

e4 are identical to those in NewProfile3.

REMARKS

For more information, type: "get-help New-GuestOSProfile -detailed".

For technical information, type: "get-help New-GuestOSProfile -full".

### Remove-GuestOSProfile

SYNOPSIS

Removes a guest operating system profile object from Virtual Machine Manage

r.

SYNTAX

Remove-GuestOSProfile [-GuestOSProfile] [<GuestOSProfile String>] [-Confirm

] [-Force] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously]

[<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent guest operating system profiles

from the Virtual Machine Manager library.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-GuestOSProfile [<GuestOSProfile String>]

Specifies a guest operating system profile object.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific guest operating system profile from the library.

PS C:\> $OSProfile = Get-GuestOSProfile –VMMServer VMMServer1.Contoso.com |

where { $\_.Name –eq "OSP-1" }

PS C:\> Remove-GuestOSProfile -GuestOSProfile $OSProfile

The first command gets the object that represents the guest operating syste

m profile named OSP-1 from the VMM library on VMMServer1 and stores the ope

rating system profile object in variable $OSProfile.

The second command removes OSP-1 from the library.

2: Remove all operating system profiles without being prompted to confirm e

ach deletion.

PS C:\> $OSProfiles = Get-GuestOSProfile -VMMServer VMMServer1.Contoso.com

PS C:\> $OSProfiles | Remove-GuestOSProfile

The first command gets all operating system profile objects from VMMServer1

and stores the profile objects in $OSProfiles (an object array).

The second command passes each object in $OSProfiles to the Remove-OSProfil

e cmdlet, which removes each of the operating system profile objects from t

he Virtual Machine Manager library.

REMARKS

For more information, type: "get-help Remove-GuestOSProfile -detailed".

For technical information, type: "get-help Remove-GuestOSProfile -full".

### Set-GuestOSProfile

SYNOPSIS

Changes the properties of a guest operating system profile used in Virtual

Machine Manager.

SYNTAX

Set-GuestOSProfile [-GuestOSProfile] [<GuestOSProfile String>] [-AdminPassw

ordCredential <PSCredential>] [-AnswerFile <Script>] [-ComputerName <String

>] [-Description <String>] [-FullName <String>] [-GuiRunOnceCommands <Strin

g[]>] [-JobVariable <String>] [-JoinWorkgroup <String>] [-Name <String>] [-

OperatingSystem <OperatingSystem>] [-OrgName <String>] [-Owner <String>] [-

ProductKey <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-TimeZone <In

t32>] [<CommonParameters>]

Set-GuestOSProfile [-GuestOSProfile] [<GuestOSProfile String>] -JoinDomain

<String> -JoinDomainCredential <PSCredential> [-AdminPasswordCredential <PS

Credential>] [-AnswerFile <Script>] [-ComputerName <String>] [-Description

<String>] [-FullName <String>] [-GuiRunOnceCommands <String[]>] [-JobVariab

le <String>] [-Name <String>] [-OperatingSystem <OperatingSystem>] [-OrgNam

e <String>] [-Owner <String>] [-ProductKey <String>] [-PROTipID <Guid>] [-R

unAsynchronously] [-TimeZone <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a guest operating system profile used in

a Virtual Machine Manager environment. Changes made to a guest operating sy

stem profile affect only the guest operating system profile itself. Changes

do not affect any existing virtual machines that were created earlier by u

sing this profile.

Properties that you can change include:

-AdminPasswordCredential

-AnswerFile

-ComputerName

-Description

-FullName

-GuiRunOnceCommands

-JoinDomain

-JoinDomainCredential

-JoinWorkgroup

-Name

-OperatingSystem

-OrgName

-Owner

-ProductKey

-TimeZone

PARAMETERS

-GuestOSProfile [<GuestOSProfile String>]

Specifies a guest operating system profile object.

-JoinDomain <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the do

main to which you want to join a virtual machine. You can use this para

meter to override the existing value on a template or on a guest operat

ing system profile. You can join a VM to a domain only if a virtual net

work adapter is configured for the VM.

-JoinDomainCredential <PSCredential>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the user name and

password of an account with permission to join a virtual machine to the

domain. A limited rights account should be used for joining machines t

o the domain.

-AdminPasswordCredential <PSCredential>

Specifies the password for the local Administrator account. Specifying

a password (on a new or existing template, on a new or existing guest

operating system profile, or on a new virtual machine) overrides any ex

isting Administrator password.

-AnswerFile <Script>

Specifies a script object stored in the VMM library to use as an answer

file. The name of the answer file script depends on the operating syst

em that you want to install on a virtual machine:

ANSWER FILE GUEST OS TO INSTALL ON VM

----------- -------------------------

Sysprep.inf Windows XP, Windows Server 2000, or Windows Server 2003

Unattend.xml Windows Vista or Windows Server 2008

Note: In VMM 2007, this parameter was named SysPrepFile.

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Description <String>

Specifies a description for the specified object.

-FullName <String>

Specifies the name of the person in whose name a virtual machine is reg

istered.

-GuiRunOnceCommands <String[]>

Specifies one or more commands to add to the [GuiRunOnce] section of an

unattended answer file (such as SysPrep.inf or Unattend.xml). Use sing

le quotes around each string enclosed in double quotes.

Example:

-GuiRunOnceCommands '"C:\APF\APFPostSysPrepCopy.cmd PARAMS1"', '"C:\APF

\APFPostSysPrepCopy.cmd PARAMS1"'

For information about how Windows PowerShell uses quotes, type: Get-Hel

p about\_Quoting\_Rules

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-JoinWorkgroup <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the wo

rkgroup to which you want to join a virtual machine. You can use this p

arameter to override the existing value on a template or on a guest ope

rating system profile.

-Name <String>

Specifies the name of a VMM object.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-OrgName <String>

Specifies the name of the organization of the person in whose name a vi

rtual machine is registered.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-ProductKey <String>

Specifies the product key to use for the operating system to be install

ed on a virtual machine. The product key is a 25-digit number that iden

tifies the product license.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-TimeZone <Int32>

Specifies a number (an "index") that identifies a geographical region t

hat shares the same standard time. For a list of time zone indexes, see

"Microsoft Time Zone Index Values" at: http://go.microsoft.com/fwlink/

?LinkId=120935. If no time zone is specified, the default time zone use

d for a virtual machine is the same time zone setting that is on the vi

rtual machine host.

Example: To specify the GMT Standard Time zone, type: -TimeZone 085

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Specify an organization name for an existing guest operating system prof

ile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $OS = Get-OperatingSystem -VMMServer "VMMServer1.contoso.com" | whe

re {$\_.Name -eq "Windows Small Business Server 2003"}

PS C:\> $OSProfile = New-GuestOSProfile -Name "NewProfile1" -ComputerName "

Contoso1" -FullName "Contoso"

PS C:\> Set-GuestOSProfile -GuestOSProfile $OSProfile -OrgName "Contoso"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents a specific operating sys

tem (Windows Small Business Server 2003) and stores the operating system ob

ject in variable $OS.

The third command creates a guest operating system profile named NewProfile

1, specifies that the computer name is Contoso1 and that the full name is C

ontoso, and stores the profile object in variable $OSProfile.

The last command sets Contoso as the organization name for the guest operat

ing system profile created in the second command and specifies the name of

the operating system.

REMARKS

For more information, type: "get-help Set-GuestOSProfile -detailed".

For technical information, type: "get-help Set-GuestOSProfile -full".

## HardDisk

### Copy-HardDisk

SYNOPSIS

Copies a volume of a physical hard disk on a source computer to a Windows-b

ased virtual hard disk file (a .vhd file) on the specified Virtual Machine

Manager host.

SYNTAX

Copy-HardDisk -SourceComputerName <String> -VMHost [<String Host>] [-Creden

tial <PSCredential>] [-DiskSizeAdd <Int32>] [-DriverPath <String>] [-Dynami

c] [-Fixed] [-JobGroup <Guid>] [-JobVariable <String>] [-Offline] [-Owner <

String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Shutdow

n] [-Trigger] [-VMMServer [<String ServerConnection>]] [-VolumeDeviceID <Gu

id>] [<CommonParameters>]

Copy-HardDisk -MachineConfig <MachineConfig> -VMHost [<String Host>] [-Cred

ential <PSCredential>] [-DiskSizeAdd <Int32>] [-DriverPath <String>] [-Dyna

mic] [-Fixed] [-JobGroup <Guid>] [-JobVariable <String>] [-Offline] [-Owner

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Shutd

own] [-Trigger] [-VMMServer [<String ServerConnection>]] [-VolumeDeviceID <

Guid>] [<CommonParameters>]

DETAILED DESCRIPTION

Copies a volume of a physical hard disk on a source computer to a Windows-b

ased virtual hard disk file (a .vhd file) on the specified Virtual Machine

Manager host. If the volume contains an operating system, after you run Cop

y-HardDisk, you must use the New-P2V cmdlet to configure the operating syst

em to run in a virtual environment.

When you used Copy-HardDisk with the -Offline parameter, the computer whose

hard disk is to be copied is first started in the Windows Preinstallation

Environment (Windows PE) and then the volumes are copied.

PARAMETERS

-MachineConfig <MachineConfig>

Specifies a physical machine configuration to use when you convert a ph

ysical machine to a virtual machine. Machine configuration includes inf

ormation about the physical computer's hardware, disks, and operating s

ystem.

-SourceComputerName <String>

Specifies the source computer for a physical-to-virtual machine convers

ion (P2V conversion) performed by VMM.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the source computer name.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-DiskSizeAdd <Int32>

Specifies, in megabytes (MB), the amount of additional disk space to ad

d to a virtual hard disk when performing a physical-to-virtual (P2V) or

virtual-to-virtual (V2V) machine conversion. Volumes located on the vi

rtual hard disk are automatically extended to fill the entire virtual h

ard disk.

-DriverPath <String>

Specifies the path to drivers for any offline physical-to-virtual machi

ne conversion (P2V conversion).

-Dynamic

Specifies that a virtual hard disk can expand dynamically.

-Fixed

Specifies that a virtual hard disk is fixed in size.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Offline

Specifies that the operation is performed offline.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-Shutdown

Specifies that the source server shuts down after a successful physical

-to-virtual machine conversion (P2V conversion).

-Trigger

Starts the execution a job group for a physical-to-virtual (P2V) conver

sion, a virtual-to-virtual (V2V) conversion, or the conversion of a phy

sical hard disk to a virtual hard disk.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VolumeDeviceID <Guid>

Specifies the device ID of the volume to convert in a physical-to-virtu

al machine conversion (P2V conversion).

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Copy a physical hard disk from a source machine to a virtual hard disk f

ile.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> Copy-HardDisk -SourceComputerName "P2VSource01.Contoso.com" -Volume

DeviceID "C" -Credential $Credential -VMHost $VMHost -Path "C:\MyVHDs" -Fix

ed -DiskSizeAdd 1024

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are either a local Administrator account

or a domain account with administrator rights on the computer on which resi

des the physical hard disk that you want to convert to a virtual hard disk.

The second command connects to VMMServer1 in the Contoso.com domain and ret

rieves the server object from the VMM database; the following commands use

this server by default.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in variable $VMHost.

The last command copies and converts the "C:" volume located on the source

computer named P2VSource01 (in the Contoso.com domain) into a new virtual h

ard disk (still named "C:") and places it on VMHost01 at the specified path

(C:\MyVHDs). The Fixed parameter specifies that the .vhd is fixed rather t

han dynamic in format, and the DiskSizeAdd parameter increases the size of

the volume by 1024 MB. As this command is processed, $Credential provides y

our credentials to Copy-HardDisk.

2: Copy a physical hard disk and configure the operating system on that vol

ume to run in a virtual environment.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> $MachineConfig = New-MachineConfig -SourceComputerName "P2VSource.C

ontoso.com" -Credential $Credential

PS C:\> Copy-HardDisk -SourceComputerName "P2VSource02.Contoso.com" -Volume

DeviceID "C" -Credential $Credential -VMHost $VMHost -Path "C:\MyVMs" -Fixe

d -DiskSizeAdd 1024

PS C:\> New-P2V -MachineConfig $MachineConfig -Name "VM01" -VMHost $VMHost

-Path "C:\MyVMs" -MemoryMB 256 -Credential $Credential -RunAsynchronously

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password and stores your credentials in $Credential.

The second command connects to VMMServer1.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in $VMHost.

The fourth command gathers the machine configuration information from the p

hysical source machine called P2VSource02 (in the Contoso.com) and stores t

he machine configuration information in $MachineConfig. As this command is

processed, $Credential provides your credentials to New-MachineConfig.

The fifth command copies and converts the "C:" volume located on the source

computer named P2VSource02 into a new virtual hard disk (still named "C:")

and places it on VMHost01 at the specified path (C:\MyVMs). The Fixed para

meter specifies that the .vhd is fixed rather than dynamic in format, and t

he DiskSizeAdd parameter increases the size of the volume by 1024 MB. As th

is command is processed, $Credential provides your credentials to Copy-Hard

Disk.

The last command uses the virtual hard disk (located at C:\MyVMs on VMHost0

1) that was created in the preceding step and the machine configuration sto

red in $MachineConfig to create a new virtual machine called VM01. The New-

P2V cmdlet automatically configures the operating system on the virtual har

d disk to run in a virtual environment; it uses the -MemoryMB parameter to

assign 256 MB of memory on the host for use by the virtual machine; and it

uses the RunAsynchronously parameter to return control to the shell immedia

tely (before the command completes).

REMARKS

For more information, type: "get-help Copy-HardDisk -detailed".

For technical information, type: "get-help Copy-HardDisk -full".

## HardwareProfile

### Get-HardwareProfile

SYNOPSIS

Gets hardware profile objects from the Virtual Machine Manager library.

SYNTAX

Get-HardwareProfile [-All] [-VMMServer [<String ServerConnection>]] [<Commo

nParameters>]

Get-HardwareProfile [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [

<CommonParameters>]

DETAILED DESCRIPTION

Gets from the Virtual Machine Manager library one or more objects that repr

esent hardware profiles. You can use a hardware profile repeatedly to creat

e new virtual machines or virtual machine templates.

For more information about hardware profiles, type:

Get-Help New-HardwareProfile -detailed

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all hardware profiles from the library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-HardwareProfile

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets from the VMM library all objects that represent har

dware profiles and displays information about these profiles to the user.

2: Get a specific hardware profile from the library.

C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

C:\> Get-HardwareProfile | where { $\_.Name -eq "NewProfile1" }

The first command connects to VMMServer1.

The second command gets the object that represent the hardware profile name

d NewProfile1 and displays information about this hardware profile to the u

ser.

REMARKS

For more information, type: "get-help Get-HardwareProfile -detailed".

For technical information, type: "get-help Get-HardwareProfile -full".

### New-HardwareProfile

SYNOPSIS

Creates a hardware profile for use in Virtual Machine Manager.

SYNTAX

New-HardwareProfile [-Name] <String> [-BootOrder <BootDevice[]>] [-CPUCount

<Int32>] [-CPUMax <Int32>] [-CPUReserve <Int32>] [-CPUType [<ProcessorType

String>]] [-Description <String>] [-DiskIO <Int32>] [-ExpectedCPUUtilizati

on <Int32>] [-HardwareProfile <HardwareProfile>] [-HighlyAvailable <Boolean

>] [-JobGroup <Guid>] [-JobVariable <String>] [-LimitCPUFunctionality <Bool

ean>] [-MemoryMB <Int32>] [-NetworkUtilization <Int32>] [-NumLock] [-Owner

<String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsynchronously]

[-VMMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a hardware profile for use in Virtual Machine Manager that stores h

ardware configuration information. You can create a standalone hardware pro

file or customize a template or virtual machine to include hardware profile

settings. The New-HardwareProfile cmdlet stores the new hardware profile o

bject in the Virtual Machine Manager library.

You can create a hardware profile based on defaults, based on an existing h

ardware profile, or you can customize a hardware profile as you create the

profile. If you specify no parameters except the Name parameter (which is r

equired), Virtual Machine Manager creates a default hardware profile object

.

Hardware profile settingsfor a virtual machine include:

\* Boot order settings in the BIOS that specify the device startup order

for a virtual machine.

NOTE: The boot order setting is available only for virtual machines

on a Hyper-V host.

\* CPU settings for a virtual machine.

\* Memory available on a virtual machine.

\* A virtual floppy drive.

\* Two virtual COM ports (COM1 and COM2).

\* A built-in virtual IDE device.

\* One or more optional virtual SCSI adapters.

\* One or more optional virtual network adapters that you can attach to

an internal network or to an external network. A virtual network

adapter can be emulated or synthetic.

\* The priority assigned to a specific virtual machine for using the

host's CPU resources in comparison to the use of the host's CPU by

other virtual machines deployed on the same host. CPU priorities are

determined by the virtualization software.

\* Whether (optionally) a virtual machine created from this profile will

be highly available. A highly available virtual machine is a virtual

machine that can be placed only on a host that is part of a host

cluster.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-BootOrder <BootDevice[]>

Specifies the order of devices that a virtual machine on a Hyper-V host

uses to start up.

Valid values: CD, IDEHardDrive, PXEBoot, or Floppy.

Example: -BootOrder PXEBoot,IDEHardDrive,CD,Floppy

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUMax <Int32>

Specifies the highest percentage of the total resources of a single CPU

on the host that can be used by a specific virtual machine at any give

n time.

Example: -CPUMax 80 (to specify 80 per cent)

-CPUReserve <Int32>

Specifies the minimum percentage of the resources of a single CPU on th

e host to allocate to a virtual machine. The percentage of CPU capacity

that is available to the virtual machine is never less than this perce

ntage.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-Description <String>

Specifies a description for the specified object.

-DiskIO <Int32>

Specifies the number of disk input/output operations per second (IOPS)

on the host that can be used by a specific virtual machine.

Example: -DiskIO 1500 (to specify 1500 IOPS).

-ExpectedCPUUtilization <Int32>

Specifies (as a percentage) the amount of CPU on the host that you expe

ct this virtual machine to use. This value is used only when VMM determ

ines a suitable host for the virtual machine.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-HighlyAvailable <Boolean>

Specifies that a virtual machine will be placed on a Hyper-V host that

is part of a host cluster. Configure this setting on a virtual machine,

or on a template or hardware profile that will be used to create virtu

al machines.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LimitCPUFunctionality <Boolean>

Enables running an older operating system (such as Windows NT 4.0) on a

virtual machine deployed on a Hyper-V host or on a VMware ESX host by

providing only limited CPU functionality for the virtual machine.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-NetworkUtilization <Int32>

Specifies, in megabits per second (Mb/s), the amount of bandwidth on th

e host's network that can be used by a specific virtual machine.

Example: -NetworkUtilization 10 (to specify 10 Mb/s)

-NumLock

Enables the BIOS value for NumLock on a virtual machine (or on a templa

te or hardware profile that is used to create virtual machines) on a Hy

per-V host. This parameter does not apply to virtual machines on Virtua

l Server hosts or on VMware ESX hosts.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a default hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> New-HardwareProfile -Name "NewProfile1"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command creates a default hardware profile named NewProfile1.

2: Create a hardware profile that sets boot order, CPU, and memory.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com" | Out-Null

PS C:\> New-HardwareProfile -Name "LargeVM" -BootOrder PXEBoot,CD,Floppy,ID

EHardDrive -MemoryMB 1024 -CPUCount 4

The first command connects to VMMServer1 and then uses the Out-Null cmdlet

to redirect the output to $Null instead of sending the output to the consol

e.

The second command creates a new hardware profile, names it "LargeVM", sets

"PXEBoot" as the first entry in the BIOS boot order, specifies 1024 MB of

memory, and specifies that a virtual machine created by using this hardware

profile will have four processors.

3: Clone and then modify an existing hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = New-HardwareProfile -Name "NewProfile3" -MemoryMB 512

PS C:\> New-HardwareProfile -Name "NewProfile4" -HardwareProfile $HWProfile

-RelativeWeight 100

The first command connects to VMMServer1.

The second command creates a hardware profile called NewProfile3, specifies

that the amount of memory on the host that a virtual machine (created by u

sing this hardware profile) can use is 512 MB, and stores the profile objec

t in variable $HWProfile.

The last command creates another hardware profile, called NewProfile4, whic

h is based on NewProfile3 but modifies the value for relative weight. All o

ther settings in NewProfile4 are identical to those in NewProfile3.

4: Create a hardware profile that contains a network adapter, a SCSI adapte

r, and a DVD drive.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $JobGroupId = [Guid]::NewGuid().ToString()

PS C:\> New-VirtualNetworkAdapter -JobGroup $JobGroupID -PhysicalAddressTyp

e Dynamic -VirtualNetwork "Internal Network"

PS C:\> New-VirtualSCSIAdapter -JobGroup $JobGroupID -AdapterID 6 -Shared $

FALSE

PS C:\> New-VirtualDVDDrive -JobGroup $JobGroupID -Bus 1 -LUN 0

PS C:\> New-HardwareProfile -Name "NewProfile2" -Owner "Contoso\Nicholas" -

Description "Temporary Hardware Config used to create a VM/Template" -Memor

yMB 512 -JobGroup $JobGroupID

The first command connects to VMMServer1.

The second command generates a globally unique identifier (GUID) and stores

the GUID string in variable $JobGroupID. The job group ID functions as an

identifier that groups subsequent commands that include this identifier int

o a single job group.

The third command will create a virtual network adapter but uses the JobGro

up parameter to specify that the network adapter is not created until just

before the New-HardwareProfile cmdlet (in the last command) runs. This comm

and sets the physical address type (MAC address type) to dynamic and specif

ies that the new virtual network adapter will connect to a virtual network

called "Internal Network."

The fourth command will create a virtual SCSI adapter but uses the JobGroup

parameter to specify that the SCSI adapter is not created until just befor

e the New-HardwareProfile cmdlet (in the last command) runs. This command s

ets the adapter ID to 6, and it sets the Shared parameter to $FALSE so that

the adapter will not be shared (as it would have had to be if you wanted t

o use the adapter in guest clustering).

The fifth command will create a virtual DVD drive but uses the JobGroup par

ameter to specify that the DVD drive is not created until just before the N

ew-HardwareProfile cmdlet (in the last command) runs. Specifying Bus 1 and

LUN 0 attaches the virtual DVD drive to Secondary Channel (0) on the IDE bu

s.

The sixth command creates a hardware profile named NewProfile2, sets the ow

ner to Nicholas (whose account is in the Contoso.com domain), specifies a d

escription, and specifies that the amount of memory on the host that a virt

ual machine (created by using this hardware profile) will use is 512 MB. Be

fore the New-HardwareProfile cmdlet creates the hardware profile, the JobGr

oup parameter in this final command executes all of the preceding cmdlets t

hat specify the same JobGroup GUID. When New-VirtualNetworkAdapter, New-Vir

tualSCSIAdapter, and New-VirtualDVDDrive execute, the resulting objects tha

t are created will be automatically associated with the new hardware profil

e.

5: Create a hardware profile and add it to a new template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com" | Out-Null

PS C:\> $HWProfile = New-HardwareProfile -Name "LargeHAVM" -CPUCount 4 -Mem

oryMB 64000 -CPUMax 100 -Owner "User1@Contoso" -HighlyAvailable $TRUE

PS C:\> $VHD = Get-VirtualHardDisk | where { $\_.Name -eq “VHD01.vhd” -and

$\_.LibraryServer.Name -eq "FileServer01.Contoso.com” }

PS C:\> $OS = Get-OperatingSystem | where {$\_.Name -eq "64-bit edition of W

indows Server 2008 Datacenter"}

PS C:\> New-Template -Name "LargeHALOBTemplate" -HardwareProfile $HWProfile

-OperatingSystem $OS -VirtualHardDisk $VHD -NoCustomization

The first command connects to VMMServer1 and then uses the Out-Null cmdlet

to redirect the output to $Null instead of sending the output to the consol

e.

The second command creates a new hardware profile, names it "LargeHAVM"; sp

ecifies that it contains four processors and that the highest percentage of

the total resources of a single CPU on a host that can be used by a virtua

l machine is 100 percent; assigns 64 GB of RAM and an owner; sets the Highl

yAvailable flag $TRUE; and stores the new hardware profile object in $HWPro

file. The HighlyAvailable flag specifies that a virtual machine created by

using this hardware profile (either directly or through a template) will be

placed on a host that is a node of a host cluster.

The third command gets a virtual hard disk object by name from the library

and stores the virtual hard disk object in $VHD.

The fourth command gets an operating system object by name and stores the o

perating system object in $OS.

The last command creates a new template, names it "LargeHALOBTemplate", and

specifies that it use the operating system, hardware profile, and virtual

hard disk retrieved or created in the preceding commands, without any custo

mization to the operating system.

REMARKS

For more information, type: "get-help New-HardwareProfile -detailed".

For technical information, type: "get-help New-HardwareProfile -full".

### Remove-HardwareProfile

SYNOPSIS

Removes a hardware profile object from Virtual Machine Manager.

SYNTAX

Remove-HardwareProfile [-HardwareProfile] <HardwareProfile> [-Confirm] [-Jo

bVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParamet

ers>]

DETAILED DESCRIPTION

Removes one or more objects that represent hardware profiles from the Virtu

al Machine Manager library.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Confirm

Prompts for confirmation before running the command.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific hardware profile from the library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "NewProfile1

"}

PS C:\> Remove-HardwareProfile -HardwareProfile $HWProfile

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the hardware profile nam

ed NewProfile1 from the VMM library and stores the hardware profile object

in variable $HWProfile.

The second command removes New-Profile1 from the library.

2: Remove all hardware profiles without being prompted to confirm each dele

tion.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-HardwareProfile | Remove-HardwareProfile -Confirm

The first command connects to VMMServer1.

The second command gets all hardware profile objects and passes each profil

e object to the Remove-HardwareProfile cmdlet, which removes each hardware

profile. The Confirm parameter prompts you to confirm whether you want to d

elete these hardware profile objects.

REMARKS

For more information, type: "get-help Remove-HardwareProfile -detailed".

For technical information, type: "get-help Remove-HardwareProfile -full".

### Set-HardwareProfile

SYNOPSIS

Changes the properties of a hardware profile used in Virtual Machine Manage

r.

SYNTAX

Set-HardwareProfile [-HardwareProfile] <HardwareProfile> [-BootOrder <BootD

evice[]>] [-CPUCount <Int32>] [-CPUMax <Int32>] [-CPUReserve <Int32>] [-CPU

Type [<ProcessorType String>]] [-Description <String>] [-DiskIO <Int32>] [-

ExpectedCPUUtilization <Int32>] [-HighlyAvailable <Boolean>] [-JobGroup <Gu

id>] [-JobVariable <String>] [-LimitCPUFunctionality <Boolean>] [-MemoryMB

<Int32>] [-Name <String>] [-NetworkUtilization <Int32>] [-NumLock] [-Owner

<String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsynchronously]

[<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a hardware profile object used in a Virtu

al Machine Manager environment. Properties that you can change include sett

ings for boot order, CPU settings, the amount memory on the host that is as

signed to a virtual machine, and other options.

If you want to change the properties of a virtual floppy drive, virtual DVD

drive, virtual COM port, virtual network adapter, or virtual SCSI adapter

associated with a specific hardware profile, you can use Set-VirtualFloppyD

rive, Set-VirtualDVDDrive, Set-VirtualCOMPort, Set-VirtualNetworkAdapter, o

r Set-VirtualSCSIAdapter, respectively.

Changes made to a hardware profile affect only the hardware profile itself.

Changes do not affect any existing virtual machines that were created earl

ier by using this profile.

For more information about hardware profiles, type:

Get-Help New-HardwareProfile -detailed

PARAMETERS

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-BootOrder <BootDevice[]>

Specifies the order of devices that a virtual machine on a Hyper-V host

uses to start up.

Valid values: CD, IDEHardDrive, PXEBoot, or Floppy.

Example: -BootOrder PXEBoot,IDEHardDrive,CD,Floppy

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUMax <Int32>

Specifies the highest percentage of the total resources of a single CPU

on the host that can be used by a specific virtual machine at any give

n time.

Example: -CPUMax 80 (to specify 80 per cent)

-CPUReserve <Int32>

Specifies the minimum percentage of the resources of a single CPU on th

e host to allocate to a virtual machine. The percentage of CPU capacity

that is available to the virtual machine is never less than this perce

ntage.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-Description <String>

Specifies a description for the specified object.

-DiskIO <Int32>

Specifies the number of disk input/output operations per second (IOPS)

on the host that can be used by a specific virtual machine.

Example: -DiskIO 1500 (to specify 1500 IOPS).

-ExpectedCPUUtilization <Int32>

Specifies (as a percentage) the amount of CPU on the host that you expe

ct this virtual machine to use. This value is used only when VMM determ

ines a suitable host for the virtual machine.

-HighlyAvailable <Boolean>

Specifies that a virtual machine will be placed on a Hyper-V host that

is part of a host cluster. Configure this setting on a virtual machine,

or on a template or hardware profile that will be used to create virtu

al machines.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LimitCPUFunctionality <Boolean>

Enables running an older operating system (such as Windows NT 4.0) on a

virtual machine deployed on a Hyper-V host or on a VMware ESX host by

providing only limited CPU functionality for the virtual machine.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-Name <String>

Specifies the name of a VMM object.

-NetworkUtilization <Int32>

Specifies, in megabits per second (Mb/s), the amount of bandwidth on th

e host's network that can be used by a specific virtual machine.

Example: -NetworkUtilization 10 (to specify 10 Mb/s)

-NumLock

Enables the BIOS value for NumLock on a virtual machine (or on a templa

te or hardware profile that is used to create virtual machines) on a Hy

per-V host. This parameter does not apply to virtual machines on Virtua

l Server hosts or on VMware ESX hosts.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Specify an amount of memory for an existing hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = New-HardwareProfile -Name "NewProfile1" -MemoryMB 512

PS C:\> Set-HardwareProfile -HardwareProfile $HWProfile -MemoryMB 1024

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command creates a hardware profile named NewProfile1; specifies

that the amount of memory on a host that a virtual machine (created by usin

g this hardware profile) will use is 512 MB; and stores the profile object

in variable $HWProfile.

The last command changes the memory value for NewProfile1 to 1024 MB.

2: Specify a new owner for multiple hardware profiles.

PS C:\> Get-VMMServer "VMMServer01.contoso.com"

PS C:\> $Profiles = Get-HardwareProfile | where {$\_.Name -match "Profile"}

PS C:\> foreach ($Profile in $Profiles) {Set-HardwareProfile -HardwareProfi

le $Profile -Owner "Contoso\<NewOwnerUserName>"}

The first command connects to VMMServer1.

The second command gets a list of hardware profile objects that match the s

earch criteria and stores the hardware profile objects in $Profiles (an obj

ect array).

The third command uses a foreach statement to specify a new owner for each

of the profiles in the array.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

3: Specify a new boot order for multiple hardware profiles.

PS C:\> Get-VMMServer "VMMServer01.Contoso.com"

PS C:\> $Profiles = @(Get-HardwareProfile | where {$\_.Name -match "RemoteBo

otProfile"})

PS C:\> foreach ($Profile in $Profiles) {Set-HardwareProfile -HardwareProfi

le $Profile -BootOrder PXEBoot,CD,IDEHardDrive,Floppy}

The first command connects to VMMServer1.

The second command gets all objects that represent hardware profiles in the

library that match the search criteria (the profile name contains the stri

ng "RemoteBootProfile") and stores the hardware profile objects in $Profile

s. Using the "@" symbol and parentheses ensures that the command stores the

results in an array (in case the command returns a single object or a null

value).

The last command uses a foreach statement to specify a new boot order for e

ach hardware profile object in the array.

4: Search for hardware profiles with a specific configuration and append te

xt to the description field.

PS C:\> Get-VMMServer "VMMServer01.contoso.com"

PS C:\> $Profiles = @(Get-HardwareProfile | where {$\_.CPUCount -eq 4})

PS C:\> foreach ($Profile in $Profiles) {$Text = $Profile.Description; Set-

HardwareProfile -HardwareProfile $Profile -Description $Text" (Contains 4 P

rocessors)"}

The first command connects to VMMServer1.

The second command gets all objects that represent hardware profiles that m

atch the search criteria (CPU Count is equal to 4) and stores the hardware

profile objects in $Profiles (an object array).

The last command uses a foreach statement to iterate through each profile o

bject in the array. For each profile, the Description text is stored to a v

ariable ($Text), and then the Set-HardwareProfile cmdlet uses the Descripti

on parameter to append "(Contains 4 Processors)" to the contents of each in

stance of $Text.

REMARKS

For more information, type: "get-help Set-HardwareProfile -detailed".

For technical information, type: "get-help Set-HardwareProfile -full".

## ISO

### Get-ISO

SYNOPSIS

Gets ISO objects from the Virtual Machine Manager library.

SYNTAX

Get-ISO [-All] [-VMMServer [<String ServerConnection>]] [<CommonParameters>

]

Get-ISO [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [<CommonParam

eters>]

DETAILED DESCRIPTION

Gets objects that represent ISO files from the Virtual Machine Manager libr

ary. The ISO file that an ISO object represents is stored in a library shar

e on a library server.

In Virtual Machine Manager, some typical uses of an ISO file include:

\* Storing an operating system ISO in the library that you can use

later to install that operating system on a new or existing virtual

machine deployed on a host.

\* Storing application software, such as a Microsoft Office ISO, in

the library, so that you can install it later on a virtual machine

deployed on a host.

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all ISOs on all VMM library servers.

PS C:\> Get-ISO -VMMServer VMMServer1.Contoso.com

Gets from the VMM library on VMMServer1 all objects that represent ISO file

s and displays information about these ISOs to the user. The ISO files that

the retrieved objects represent are stored in library shares on library se

rvers.

2: Get all ISOs on a specific VMM library server.

PS C:\> Get-ISO -VMMServer VMMServer1.Contoso.com | where { $\_.LibraryServe

r.Name -eq "FileServer01.Contoso.com" }

Gets from the library on VMMServer1 all objects that represent ISO files st

ored on library server FileServer01 and displays information about these IS

Os to the user.

3: Get all ISOs with a specific string in the file name on any VMM library

server.

PS C:\> Get-ISO -VMMServer VMMServer1.Contoso.com | where { $\_.Name -match

"WinISO" }

Gets all objects on any VMM library server managed by VMMServer1 that repre

sent ISO files that contain "WinISO" in the file name and displays informat

ion about these ISOs to the user.

NOTE: By default, the name of an ISO object in the library is the same name

as the name of the actual ISO file stored in the file system on the librar

y server.

REMARKS

For more information, type: "get-help Get-ISO -detailed".

For technical information, type: "get-help Get-ISO -full".

### Remove-ISO

SYNOPSIS

Removes an ISO object from the library in Virtual Machine Manager.

SYNTAX

Remove-ISO [-ISO] <ISO> [-Confirm] [-Force] [-JobVariable <String>] [-PROTi

pID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents an ISO image from the Virtual Machine Man

ager library and deletes the corresponding ISO file on the library server.

If the ISO is attached to a virtual machine, template, or hardware profile

(and if you do not use the Force parameter), VMM lists the container that c

ontains the ISO and prompts you to confirm that you want to remove the ISO:

\* If you reply Yes, VMM removes the association between the ISO and

the container to which it is attached, and then deletes the ISO

object from VMM.

\* If you reply No, the operation is cancelled.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-ISO <ISO>

Specifies an ISO object.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove an ISO object and delete the corresponding .iso file.

PS C:\> $Iso = Get-ISO -VMMServer VMMServer1.Contoso.com | where { $\_.Name

-eq "VMAdditions.iso" -and $\_.LibraryServer.Name -eq "FileServer01.Contoso.

com" }

PS C:\> Remove-ISO -ISO $Iso

The first command gets the object that represents the ISO file named VMAddi

tions (stored in the file system on library server Fileserver01) from VMMSe

rver1 and stores the ISO object in variable $Iso.

The second command removes the ISO object from the library and deletes the

corresponding .iso file from the file system on the library server.

NOTE: This example assumes that the VMM environment includes at least one V

irtual Server host and, therefore, that Virtual Machine Addtions (VMAdditio

ns.iso) is available for installation on virtual machines deployed on a Vir

tual Server host.

2: Remove multiple ISO objects from the library.

PS C:\> $Isos = Get-ISO -VMMServer VMMServer1.Contoso.com | where { $\_.Name

-match “VMAdditions.iso” }

PS C:\> $Isos | Remove-ISO

The first command gets all objects that represent ISO files whose name incl

udes the string “VMAdditions” and stores these ISO objects in $Isos.

The second command passes each ISO object in $Isos to the Remove-ISO cmdlet

, which removes each ISO object from the library and deletes the correspond

ing .iso file from the file system on the library server.

REMARKS

For more information, type: "get-help Remove-ISO -detailed".

For technical information, type: "get-help Remove-ISO -full".

### Set-ISO

SYNOPSIS

Changes properties of an ISO object used in Virtual Machine Manager.

SYNTAX

Set-ISO [-ISO] <ISO> [-Description <String>] [-Enabled <Boolean>] [-JobVari

able <String>] [-Name <String>] [-Owner <String>] [-PROTipID <Guid>] [-RunA

synchronously] [-SharePath <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of an ISO object used in a Virtual Machine M

anager environment. Properties that you can change include:

- Description

- Enabled

- Name

- Owner

- SharePath

PARAMETERS

-ISO <ISO>

Specifies an ISO object.

-Description <String>

Specifies a description for the specified object.

-Enabled <Boolean>

Enables a library object (when set to TRUE) or disables a library objec

t (when set to FALSE). For example, if you want to upgrade software on

a virtual machine template, you can disable the template object in the

VMM library to temporarily prevent users from using that object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the owner of an ISO file.

PS C:\> $Iso = @(Get-ISO -VMMServer VMMServer1.Contoso.com | where { $\_.Nam

e -eq "VMAdditions.iso" -and $\_.LibraryServer.Name -eq "FileServer01.Contos

o.com" } )

PS C:\> Set-ISO -ISO $Iso[0] -Owner “Contoso\HanyingFeng”

The first command gets from the library on VMMServer1 the object that repre

sents the ISO file named VMAdditions.iso (on library server FileServer01) a

nd stores the ISO object in variable $Iso. Using the "@" symbol and parenth

eses ensures that the cmdlet stores the returned object in an array (in cas

e the command returns a single object or a null value).

The second command changes the owner of the first ISO object stored in $Iso

to Hanying Feng, a member of the Contoso.com domain.

NOTE: In VMM 2008, by default, the name of an ISO object in the VMM library

is the same name (including the file extension) as the name of the actual

ISO file on the library server.

2: Specify an owner for all ISO objects with an "Unknown" owner.

PS C:\> Get-ISO -VMMServer "VMMServer1.Contoso.com" | where {$\_.Owner -eq "

Unknown"} | Set-ISO -Owner "Contoso\ChrisGray"

Gets all ISO objects from the VMM library, selects only those ISO objects w

hose owner is "Unknown", and specifies an owner for each ISO object.

3: Disable an ISO object stored in the VMM library.

PS C:\> $Iso = Get-ISO -VMMServer VMMServer1.Contoso.com | where { $\_.Name

-eq "VMAdditions.iso" -and $\_.LibraryServer.Name -eq "FileServer01.Contoso.

com" }

PS C:\> Set-ISO -ISO $Iso -Enabled $FALSE

The first command gets the object that represents the ISO image named VMAdd

itions.iso (whose file is stored on the library server named FileServer01)

from VMMServer1 and stores the ISO object in $Iso.

The second command disables the ISO represented by the $Iso variable.

REMARKS

For more information, type: "get-help Set-ISO -detailed".

For technical information, type: "get-help Set-ISO -full".

## Job

### Get-Job

SYNOPSIS

Gets Virtual Machine Manager job objects on the Virtual Machine Manager ser

ver.

SYNTAX

Get-Job [[-Name] <String>] [-All] [-Full <Boolean>] [-ID <Guid>] [-Job [<St

ring Task>]] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent Virtual Machine Manager jobs on the

Virtual Machine Manager server. A job is a series of steps that are perfor

med sequentially to complete an action in the Virtual Machine Manager envir

onment. You can retrieve job objects based on specified criteria.

In Virtual Machine Manager, you can group a series of jobs and run them tog

ether as a set. For example, a complex action in Virtual Machine Manager, s

uch as creating a template, might incorporate a series of jobs, known as a

job group. For examples that show you how to use job groups, type:

Get-Help New-Template -example

Get-Help NewHardwareProfile -example

Get-Help New-VirtualDiskDrive -example

Get-Help VirtualDVDDrive -example

Get-Help New-VM -example

Get-Help Set-VirtualCOMPort -example

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-Full <Boolean>

Specifies that the cmdlet returns the job object with an audit record.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-Job [<String Task>]

Specifies a VMM job object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all jobs currently running on a VMM server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MyJobs = Get-Job | where { $\_.Status -eq "Running" } | Format-List

-property Name, ID, Status

PS C:\> $MyJobs

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command performs the following operations:

\* Gets from the VMM database all objects that represent

VMM jobs.

\* Passes these job objects to the "where" filter, which selects

only those jobs that are currently running on VM01.

\* Passes each job object for these running jobs to the Format-List

cmdlet, which stores the name, ID number, and Status properties

for each job in $MyJobs.

The last command displays the information stored in $MyJobs to the user.

2: Get information about the .NET type, methods, and properties of VMM job

objects.

PS C:> Get-Job -VMMServer VMMServer1.Contoso.com | Get-Member

Gets job objects from VMMServer1 and passes each job object to the Get-Memb

er cmdlet, which displays the .NET type for a job object:

TypeName: Microsoft.SystemCenter.VirtualMachineManager.Task

The command also displays a list of methods and properties that are associa

ted with a VMM job object.

REMARKS

For more information, type: "get-help Get-Job -detailed".

For technical information, type: "get-help Get-Job -full".

### Restart-Job

SYNOPSIS

Restarts a failed or canceled Virtual Machine Manager job.

SYNTAX

Restart-Job [-Job] [<String Task>] [-Credential <PSCredential>] [<CommonPar

ameters>]

DETAILED DESCRIPTION

Restarts one or more Virtual Machine Manager jobs that have failed or that

have been canceled by a user. Jobs that are currently running must be cance

led before they can be restarted. All restarted jobs start from the last kn

own good checkpoint before a failure or a cancellation (some jobs have only

a single checkpoint).

Restarting a job displays the object properties of the job to the user and

shows the "Status" property as “Running”.

PARAMETERS

-Job [<String Task>]

Specifies a VMM job object.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Restart all jobs that were cancelled on any virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-Job | where { $\_.ResultName -eq "VM01" -and $\_.Status -eq "Canc

eled" } | Restart-Job

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command performs the following operations:

\* Gets from the VMM database all objects that represent

VMM jobs.

\* Passes each job object to the "where" filter to select from

the job results only jobs on the virtual machine named

VM01 that have been cancelled.

\* Passes the cancelled job objects on VM01 to the Restart-Job

cmdlet, which restarts each cancelled job.

2: Restart a specific failed job.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-Job | where { $\_.ID -eq "1234-1234-1234-1234" } | Restart-Job

The first command connects to VMMServer1.

The second command gets all objects that represent VMM jobs, selects only t

he job with the ID 1234-1234-1234-1234, and restarts that job.

REMARKS

For more information, type: "get-help Restart-Job -detailed".

For technical information, type: "get-help Restart-Job -full".

### Stop-Job

SYNOPSIS

Stops running jobs in Virtual Machine Manager.

SYNTAX

Stop-Job [-Job] [<String Task>] [<CommonParameters>]

DETAILED DESCRIPTION

Stops one or more Virtual Machine Manager jobs that are running, and return

s the object for each job in a stopped state. If the Virtual Machine Manage

r job is not currently running, this cmdlet has no effect.

PARAMETERS

-Job [<String Task>]

Specifies a VMM job object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Stop all currently running jobs.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Job = Get-Job | where { $\_.Status -eq "Running" }

PS C:\> $Job | Stop-Job

The first command connects to VMMServer1 and retrieves the server object fr

om the VMM database. The following commands use this server by default.

The second command gets all objects that represent VMM jobs, passes each jo

b object to the "where" filter to select only the objects for jobs that are

currently running, and stores these job objects in variable $Job (an objec

t array).

The last command passes each job object in $Job to the Stop-Job cmdlet, whi

ch stops each running job.

2: Stop a specific running job asynchronously.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Job = Get-Job | where { $\_.ResultName -eq "VM01" -and $\_.ID -eq "1

234-1234-1234-1234" }

PS C:\> Stop-Job -Job $Job

The first command connects to VMMServer1.

The second command gets all objects that represent VMM jobs and, from the j

ob results, selects only the job on VM01 identified by job ID 1234-1234-123

4-1234, and stores this job object in $Job.

The last command stops the job and returns the stopped job object to the us

er.

REMARKS

For more information, type: "get-help Stop-Job -detailed".

For technical information, type: "get-help Stop-Job -full".

## LibraryRating

### Get-LibraryRating

SYNOPSIS

Calculates the placement rating of virtual machine libraries managed by Vir

tual Machine Manager to determine whether a SAN transfer can be used to tra

nsfer a virtual machine from a host to the library.

SYNTAX

Get-LibraryRating -LibraryServer [<LibraryServer LibraryServer[]>] [-VM [<S

tring VM>]] [<CommonParameters>]

DETAILED DESCRIPTION

Calculates the placement rating of virtual machine libraries managed by Vir

tual Machine Manager. Specifically, this rating indicates whether Virtual M

achine Manager can use a SAN transfer to transfer a particular virtual mach

ine from a host server to a library server. If a SAN transfer is not possib

le, you can use a LAN transfer to store the virtual machine in the library.

For information about how to store a virtual machine in the Virtual Machine

Manager library, type: Get-Help Store-VM -detailed

PARAMETERS

-LibraryServer [<LibraryServer LibraryServer[]>]

Specifies a VMM library server object or an array of library server obj

ects.

-VM [<String VM>]

Specifies a virtual machine object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Determine whether you can use a SAN transfer to store a VM on the specif

ied library server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $LibServ = Get-LibraryServer -ComputerName "LibraryServer01.Contoso

.com"

PS C:\> $LibRate = Get-LibraryRating -LibraryServer $LibServ -VM $VM

PS C:\> $LibRate

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command gets the object that represents the virtual machine name

d VM01 and stores the virtual machine object in variable $VM.

The third command gets the object that represents the library server named

LibraryServer01 and stores the library server object in variable $LibServ.

The fourth command returns the placement rating for LibraryServer01 (which

indicates whether VMM can use a SAN transfer to transfer VM01 to LibrarySer

ver01)and stores the rating object in variable $LibRate.

The last command displays the rating for LibraryServer01 to the user.

NOTE: The Get-LibraryRating cmdlet does not return a star rating (as does t

he Get-VMHostRating cmdlet for hosts), but indicates only whether it is pos

sible to use a SAN transfer to transfer a virtual machine currently deploye

d on a host to a specific library server. If a SAN transfer is not possible

, you can use a LAN transfer to store the virtual machine in the library.

2: Get placement ratings to place a VM on each available library server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $LibServ = Get-LibraryServer

PS C:\> $LibRating = Get-LibraryRating -LibraryServer $LibServ -VM $VM

PS C:\> $LibRating

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM01 and stores the virtual machine object in variable $VM.

The third command gets all objects that represent library servers managed b

y VMM and stores the library server objects in $LibServ.

The fourth command returns the placement rating for each library server obj

ect in $LibServ (which indicates whether VMM can use a SAN transfer to tran

sfer VM01 to each of the library servers) and stores the rating for each li

brary server object in $LibRating (an object array).

The last command displays the rating information to the user.

NOTE: The Get-LibraryRating cmdlet does not return a star rating (as does t

he Get-VMHostRating cmdlet for hosts), but indicates only whether it is pos

sible to use a SAN transfer to transfer a VM currently deployed on a host t

o a specific library server. If a SAN transfer is not possible, you can use

a LAN transfer to store the VM in the library.

REMARKS

For more information, type: "get-help Get-LibraryRating -detailed".

For technical information, type: "get-help Get-LibraryRating -full".

## LibraryServer

### Add-LibraryServer

SYNOPSIS

Adds a computer as a library server to Virtual Machine Manager.

SYNTAX

Add-LibraryServer [-ComputerName] <String> -Credential <PSCredential> [-Des

cription <String>] [-JobGroup <Guid>] [-JobVariable <String>] [-LibraryGrou

p <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String Se

rverConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Adds one or more computers as library servers to the Virtual Machine Manage

r database. For a computer to be a library server, it must be in the same d

omain as, or in a trusted domain with, the Virtual Machine Manager server a

nd must have one of the following operating systems installed:

\* Windows Server 2008

\* Windows Server 2003 with Service Pack 1 (SP1) or later

When you add a server as a library server to Virtual Machine Manager, Virtu

al Machine Manager automatically installs the Virtual Machine Manager agent

software on that server.

The Virtual Machine Manager library is made up of two primary components:

\* LIBRARY. The portion of the Virtual Machine Manager database that stores

objects that represent all library resources.

\* LIBRARY RESOURCE FILES. Files that are stored in library shares on one

or more physical library servers. Library resources can be distributed

across multiple physical library servers. Some library objects have

files and others do not.

Virtual Machine Manager library resources include virtual machine templates

, hardware profiles, guest operating system profiles, virtual hard disks (W

indows-based .vhd files or VMware-based .vmdk files), virtual floppy disks

(Windows-based .vfd files or VMware-based .flp files), ISO images (.iso fil

es), and scripts. In addition, you can store virtual machines in the librar

y that, currently, you do not want to deploy on a host.

Of these resources, templates, hardware profiles, and guest operating syste

m profiles are represented only by objects stored in the library. The other

resources are files stored in the file system on library servers and objec

ts that correspond to those files stored in the library.

For more information about the VMM library, type:

Get-Help about\_VMM\_2008\_Library\_Enhancements

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-Description <String>

Specifies a description for the specified object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryGroup <String>

Specifies the group to which a VMM library server belongs. You can use

a library group to associate library servers with particular host group

s or to associate library servers with one another.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a library server object to the VMM database.

PS C:\> Add-LibraryServer -VMMServer VMMServer1.Contoso.com -ComputerName "

FileServer01.Contoso.com"

Adds a library server object that represents FileServer01 to the library in

the VMM database provided by VMMServer1. Both servers are in the Contoso.c

om domain.

When a dialog box appears prompting you for credentials to perform this ope

ration, type the user name and password for either a local Administrator ac

count or a domain account with administrator rights on the library server.

2: Add a highly available file server with two nodes as a library server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Credential = Get-Credential

PS C:\> $Nodes = "LibraryNode01.Contoso.com","LibraryNode02.Contoso.com"

PS C:\> foreach ($Node in $Nodes) { Add-LibraryServer -ComputerName $Node

-Credential $Credential}

PS C:\> Add-LibraryServer -ComputerName "HAFileServer01.Contoso.com" -Crede

ntial $Credential

PS C:\> Add-LibraryShare -SharePath "\\HAFileServer01.Contoso.com\LibShare"

-Credential $Credential

The first command connects to VMMServer1 in the Contoso.com domain; the fol

lowing commands use this server by default.

The second command uses Get-Credential to prompt you to supply a user name

and password and stores your credentials in $Credential. The required crede

ntials for this operation are a domain account with administrator rights on

each node of a failover cluster hosting the highly available file server t

hat you want to add to VMM.

This example assumes that:

\* The Windows Server 2008 Failover Cluster Management

console was used earlier to create a 2-node failover cluster

with nodes LibraryNode01 and LibraryNode02.

\* The High Availability Wizard or "Configure a Service or

Application" action in the Failover Cluster Management

console was used earlier to create a highly available file

server named HAFileServer01.

\* The Failover Cluster Management console was used earlier to

add the share "\\HAFileServer01.Contoso.com\LibraryShare".

The third command stores, in $Nodes, the strings "LibraryNode01.Contoso.com

" and "LibraryNode02.Contoso.com".

The fourth command uses a foreach loop to pass each failover cluster node n

ame stored in $Nodes to Add-LibraryServer, which adds the two nodes as libr

ary servers. NOTE: For more information about the standard Windows PowerShe

ll foreach loop statement, type: Get-Help about\_ForEach. The foreach loop s

tatement is not the same as the Foreach-Object cmdlet, which uses “foreach”

as an alias.

The fifth command uses Add-LibraryServer to add the highly available file s

erver "HAFileServer01.Contoso.com" as a library server. Thus, in this examp

le, the highly available file server (HAFileServer01) has two physical file

servers (LibraryNode01 and LibraryNode02) as nodes.

The sixth command uses Add-LibraryShare to add the share on the highly avai

lable file server, specifying the share path: "\\HAFileServer01.Contoso.com

\LibraryShare".

REMARKS

For more information, type: "get-help Add-LibraryServer -detailed".

For technical information, type: "get-help Add-LibraryServer -full".

### Get-LibraryServer

SYNOPSIS

Gets Virtual Machine Manager library server objects from the Virtual Machin

e Manager database.

SYNTAX

Get-LibraryServer [[-ComputerName] <String>] [-ID <Guid>] [-VMMServer [<Str

ing ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent Virtual Machine Manager library ser

vers from the Virtual Machine Manager library.

For more information about library servers, type:

Get-Help Add-LibraryServer -detailed

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all library servers.

PS C:\> Get-LibraryServer -VMMServer VMMServer1.Contoso.com

Gets all objects that represent library servers from the VMM library on VMM

Server1 and displays information about these library servers to the user.

NOTE: The name of a library server is the same as its computer name.

2: Get a specific library server.

PS C:\> Get-LibraryServer -VMMServer VMMServer1.Contoso.com -ComputerName "

FileServer01.Contoso.com"

Gets from the library on VMMServer1 the object that represents the library

server FileServer01 and displays information about this library server to t

he user.

3: Get all library servers that match specified criteria.

PS C:\> $LibServers = Get-LibraryServer -VMMServer "VMMServer1.Contoso.com"

| where { $\_.Name -match "FileServer" }

Gets from the library on VMMServer1 all objects that represent library serv

ers whose name includes the string “FileServer” (such as FileServer01, File

Server02, and so on) and stores these library server objects in $LibServers

.

REMARKS

For more information, type: "get-help Get-LibraryServer -detailed".

For technical information, type: "get-help Get-LibraryServer -full".

### Remove-LibraryServer

SYNOPSIS

Removes a library server object from Virtual Machine Manager.

SYNTAX

Remove-LibraryServer [-LibraryServer] [<String LibraryServer>] -Credential

<PSCredential> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronous

ly] [<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents a library server (and all library objects

on that library server) from the Virtual Machine Manager database. Library

objects that have a corresponding file (such as .vhd or .vmdk files) store

d on the server's file system are not removed from the file system by this

cmdlet.

This cmdlet operates as follows:

\* If this library server is also the VMM server, you cannot remove the

library server, so the remove library server operation will fail.

\* If this computer is both a library server and a host, this cmdlet

removes only the library server component from VMM, but the

computer continues to function as a host.

\* If this computer is only a library server (not also a host or a VMM

server), the library server is removed from VMM.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a library server object from VMM.

PS C:\> $LibServ = Get-LibraryServer -VMMServer VMMServer1.Contoso.com -Com

puterName “FileServer01.Contoso.com”

PS C:\> Remove-LibraryServer -LibraryServer $LibServ

The first command retrieves from the VMM library on VMMServer1 the object t

hat represents library server FileServer01 and stores the library server ob

ject in variable $LibServ.

The second command removes the library server object, and all library share

s on this server, from the VMM library. When the Remove-LibraryServer cmdle

t is used with the -LibraryServer parameter as shown in this example, you c

an pass only one library server object to the cmdlet.

2: Remove from VMM multiple library server objects that have the specified

string in the server name.

PS C:\> $LibServers = Get-LibraryServer -VMMServer VMMServer1.Contoso.com |

where { $\_.Name -match “FileServer” }

PS C:\> $LibServer | Remove-LibraryServer

The first command gets from VMMServer1 all objects that represent library s

ervers with computer names that include the string “FileServer” and stores

the returned objects in $LibServers (an object array).

The second command passes each library server object in $LibServer to Remov

e-LibraryServer, which removes each object from VMM.

3: Remove a highly available library server and all of its nodes.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Credential = Get-Credential

PS C:\> $Cluster = Discover-Cluster -ComputerName "HAFileServer01.Contoso.c

om" -Credential $Credential

PS C:\> Remove-LibraryServer -LibraryServer "HAFileServer01.Contoso.com" -C

redential $Credential -RunAsynchronously

PS C:\> ForEach($ClusterNode in $Cluster.ClusterNodes) {Remove-LibraryServe

r -LibraryServer $ClusterNode -Credential $Credential -RunAsynchronously}

The first command connects to VMMServer1.

The second command uses Get-Credential to prompt you to supply a user name

and password and stores your credentials in $Credential. The required crede

ntials for this operation are either a local Administrator account or a dom

ain account with administrator rights on the library server. The following

commands use $Credential to pass your credentials to each cmdlet that requi

res credentials.

The third command uses Discover-Cluster to confirm that HAFileServer01 is a

highly available file server and stores the objects that represent the clu

ster nodes in $Cluster (an object array).

The fourth command removes the highly available file server (by specifying

its name) as a library server from VMM. The command uses the -RunAsynchrono

usly parameter to return control to the shell immediately (before this comm

and completes) because the last command does not need to wait until after t

his command finishes.

The last command uses a foreach loop to pass each object stored in $Cluster

(each of which is a node of the cluster) to the Remove-LibraryServer cmdle

t, which removes each node from VMM. The command uses the -RunAsynchronousl

y parameter to return control to the shell immediately.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

REMARKS

For more information, type: "get-help Remove-LibraryServer -detailed".

For technical information, type: "get-help Remove-LibraryServer -full".

### Set-LibraryServer

SYNOPSIS

Changes specific properties of a Virtual Machine Manager library server obj

ect.

SYNTAX

Set-LibraryServer [-LibraryServer] [<String LibraryServer>] [-Description <

String>] [-JobGroup <Guid>] [-JobVariable <String>] [-LibraryGroup <String>

] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Changes specific properties of a Virtual Machine Manager library server obj

ect. You can change the Description property and the LibraryGroup property.

You can also use this cmdlet as part of a job group, when used with the Ad

d-LibraryShare cmdlet, to add a set of library shares.

PARAMETERS

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-Description <String>

Specifies a description for the specified object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryGroup <String>

Specifies the group to which a VMM library server belongs. You can use

a library group to associate library servers with particular host group

s or to associate library servers with one another.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the description of a library server.

PS C:\> $LibServer = Get-LibraryServer -VMMServer "VMMServer1.Contoso.com"

-ComputerName "FileServer01.Contoso.com"

PS C:\> Set-LibraryServer -LibraryServer $LibServer -Description "Library s

erver for lab and pre-production"

The first command retrieves from the VMM library on VMMServer1 the object t

hat represents the library server FileServer01 and stores the library serve

r object in variable $LibServer.

The second command changes the description for FileServer01 to “Library ser

ver for lab and pre-production.”

2: Specify a library group for a library server.

PS C:\> $LibServer = Get-LibraryServer -VMMServer "VMMServer1.Contoso.com"

-ComputerName "FileServer02.Contoso.com"

PS C:\> Set-LibraryServer -LibraryServer $LibServer -LibraryGroup "LibraryG

roup01"

The first command gets the object that represents the library server named

FileServer02 and stores the library server object in $LibServer.

The second command changes the library group for FileServer02 to “LibraryGr

oup01.”

REMARKS

For more information, type: "get-help Set-LibraryServer -detailed".

For technical information, type: "get-help Set-LibraryServer -full".

### Add-LibraryShare

SYNOPSIS

Adds Windows shares on the file system of a library server as library share

s to the Virtual Machine Manager library.

SYNTAX

Add-LibraryShare [-SharePath] <String> [-Credential <PSCredential>] [-Descr

iption <String>] [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Gui

d>] [-RunAsynchronously] [-VMMServer [<String ServerConnection>]] [<CommonP

arameters>]

DETAILED DESCRIPTION

Adds Windows shares on the file system of a library server as library share

s to the Virtual Machine Manager library.

Before you can add a library share to the VMM library, you must first creat

e the share in the Windows file system. You can, for example, use Windows E

xplorer to create and share a folder that you want to add to the library.

\* Add-LibraryShare. If you create a Windows share at the same level as

the default library share (MSSCVMMLibrary) created by VMM Setup or on

a separate library server, use the Add-LibraryShare cmdlet to add that

share to the VMM library.

\* Refresh-LibraryShare. If you create a Windows folder under the default

VMM library share (MSSCVMMLibrary), you can use the Refresh-LibraryShare

cmdlet to import that share and its contents into the VMM library.

Alternatively, after you add a library share, VMM automatically scans

the share, discovers all existing objects stored on that share that

qualify as library objects, and adds the library objects to the library

in the VMM database.

NOTE: Library resources that can be discovered only by the library refreshe

r but not created by an administrator include virtual hard disks (Windows-b

ased .vhd files or VMware-based .vmdk files), virtual floppy disks (Windows

-based .vfd files or VMware-based .flp files), ISO images (.iso files), and

scripts.

PARAMETERS

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-Description <String>

Specifies a description for the specified object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a Windows share as a library share object to the VMM library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Add-LibraryShare –SharePath "\\FileServer01\AllVHDs"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command adds an object to the library that represents AllVHDs (a

Windows share located on FileServer01) as a library share object. This exa

mple assumes that FileServer01 is already a VMM library server.

2: Add two Windows shares as library share objects to the VMM library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $SharePaths = "\\FileServer01\AllVHDs", "\\FileServer01\AllISOs"

PS C:\> foreach( $SharePath in $SharePaths) { Add-LibraryShare -SharePath $

SharePath }

The first command connects to VMMServer1.

The second command stores the strings "\\FileServer01\AllVHDs" and "\\FileS

erver01\AllSOs" in $SharePaths. This example assumes that FileServer01 is a

lready a VMM library server.

The last command uses a foreach loop to pass the two share names stored in

$SharePaths to the Add-LibraryShare cmdlet, which adds each Windows share a

s a library share to VMM. As this command is processed, $Credential provide

s your credentials to Add-LibraryShare.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

REMARKS

For more information, type: "get-help Add-LibraryShare -detailed".

For technical information, type: "get-help Add-LibraryShare -full".

### Discover-LibraryShare

SYNOPSIS

Discovers all of the shares on the specified computer or library server man

aged by Virtual Machine Manager on which it is possible to add a library sh

are.

SYNTAX

Discover-LibraryShare [-ComputerName] <String> -Credential <PSCredential> [

<CommonParameters>]

Discover-LibraryShare -LibraryServer [<String LibraryServer>] [<CommonParam

eters>]

DETAILED DESCRIPTION

Discovers all of the shares on the specified computer or library server man

aged by Virtual Machine Manager on which it is possible to add a library sh

are.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Discover Windows shares on a computer that is not yet a VMM library serv

er.

PS C:\> $Credential = Get-Credential

PS C:\> Discover-LibraryShare -Credential $Credential -ComputerName "Server

1.Contoso.com"

The first command uses Get-Credential to prompt you to supply a user name a

nd password with permissions to access Windows shares on Server1 and stores

your credentials in $Credential.

The second command confirms that you have valid credentials for this operat

ion and then displays all existing Windows shares capable of becoming VMM l

ibrary shares on Server1.

2: Discover shares on a VMM library server.

PS C:\> Discover-LibraryShare -LibraryServer LibraryServer1.Contoso.com

Displays all Windows shares capable of becoming library shares that exist L

ibraryServer1 as well as all shares that are already VMM library shares.

REMARKS

For more information, type: "get-help Discover-LibraryShare -detailed".

For technical information, type: "get-help Discover-LibraryShare -full".

### Get-LibraryShare

SYNOPSIS

Gets Virtual Machine Manager library share objects from the Virtual Machine

Manager library.

SYNTAX

Get-LibraryShare [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [<Co

mmonParameters>]

DETAILED DESCRIPTION

Gets from the Virtual Machine Manager library one or more objects that repr

esent Virtual Machine Manager library shares located on library servers.

A library share is a Windows share on a Virtual Machine Manager library ser

ver that is used to store files that contain library resources. Resources c

an include virtual machine templates, hardware profiles, guest operating sy

stem profiles, virtual hard disks (Windows-based .vhd files or VMware-based

.vmdk files), virtual floppy disks (Windows-based .vfd files or VMware-bas

ed .flp files), ISO images (.iso files), and scripts, as well as stored vir

tual machines.

For more information, type:

Get-Help Add-LibraryShare -detailed

PARAMETERS

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all library shares.

PS C:\> Get-LibraryShare -VMMServer VMMServer1.Contoso.com

Gets from the VMM library on VMMServer1 all objects that represent library

shares on library servers and displays information about these library shar

es to the user.

2: Get a specific library share on the specified library server.

PS C:\> $LibShare = Get-LibraryShare -VMMServer VMMServer1.Contoso.com | wh

ere { $\_.LibraryServer.Name -eq "FileServer01.Contoso.com" -and $\_.Name -eq

"AllVhds" }

Gets from the library on VMMServer1 the object that represents the library

share named AllVhds (on library server FileServer01) and stores the share o

bject in $LibShare.

3: Get all library shares on a specific library server.

PS C:\> $LibServer = Get-LibraryServer -VMMServer VMMServer1.Contoso.com -C

omputerName "FileServer01.Contoso.com"

PS C:\> $AllLibShares = Get-LibraryShare | where { $\_.LibraryServer.Name -e

q "$LibServer" }

PS C:\> $AllLibShares | Get-Member

The first command retrieves from the library on VMMServer1 the object that

represents the library server FileServer01 and stores the library server ob

ject in $LibServer.

The second command gets all library share objects on FileServer01 and store

s the share objects in $AllLibShares.

The last command passes each share object in $AllLibShares to the Get-Membe

r cmdlet, which displays the .NET type for a library share object:

TypeName: Microsoft.SystemCenter.VirtualMachineManager.LibraryShare

The command also displays a list of methods and properties that are associa

ted with a VMM library share object.

REMARKS

For more information, type: "get-help Get-LibraryShare -detailed".

For technical information, type: "get-help Get-LibraryShare -full".

### Refresh-LibraryShare

SYNOPSIS

Refreshes the state and metadata of Virtual Machine Manager library objects

stored in a library share.

SYNTAX

Refresh-LibraryShare [-LibraryShare] <LibraryShare> [-JobVariable <String>]

[-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Refreshes (updates) the state and metadata of all Virtual Machine Manager l

ibrary objects stored in the specified library share. This refresh also fin

ds new library files (if any exist) on the specified library share as well

as new child shared folders under the specified library share, and adds an

object for each new library file or share to the Virtual Machine Manager li

brary.

PARAMETERS

-LibraryShare <LibraryShare>

Specifies a VMM library share object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Refresh the specified library share.

PS C:\> $LibShare = Get-LibraryShare -VMMServer VMMServer1.Contoso.com | wh

ere { $\_.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -eq

“AllVhds” }

PS C:\> Refresh-LibraryShare -LibraryShare $LibShare

The first command retrieves from the library on VMMServer1 the object that

represents the library share named AllVhds (on library server FileServer01)

and stores the share object for AllVhds in variable $LibShare.

The second command updates the state and metadata information for all libra

ry objects in the share stored in $LibShare, and then it adds any new libra

ry objects found in the share to the VMM library.

2: Refresh multiple library shares.

PS C:\> $LibShares = Get-LibraryShare -VMMServer VMMServer1.Contoso.com | w

here { $\_.LibraryServer.Name -eq "FileServer01.Contoso.com" -and $\_.Name -m

atch "vhd" }

PS C:\> $LibShares | Refresh-LibraryShare

The first command retrieves from the library on VMMServer1 all objects that

represent library shares on library server FileServer01 whose share name c

ontains the string "vhd" and stores these share objects in variable $LibSha

res.

The second command updates the information for all library shares stored in

$LibShares, and then it adds any new library objects found in these shares

to the VMM library.

REMARKS

For more information, type: "get-help Refresh-LibraryShare -detailed".

For technical information, type: "get-help Refresh-LibraryShare -full".

### Remove-LibraryShare

SYNOPSIS

Removes a library share object from Virtual Machine Manager but does not de

lete the share from the Windows file system.

SYNTAX

Remove-LibraryShare [-LibraryShare] <LibraryShare> [-JobVariable <String>]

[-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents a library share from the Virtual Machine

Manager library. This cmdlet does not remove any shares or files from the f

ile system on the computer.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-LibraryShare <LibraryShare>

Specifies a VMM library share object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a library share object from the VMM library,

PS C:\> $LibShare = Get-LibraryShare -VMMServer VMMServer1.Contoso.com | wh

ere { $\_.LibraryServer.name -eq “FileServer01.Contoso.com” -and $\_.Name -eq

“AllVhds” }

PS C:\> Remove-LibraryShare -LibraryShare $LibShare

The first command gets the object that represents the library share named A

llVhds (on library server FileServer01) from the VMM library on VMMServer1

and stores the share object in variable $LibShare.

The second command removes the library share object and all library objects

in this share from the VMM library but does not delete the share from the

file system on the library server.

2: Remove multiple library share objects from the VMM library.

PS C:\> $LibShares = Get-LibraryShare -VMMServer VMMServer1.Contoso.com | w

here { $\_.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -m

atch “vhd” }

PS C:\> $LibShares | Remove-LibraryShare

The first command gets from VMMServer1 all objects that represent library s

hares on library server FileServer01 whose share name includes the string “

vhd” and stores these share objects in $LibShares (an object array).

The second command passes each library share object in $LibShares to Remove

-LibraryShare. The cmdlet removes each of the library share objects from th

e VMM library but does not delete the corresponding shares from the file sy

stem on the library server.

REMARKS

For more information, type: "get-help Remove-LibraryShare -detailed".

For technical information, type: "get-help Remove-LibraryShare -full".

### Set-LibraryShare

SYNOPSIS

Changes the Description property of a Virtual Machine Manager library share

object.

SYNTAX

Set-LibraryShare [-LibraryShare] <LibraryShare> [-Description <String>] [-J

obVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParame

ters>]

DETAILED DESCRIPTION

Changes a property of a Virtual Machine Manager library share object. The o

nly property that you can change by using the Set-LibraryShare cmdlet is th

e Description property.

PARAMETERS

-LibraryShare <LibraryShare>

Specifies a VMM library share object.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the description of a library share.

PS C:\> $LibShare = Get-LibraryShare -VMMServer VMMServer1.Contoso.com | wh

ere { $\_.LibraryServer.Name -eq "FileServer01.Contoso.com" -and $\_.Name -eq

"FileShare01" }

PS C:\> Set-LibraryShare -LibraryShare $LibShare -Description "Library shar

e for lab and pre-production"

The first command retrieves from the VMM library on VMMServer1 the object t

hat represents the library share named FileShare01 on FileServer01 and stor

es the library share object in variable $LibShare.

The second command changes the description for FileShare01 to “Library shar

e for lab and pre-production.”

REMARKS

For more information, type: "get-help Set-LibraryShare -detailed".

For technical information, type: "get-help Set-LibraryShare -full".

## MachineConfig

### Get-MachineConfig

SYNOPSIS

Gets physical machine configuration objects from the Virtual Machine Manage

r database.

SYNTAX

Get-MachineConfig [-SourceComputerName <String>] [-VMMServer [<String Serve

rConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects from the Virtual Machine Manager database that rep

resent the physical machine configuration associated with one or more physi

cal computers. Information about a computer's hardware, physical disks, and

operating system is stored in the machine configuration object.

A physical machine configuration is used by the New-P2V cmdlet when it conv

erts a physical machine to a virtual machine. To perform this conversion, y

ou use a physical computer as a model from which to create an identical, or

nearly identical, virtual machine that has the same identity (ComputerName

.DomainName) as the physical machine.

PARAMETERS

-SourceComputerName <String>

Specifies the source computer for a physical-to-virtual machine convers

ion (P2V conversion) performed by VMM.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the source computer name.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get the machine configuration object for a particular physical machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-MachineConfig | where { $\_.Name -eq "Server1" }

The first command connects to VMMServer1 and gets the server object from th

e VMM database. The next command uses this server by default.

The second command gets from the VMM database the machine configuration (cr

eated earlier by using New-MachineConfig) for the physical machine named Se

rver1 and displays information about this object to the user.

For more information about creating a new machine configuration by gatherin

g machine configuration information from a physical source machine that you

plan to convert to a virtual machine managed by VMM, type: Get-Help New-Ma

chineConfig -detailed

2: Get all machine configuration objects in your VMM environment.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-MachineConfig

The first command connects to VMMServer1.

The second command gets all objects that represent machine configurations (

created earlier by using New-MachineConfig) and displays information about

these machine configuration objects to the user.

REMARKS

For more information, type: "get-help Get-MachineConfig -detailed".

For technical information, type: "get-help Get-MachineConfig -full".

### New-MachineConfig

SYNOPSIS

Creates a machine configuration object by gathering machine configuration i

nformation from a physical source machine that you plan to convert to a vir

tual machine managed by Virtual Machine Manager.

SYNTAX

New-MachineConfig -Credential <PSCredential> -SourceComputerName <String> [

-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer

[<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a machine configuration object by gathering machine configuration i

nformation from a physical machine that you plan to convert to a virtual ma

chine managed by Virtual Machine Manager. Information about a computer's ha

rdware, physical disks, and operating system is stored in the machine confi

guration object.

The New-MachineConfig cmdlet installs the Virtual Machine Manager P2V agent

software on the physical source machine, runs the configuration informatio

n gathering process, and creates and stores the resulting machine configura

tion object (which is associated with this physical source computer) in the

Virtual Machine Manager database.

A physical machine configuration is used when you use the New-P2V cmdlet to

convert a physical machine to a virtual machine. To perform this conversio

n, you use a physical computer as a model from which to create an identical

, or nearly identical, virtual machine that has the same identity (Computer

Name.DomainName) as the physical machine.

This cmdlet supports collecting machine configuration information from a ph

ysical source machine running the operating systems shown in the following

table:

ONLINE P2V OFFLINE P2V

---------- -----------

Windows Server 2008 Yes Yes

Windows Server 2003 with Service Pack 1 (or later) Yes Yes

Windows Server 2003 x64 Edition Yes Yes

Windows 2000 with Service Pack 4 No Yes

Windows XP with Service Pack 2 (or later) Yes Yes

Windows XP x64 Edition Yes Yes

Windows Vista Service Pack 1 Yes Yes

Windows Vista x64 Service Pack 1 Yes Yes

NOTE: Virtual Machine Manager support for P2V for Windows Server 2008 inclu

des both the typical installation of the operating system and the Server Co

re installation option of Windows Server 2008. P2V supports both 32-bit and

64-bit for any version of Windows Server 2008.

PARAMETERS

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-SourceComputerName <String>

Specifies the source computer for a physical-to-virtual machine convers

ion (P2V conversion) performed by VMM.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the source computer name.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Gather information from a physical source machine.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> New-MachineConfig -SourceComputerName "P2VSource01.Contoso.com" -Cr

edential $Credential

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are either a local Administrator account

or a domain account with administrator rights on the computer from which yo

u want to gather information.

The second command connects to VMMServer1 in the Contoso.com domain and ret

rieves the server object from the VMM database. The following command uses

this server by default.

The last command gathers the machine configuration information from the phy

sical source machine called P2VSource01 in the Contoso.com domain. As this

command is processed, $Credential provides your credentials to New-MachineC

onfig. The New-MachineConfig cmdlet stores the resulting machine configurat

ion object associated with P2VSource01.Contoso.com in the VMM database.

2: Determine the required patches for a particular conversion.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MachineConfig = New-MachineConfig -SourceComputerName "P2VSource02

.Contoso.com" -Credential $Credential

PS C:\> $MachineConfig.ErrorList

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password and stores your credentials in $Credential.

The second command connects to VMMServer1.

The third command gathers the machine configuration information from the ph

ysical source machine called P2VSource02 in the Contoso.com, and it stores

the machine configuration information in $MachineConfig. As this command is

processed, $Credential provides your credentials to New-MachineConfig.

The last command displays the list of errors, if any, that were detected on

the source machine. Any items with the value 'Error' must be resolved befo

re you attempt a physical-to-virtual conversion.

REMARKS

For more information, type: "get-help New-MachineConfig -detailed".

For technical information, type: "get-help New-MachineConfig -full".

### Remove-MachineConfig

SYNOPSIS

Removes a machine configuration object from Virtual Machine Manager.

SYNTAX

Remove-MachineConfig [-MachineConfig] <MachineConfig> [-Confirm] [-Credenti

al <PSCredential>] [-Force] [-JobVariable <String>] [-PROTipID <Guid>] [-Ru

nAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent a machine configuration from the

Virtual Machine Manager database and removes the Virtual Machine Manager P

2V agent from the physical source machine (if the agent is still installed)

.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-MachineConfig <MachineConfig>

Specifies a physical machine configuration to use when you convert a ph

ysical machine to a virtual machine. Machine configuration includes inf

ormation about the physical computer's hardware, disks, and operating s

ystem.

-Confirm

Prompts for confirmation before running the command.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific machine configuration from the VMM database.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MachineConfig = Get-MachineConfig | where { $\_.Name -eq "P2VSource

01.Contoso.com" }

PS C:\> Remove-MachineConfig -MachineConfig $MachineConfig -Force

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets from the VMM database the object that represents th

e machine configuration for the source computer named P2VSource01 and store

s the machine configuration object in variable $MachineConfig.

The third command removes the machine configuration object for Server1 from

the VMM database by using the -Force parameter.

2: Remove a P2V agent from a specific source computer.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MachineConfig = Get-MachineConfig | where { $\_.Name -eq "P2VSource

01.Contoso.com" }

PS C:\> Remove-MachineConfig -MachineConfig $MachineConfig -Credential $Cre

dential

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in $Credential. The required creden

tials for this operation are either a local Administrator account or a doma

in account with administrator rights on the computer from which you want to

gather information.

The second command connects to VMMServer1.

The third command gets the object that represents the machine configuration

for the source computer named P2VSource01 and stores the machine configura

tion object in $MachineConfig.

The fourth command removes the P2V agent from P2VSource01.

REMARKS

For more information, type: "get-help Remove-MachineConfig -detailed".

For technical information, type: "get-help Remove-MachineConfig -full".

## NetworkLocation

### Get-NetworkLocation

SYNOPSIS

Gets the list of network locations that the specified Virtual Machine Manag

er server can access.

SYNTAX

Get-NetworkLocation [-VMMServer [<String ServerConnection>]] [<CommonParame

ters>]

DETAILED DESCRIPTION

Gets the list of network locations that the specified Virtual Machine Manag

er server can access. The list includes both external and internal networks

and the network tag (if any) for each network.

PARAMETERS

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all network locations for the specified VMM server.

PS C:\> $VMMServer = Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> $NetLoc = Get-NetworkLocation -VMMServer $VMMServer

PS C:\> $NetLoc[1]

The first command connects to VMMServer1, retrieves the server object, and

stores it in variable $VMMServer.

The second command uses Get-NetworkLocation to retrieve the list of network

locations that VMMServer1 can access and stores the network location in va

riable $NetLoc (an object array). This example assumes that VMMServer1 can

access at least two networks (counting 0 to 1).

The last command displays the name of the second network location stored in

$NetLoc and the network tag (if any) for that network.

REMARKS

For more information, type: "get-help Get-NetworkLocation -detailed".

For technical information, type: "get-help Get-NetworkLocation -full".

## OperatingSystem

### Get-OperatingSystem

SYNOPSIS

Gets valid operating system objects from the Virtual Machine Manager databa

se.

SYNTAX

Get-OperatingSystem [-VMMServer [<String ServerConnection>]] [<CommonParame

ters>]

DETAILED DESCRIPTION

Gets one or more objects that represent operating systems from the Virtual

Machine Manager database. An operating system object is used to identify th

e operating system that is installed on a particular virtual hard disk.

PARAMETERS

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all operating system objects in your VMM environment.

PS C:\> Get-OperatingSystem -VMMServer VMMServer1.Contoso.com

Gets from the VMM database on VMMServer1 all objects that represent operati

ng systems and displays information about these operating system objects to

the user.

2: Get all operating system objects in your VMM environment with the specif

ied processor architecture.

PS C:\> Get-OperatingSystem -VMMServer VMMServer1.Contoso.com | where {$\_.A

rchitecture -eq "x86"} | Format-Table -property Name,Architecture

Gets all operating system objects from VMMServer1 and then selects only tho

se operating systems that have an x86 processor architecture. The command u

ses the Format-Table cmdlet to display only the Name and Architecture (such

as amd64 or x86) properties for each selected operating system.

REMARKS

For more information, type: "get-help Get-OperatingSystem -detailed".

For technical information, type: "get-help Get-OperatingSystem -full".

## P2V

### New-P2V

SYNOPSIS

Converts a physical machine to a virtual machine on a Windows-based host (H

yper-V or Virtual Server) managed by Virtual Machine Manager.

SYNTAX

New-P2V -MachineConfig <MachineConfig> -VMHost [<String Host>] [-Bus <Int32

>] [-Check] [-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-Crede

ntial <PSCredential>] [-DelayStart <Int32>] [-Description <String>] [-DiskS

izeAdd <Int32>] [-DriverPath <String>] [-Dynamic] [-Fixed] [-IDE <Boolean>]

[-JobGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>] [-MemoryMB <Int3

2>] [-Name <String>] [-NetworkLocation <String>] [-NetworkTag <String>] [-N

oConnection] [-Offline] [-OfflineDefaultGateway <String>] [-OfflineIPAddres

s <String>] [-OfflineNICMACAddress <String>] [-OfflinePrefixLength <String>

] [-OfflineSubnetMask <String>] [-OverridePatchPath <String>] [-Owner <Stri

ng>] [-Path <String>] [-PhysicalAddress <String>] [-PhysicalAddressType <St

ring>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSystem] [-RunAsU

serCredential <PSCredential>] [-RunAsynchronously] [-SCSI <Boolean>] [-Shut

down] [-SkipInstallVirtualizationGuestServices] [-SourceNetworkConnectionID

<String>] [-StartAction <String>] [-StartVM] [-StopAction <String>] [-Trig

ger] [-UseHardwareAssistedVirtualization] [-VirtualNetwork <VirtualNetwork>

] [-VLANEnabled] [-VLANID <Int32>] [-VMMServer [<String ServerConnection>]]

[-VolumeDeviceID <Guid>] [<CommonParameters>]

New-P2V -SourceComputerName <String> -VMHost [<String Host>] [-Bus <Int32>]

[-Check] [-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-Credent

ial <PSCredential>] [-DelayStart <Int32>] [-Description <String>] [-DiskSiz

eAdd <Int32>] [-DriverPath <String>] [-Dynamic] [-Fixed] [-IDE <Boolean>] [

-JobGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>] [-MemoryMB <Int32>

] [-Name <String>] [-NetworkLocation <String>] [-NetworkTag <String>] [-NoC

onnection] [-Offline] [-OfflineDefaultGateway <String>] [-OfflineIPAddress

<String>] [-OfflineNICMACAddress <String>] [-OfflinePrefixLength <String>]

[-OfflineSubnetMask <String>] [-OverridePatchPath <String>] [-Owner <String

>] [-Path <String>] [-PhysicalAddress <String>] [-PhysicalAddressType <Stri

ng>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSystem] [-RunAsUse

rCredential <PSCredential>] [-RunAsynchronously] [-SCSI <Boolean>] [-Shutdo

wn] [-SkipInstallVirtualizationGuestServices] [-SourceNetworkConnectionID <

String>] [-StartAction <String>] [-StartVM] [-StopAction <String>] [-Trigge

r] [-UseHardwareAssistedVirtualization] [-VirtualNetwork <VirtualNetwork>]

[-VLANEnabled] [-VLANID <Int32>] [-VMMServer [<String ServerConnection>]] [

-VolumeDeviceID <Guid>] [<CommonParameters>]

DETAILED DESCRIPTION

Converts a physical machine to a virtual machine on a Windows-based host (H

yper-V or Virtual Server) managed by Virtual Machine Manager. You cannot sp

ecify a VMware ESX host as the destination host for the new virtual machine

.

In a P2V conversion, you create a virtual machine from a physical source co

mputer. The New-P2V cmdlet configures the new virtual machine to have the s

ame hardware, software, and configuration settings as the physical source c

omputer and to have the same identity (ComputerName.DomainName) as the sour

ce computer. During the P2V conversion, disk images of the hard disks on th

e physical computer are copied to Windows-based virtual hard disk files (.v

hd files) for use in the new virtual machine.

VMM 2008 SUPPORTS P2V CONVERSION FOR THESE OPERATING SYSTEMS

---------------------------------------------------

This cmdlet supports a P2V conversion for a physical source machine running

one of the operating systems listed in the following table:

ONLINE P2V OFFLINE P2V

---------- -----------

Windows Server 2008 Yes Yes

Windows Server 2003 with Service Pack 1 (or later) Yes Yes

Windows Server 2003 x64 Edition Yes Yes

Windows 2000 with Service Pack 4 No Yes

Windows XP with Service Pack 2 (or later) Yes Yes

Windows XP x64 Edition Yes Yes

Windows Vista Service Pack 1 Yes Yes

Windows Vista x64 Service Pack 1 Yes Yes

NOTES:

\* STANDARD OR CORE - Virtual Machine Manager support for P2V for

Windows Server 2008 includes both the typical installation of the

operating system and the Server Core installation option. P2V

supports both 32-bit and 64-bit for any version of Windows Server 2008.

\* OFFLINE USES WINDOWS PE - When you specify the -Offline parameter with

New-P2V, the conversion process will automatically start the source

computer in the Windows Preinstallation Environment (Windows PE) and

then perform the conversion. You can use the -DriverPath parameter with

the New-P2V cmdlet to specify the driver used by Windows PE for network

access and storage access during the offline conversion.

\* WINDOWS AIK - In VMM 2008, when you install a Virtual Machine Manager

server, Setup also automatically installs the Windows Automated

Installation Kit (Windows AIK), which is required for a P2V conversion.

ADDING NEEDED FILES TO THE PATCH CACHE

--------------------------------------

Some conversions might require that additional files be added to the intern

al cache. You can use the Add-Patch cmdlet to add the required files to the

cache.

PARAMETERS

-MachineConfig <MachineConfig>

Specifies a physical machine configuration to use when you convert a ph

ysical machine to a virtual machine. Machine configuration includes inf

ormation about the physical computer's hardware, disks, and operating s

ystem.

-SourceComputerName <String>

Specifies the source computer for a physical-to-virtual machine convers

ion (P2V conversion) performed by VMM.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the source computer name.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-Bus <Int32>

Specifies the IDE bus to which to attach a virtual disk drive or virtua

l DVD drive, or the SCSI bus to which to attach a virtual disk drive.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter. For more information about how VMM 2008 impleme

nts IDE and SCSI buses, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enha

ncements.

-Check

Checks the source computer for any patches required for conversion.

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-DelayStart <Int32>

Specifies the number of seconds to wait after the virtualization servic

e starts before automatically starting a virtual machine. Used to stagg

er the startup time of multiple virtual machines to help reduce the dem

and on the physical computer’s resources. A typical setting might be 30

to 60 seconds.

TYPE OF HOST MAXIMUM CONFIGURABLE DELAY

------------ --------------------------------

Hyper-V 1000000000 seconds (277777 hours)

Virtual Server 86400 seconds (24 hours)

VMware ESX 65535 seconds (18 hours)

-Description <String>

Specifies a description for the specified object.

-DiskSizeAdd <Int32>

Specifies, in megabytes (MB), the amount of additional disk space to ad

d to a virtual hard disk when performing a physical-to-virtual (P2V) or

virtual-to-virtual (V2V) machine conversion. Volumes located on the vi

rtual hard disk are automatically extended to fill the entire virtual h

ard disk.

-DriverPath <String>

Specifies the path to drivers for any offline physical-to-virtual machi

ne conversion (P2V conversion).

-Dynamic

Specifies that a virtual hard disk can expand dynamically.

-Fixed

Specifies that a virtual hard disk is fixed in size.

-IDE <Boolean>

Specifies IDE as the bus type to which to attach a virtual disk drive o

bject or a virtual DVD drive object configured on a virtual machine or

on a template. (For more information about how VMM 2008 implements the

IDE bus, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements.)

Example format: -IDE –Bus 0 –LUN 1

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LUN <Int32>

Specifies the logical unit number (LUN) for a virtual disk drive object

or for a virtual DVD drive object on an IDE bus, or for a virtual disk

drive object on a SCSI bus.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-Name <String>

Specifies the name of a VMM object.

-NetworkLocation <String>

Specifies the network location for a physical network adapter or for a

virtual network adapter, or changes the default network location of a h

ost's physical network adapter.

Example formats:

-NetworkLocation $NetLoc ($NetLoc might contain "Corp.Contoso.com")

-OverrideNetworkLocation $TRUE –NetworkLocation "HostNICNewLocation.Co

ntoso.com"

-NetworkTag <String>

Specifies a word or phrase to associate with a virtual network adapter

that is configured to connect to a specific internal or external networ

k on the host. The NetworkTag identifies all VMs with the same NetworkT

ag as members of the same network. VMM uses a NeworkTag (if one exists)

when it evaluates hosts as possible candidates on which to deploy a VM

. If the host does not include VMs on the network with the same Network

Tag as the VM to be placed, the host receives zero stars in the placeme

nt process.

-NoConnection

Disconnects a virtual network adapter from a virtual network.

-Offline

Specifies that the operation is performed offline.

-OfflineDefaultGateway <String>

Specifies the gateway router that Windows PE uses during an offline P2V

conversion.

-OfflineIPAddress <String>

Specifies an IPv4 or IPv6 address on the source computer that Windows P

E uses during an offline P2V conversion. Use with the OfflineSubnetMask

parameter (for an IPv4 address) or with the OfflinePrefixLength parame

ter (for an IPv6 address).

-OfflineNICMACAddress <String>

Specifies the MAC address of the network interface card (NIC) on the so

urce computer that Windows PE uses during an offline P2V conversion.

-OfflinePrefixLength <String>

Specifies the length of the prefix for the IPv6 address on the source c

omputer that Windows PE uses during an offline P2V conversion.

-OfflineSubnetMask <String>

Specifies the subnet mask for the IPv4 address on the source computer t

hat Windows PE uses during an offline P2V conversion.

-OverridePatchPath <String>

For internal use only (not for use in your code).

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PhysicalAddress <String>

Specifies the physical address (MAC address) of a physical or virtual n

etwork adapter.

Note: In VMM 2007, this parameter was named EthernetAddress.

-PhysicalAddressType <String>

Specifies the type of physical address (MAC address) to use for a virtu

al network adapter:

Valid values: Static, Dynamic

Note: In VMM 2007, this parameter was named EthernetAddressType.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsSystem

Specifies that a virtual machine on a Virtual Server host will run unde

r the local system account. If specified, Virtual Server will not autom

atically start the virtual machine when the Virtual Server service star

ts. (This parameter does not apply to virtual machines on Hyper-V or VM

ware ESX hosts because Hyper-V and VMware run a virtual machine under t

he local system account by default; you cannot change this setting on t

hose virtualization platforms.)

-RunAsUserCredential <PSCredential>

Specifies the guest account (domain\account) that a virtual machine on

a Virtual Server host runs under. If specified, Virtual Server will aut

omatically start a virtual machine when the Virtual Server service star

ts. For enhanced security, create a special account with limited permis

sions:

FILE TYPE MINIMUM REQUIRED PERMISSIONS FOR GUEST ACCOUNT

----------- ----------------------------------------------

.vmc file Read Data, Write Data, Execute File

.vmc folder List Folder, Write/Create File (required to save VM state)

.vhd file Read Data, Read Attributes, Read Extended Attributes,

Write Data

.vnc file Execute File, Read Data, Read Attributes, Read

(required if VM connects to a virtual network)

Note: This parameter does not apply to virtual machines on Hyper-V or V

Mware ESX hosts.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SCSI <Boolean>

Specifies SCSI as the bus type to which to attach a virtual disk drive

object configured on a virtual machine or on a template.

Example format: -SCSI -Bus 0 -LUN 0

For information about the number of devices per controller on a SCSI bu

s that VMM 2008 supports for each virtualization platform (Hyper-V, Vir

tual Server, or VMware), type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enh

ancements.

-Shutdown

Specifies that the source server shuts down after a successful physical

-to-virtual machine conversion (P2V conversion).

-SkipInstallVirtualizationGuestServices

Skips the installation of virtualization guest services on a Windows-ba

sed virtual machine. By default, this parameter is set to FALSE and

VMM installs the appropriate virtualization guest service automatically

. For a virtual machine on a Hyper-V host, the virtualization guest ser

vice is called Integration Components (VMGuest.iso). For a virtual mach

ine on a Virtual Server host, the virtualization guest service is calle

d Virtual Machine Additions (VMAdditions.iso). Virtual machines on a VM

ware ESX host do not use a virtualization guest service.

-SourceNetworkConnectionID <String>

Specifies the MAC address or network name of the physical network adapt

er to be converted into a virtual network adapter in the virtual machin

e.

-StartAction <String>

Specifies the behavior of a virtual machine when the virtualization ser

vice (Hyper-V, Virtual Server, or VMware) starts. To specify that a vir

tual machine deployed on a Virtual Server host starts automatically, us

e the -RunAsUserCredential parameter to specify an account with appropr

iate permissions (otherwise, the StartAction reverts to NeverAutoTurnOn

VM).

Valid values: AlwaysAutoTurnOnVM, NeverAutoTurnOnVM, TurnOnVMIfRunningW

henVSStopped

-StartVM

Specifies that the virtual machine starts when it arrives at the destin

ation host.

-StopAction <String>

Specifies the behavior of the virtual machine when the virtualization s

ervice (Hyper-V, Virtual Server, or VMware) stops.

Valid values: SaveVM, TurnOffVM, ShutdownGuestOS

-Trigger

Starts the execution a job group for a physical-to-virtual (P2V) conver

sion, a virtual-to-virtual (V2V) conversion, or the conversion of a phy

sical hard disk to a virtual hard disk.

-UseHardwareAssistedVirtualization

Specifies that, for a virtual machine deployed on a Virtual Server host

, hardware-assisted virtualization is used if it is available (when set

to TRUE). The Virtual Server host must support AMD Virtualization (AMD

-V) or Intel Virtualization Technology (Intel-VT) hardware virtualizati

on. This parameter does not apply to virtual machines on Hyper-V hosts

or VMware ESX hosts.

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-VLANEnabled

Enables a virtual LAN (a VLAN) for use by virtual machines on a Hyper-V

or VMware ESX host.

Example format for a single VLAN:

-VLANEnabled -VLANMode "Access" -VLANID 35

Example format for multiple VLANs:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

-VLANID <Int32>

Assigns a numerical identifier in the range 1-4094 to a virtual network

adapter on a virtual machine or to a physical network adapter on a Hyp

er-V or VMware ESX host.

Configure a VLANID on a Hyper-V or VMware ESX host:

\* On an externally bound physical network adapter.

Configure a VLANID on a virtual network adapter of a virtual machine:

\* Bound to a physical network adapter on the host, or

\* Bound to an internal virtual network on the host.

Example format:

-VLANEnabled

-VLANMode "Access" -VLANID 35

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VolumeDeviceID <Guid>

Specifies the device ID of the volume to convert in a physical-to-virtu

al machine conversion (P2V conversion).

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Convert a physical machine to a virtual machine.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> New-P2V -SourceComputerName "P2VSource01.Contoso.com" -VMHost $VMHo

st -Name "VM01" -Path "C:\MyVMs" -MemoryMB 256 -Credential $Credential -Run

Asynchronously

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are either a local Administrator account

or a domain account with administrator rights on the computer that you want

to convert.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands use this

server by default.

The third command gets from the VMM database the object that represents the

host named VMHost01 and stores the host object in variable $VMHost.

The last command performs the following operations:

\* Creates a virtual machine named VM01 from the source physical

machine named P2VSource01 in the Contoso.com domain.

\* Deploys the new virtual machine on the C: drive of VMHost01

in the MyVMs folder. In this example, all of the physical disks

on P2VSource01.Contoso.com will be imaged and attached to

the new virtual machine.

\* Assigns 256 MB of memory on the host to the new virtual machine.

\* Uses $Credential to provide your credentials to New-P2V.

\* Uses the -RunAsynchronously parameter to return control to the shell

immediately (before the command completes).

2: Convert a physical machine to a large-capacity VM with specific network

settings.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> New-P2V -SourceComputerName "P2VSource02.Contoso.com" -VMHost $VMHo

st -Name "VM01" -Path "C:\MyVMs" -CPUCount 4 -MemoryMB 4096 -Credential $Cr

edential -RunAsynchronously -StopAction SaveVM -SourceNetworkConnectionID "

00:15:5D:BC:42:02" -PhysicalAddress "00:15:5D:BC:42:02" -PhysicalAddressTyp

e Static -VirtualNetwork "External Network (Network Adapter #2)" -NetworkLo

cation "Contoso.com"

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password and then stores your credentials in $Credential.

The second command connects to VMMServer1.

The third command gets the object that represents the host named VMHost02 a

nd stores the host object in $VMHost.

The last command performs the following operations:

\* Creates a virtual machine named VM01 from the source physical

machine named P2VSource02 (in the Contoso.com domain).

\* Deploys the new virtual machine on the C: drive of VMHost02

in the MyVMs folder. In this example, all of the physical disks

on P2VSource02.Contoso.com will be imaged and attached to

the new virtual machine (this is the default behavior).

\* Assigns 4 CPUs to the new virtual machine.

\* Assigns 4096 MB of memory on the host to the new virtual machine.

\* Uses $Credential to provide your credentials to New-P2V.

\* Specifies that the host should save the virtual machine when

the hypervisor is stopped (that is, when the host on which the

virtual machine is deployed is stopped).

\* Specifies the MAC address of the source host

(-SourceNetworkConnectionID "00:15:5D:BC:42:02").

\* Specifies the MAC address of the new virtual machine

(-PhysicalAddress "00:15:5D:BC:42:02").

\* Specifies that the MAC address in the new virtual machine is

static (-PhysicalAddressType Static).

\* Specifies the virtual network (and network adapter) on the host

that the new virtual machine connects to.

\* Specifies the network location of the new virtual machine as

Contoso.com.

\* Uses the -RunAsynchronously parameter to return control to the

shell immediately (before the command completes).

3: Convert a physical machine to a VM by performing an offline conversion.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> New-P2V -SourceComputerName "P2VSource01.Contoso.com" -Credential $

Credential -VMHost $VMHost -Path "D:\vms" -RunAsynchronously -Trigger -Name

"VM01" -Offline -CPUCount 2 -MemoryMB 1024

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are either a local Administrator account

or a domain account with administrator rights on the computer that you want

to convert.

The second command connects to VMMServer1.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in $VMHost.

The last command performs the following operations:

\* Creates a virtual machine named VM01 from the source physical

machine named P2VSource01 in the Contoso.com domain.

\* Deploys the new virtual machine on the D: drive of VMHost01

in the VMs folder. In this example, all of the physical disks

on P2VSource01.Contoso.com will be imaged and attached to

the new virtual machine.

\* Assigns 1024 MB of memory and 2 virtual CPUs to the new virtual

machine.

\* Uses $Credential to provide your credentials to New-P2V.

\* Performs the conversion offline (boots source into WINPE).

Offline is optional for all operating systems except Windows 2000.

\* Uses the -RunAsynchronously parameter to return control to the shell

immediately (before the command completes).

4: Convert a physical machine to a VM offline, using a static IP address du

ring the conversion.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Credential = get-credential

PS C:\> New-MachineConfig -SourceComputerName "P2VSource01.Contoso.com" -Cr

edential $Credential

PS C:\> $MachineConfig = Get-MachineConfig | where {$\_.Name -eq "P2VSource0

1.Contoso.com"}

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "VMHost01.Contoso.com"}

PS C:\> New-P2V -Credential $Credential -VMHost $VMHost -Path "D:\VMs" -Own

er "Contoso\User1" -Trigger -Name "VM01" -MachineConfig $MachineConfig -Off

line -Shutdown -OfflineIPAddress "192.168.100.100" -OfflineNICMacAddress "0

0:11:22:33:44:55" -OfflineDefaultGateway "192.168.100.1" -OfflineSubnetMask

"255.255.255.0" -CPUCount 1 -MemoryMB 1024 -RunAsSystem -StartAction Never

AutoTurnOnVM -UseHardwareAssistedVirtualization $FALSE -StopAction SaveVM -

StartVM -RunAsynchronously

The first command connects to VMMServer1.

The second command uses Get-Credential to prompt you to supply a user name

and password and stores your credentials in $Credential. The required crede

ntials for this operation are either a local Administrator account or a dom

ain account with administrator rights on the computer that you want to conv

ert.

The third command gathers the machine configuration information from the ph

ysical source machine called P2VSource01 (in the Contoso.com domain), using

$Credential to verify your credentials, and stores the resulting machine c

onfiguration object for P2VSource01 in the VMM database.

The fourth command gets the machine configuration object created by the pre

ceding command and stores the returned object in $MachineConfig.

The fifth command gets the object that represents the host named VMHost01 a

nd stores the returned object in $VMHost.

The last command performs the following operations:

\* Uses $Credential to provide your credentials to New-P2V.

\* Deploys the new virtual machine on the D: drive of VMHost01 in the

"VMs" folder. In this example, all of the physical disks on

P2VSource01.Contoso.com will be imaged and attached to the new

virtual machine.

\* Specifies an owner for the virtual machine created by New-P2V.

\* Creates a virtual machine named VM01 from the source physical machine

named P2VSource01.

\* Performs the conversion offline (that is, boots the source computer

into Windows PE). The IP address, subnet mask, and gateway used by

Windows PE are specified, as is the MAC address of the physical

network adapter to which the IP address will be bound. VMM turns off

the source computer after the conversion is complete.

NOTE: An offline conversion is an available option for all operating

systems except Windows 2000.

\* Assigns 1024 MB of memory and 2 virtual CPUs to the new virtual

machine.

\* Uses the -RunAsynchronously parameter to return control to the

command shell immediately (before the command completes).

REMARKS

For more information, type: "get-help New-P2V -detailed".

For technical information, type: "get-help New-P2V -full".

## Patch

### Add-Patch

SYNOPSIS

Adds information about patches and binaries to the Virtual Machine Manager

patch cache.

SYNTAX

Add-Patch [-JobVariable <String>] [-PatchFilePath <String>] [-PROTipID <Gui

d>] [-RunAsynchronously] [-VMMServer [<String ServerConnection>]] [<CommonP

arameters>]

DETAILED DESCRIPTION

Adds information about patches and binaries to the Virtual Machine Manager

patch cache. Patches are required for physical-to-virtual machine conversio

ns (P2V conversions) as well as for virtual-to-virtual machine conversions

(V2V conversions).

The Add-Patch cmdlet:

\* Updates the Virtual Machine Manager database with information

about patches.

\* Extracts required binaries to:

<C>:\Program Files\Microsoft System Center Virtual Machine Manager 2008\

VMMData

To determine which patches are required for a particular conversion, run th

e appropriate cmdlet to gather information about the source:

\* New-MachineConfig, for P2V conversions

\* New-VMXMachineConfig, for V2V conversions

PARAMETERS

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PatchFilePath <String>

Specifies the path (to a folder on the file system where VMM is install

ed or to a network share) where P2V or V2V patch files are located.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a new patch from the default patch import directory.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Add-Patch

Before running the commands illustrated in this example, place any patch fi

les (.cab or .exe files) into the Patch Cache folder located in the VMM ins

tallation directory on the VMM server. The default location is:

<C>:\Program Files\Microsoft System Center Virtual Machine Manager 2008\Pat

ch Import

The first command connects to VMMServer1 in the Contoso.com domain; the fol

lowing command uses this server by default.

The second command extracts any patches found in the Patch Import folder an

d adds these patches to the VMM patch cache.

NOTE: The patch files will automatically be deleted from the Patch Import f

older after they are successfully added to the patch cache.

REMARKS

For more information, type: "get-help Add-Patch -detailed".

For technical information, type: "get-help Add-Patch -full".

## PhysicalAddress

### New-PhysicalAddress

SYNOPSIS

Returns the next available physical address (MAC address) if a range of MAC

addresses has been configured for your Virtual Machine Manager environment

.

SYNTAX

New-PhysicalAddress [-Commit] [-VMMServer [<String ServerConnection>]] [<Co

mmonParameters>]

DETAILED DESCRIPTION

Returns the next available physical address (MAC address) if a range of MAC

addresses has been configured for your Virtual Machine Manager environment

. You can configure a range of MAC addresses by using the Set-VMMServer cmd

let (or by using the Administrator Console).

If a range of MAC addresses has been configured on the Virtual Machine Mana

ger server, these addresses are available for use on:

\* Any virtual network adapter configured on any virtual machine deployed

on a managed host.

\* Any hardware profile or template stored on any managed library server.

If you use this cmdlet with the –Commit parameter, you can pass the retriev

ed MAC address object to the New-VirtualNetworkAdapter cmdlet to assign a u

nique MAC address to a virtual machine, a template, or a hardware profile.

PARAMETERS

-Commit

Allocates the next available physical address (MAC address) for use on

a virtual network adapter configured on a virtual machine, a template,

or a hardware profile, and increments the next available address to pre

vent reuse of the address just allocated.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Obtain a new MAC address from the range specified on the VMM server.

PS C:\> Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> $MACAddress = New-PhysicalAddress -Commit

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command obtains the physical address (MAC address) from the rang

e of available addresses configured on the VMM server and stores the MAC ad

dress object in $MACAddress for later use.

The command uses the -Commit parameter to commit the allocation so that the

address is available for use with the New-VirtualNetworkAdapter cmdlet (wh

ich you can use to assign the MAC address to a virtual network adapter on a

VM, a template, or a hardware profile). If you use the –Commit parameter w

ith New-PhysicalAddress, this specific address will not be re-assigned to a

future request after you have used New-VirtualNetworkAdapter to assign it

to a virtual network adapter.

IMPORTANT:

If the VMM server currently does not have a range of MAC addresses configur

ed, you can use the Set-VMMServer cmdlet with the -PhysicalAddressRangeStar

t and -PhysicalAddressRangeEnd parameters to make a range of addresses avai

lable. For more information, type:

Get-Help Set-VMMServer -example

2: Set a MAC address for a virtual network adapter in an existing virtual m

achine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MACAddress = New-PhysicalAddress -Commit

PS C:\> $VMHost Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02" -VMHost $VMHost

PS C:\> $VirtualNetworkAdapter = Get-VirtualNetworkAdapter -VM $VM

PS C:\> Set-VirtualNetworkAdapter -VirtualNetworkAdapter $VirtualNetworkAda

pter -PhysicalAddress $MACAddress -PhysicalAddressType Static

The first command connects to VMMServer1.

The second command obtains the physical address (MAC address) from the rang

e of available addresses configured on the VMM server and stores the MAC ad

dress object in $MACAddress for later use.

NOTE: Using the Commit parameter commits the allocation of this

MAC address so that the address is available for use with the

Set-VirtualNetworkAdapter cmdlet in the last command. When you use

the Commit parameter with New-PhysicalAddress, this specific address

will not be re-assigned to a future request after you have used

Set-VirtualNetworkAdapter to assign it to a virtual network adapter.

The third, forth and fifth commands are used to get the virtual network ada

pter of the specified virtual machine (VM02) from the specified host (VMHos

t02) and to store the virtual network adapter object in $VirtualNetworkAdap

ter.

The last command changes the MAC address type of the virtual network adapte

r to a static MAC address type and uses the MAC address that was obtained e

arlier from the address pool by using the New-PhysicalAddress cmdlet.

IMPORTANT:

If your VMM server currently does not have a range of MAC addresses configu

red, you can use the Set-VMMServer cmdlet with the -PhysicalAddressRangeSta

rt and -PhysicalAddressRangeEnd parameters to make a range of addresses ava

ilable.

REMARKS

For more information, type: "get-help New-PhysicalAddress -detailed".

For technical information, type: "get-help New-PhysicalAddress -full".

## PROTip

### Dismiss-PROTip

SYNOPSIS

Dismisses a PRO tip object that is no longer needed in Virtual Machine Mana

ger.

SYNTAX

Dismiss-PROTip [-PROTip] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAs

ynchronously] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Dismisses a PRO tip object that is no longer needed in Virtual Machine Mana

ger. You can use this cmdlet to dismiss a PRO tip if, for example, its reco

mmended action is no longer valid or is out-of-date.

Virtual Machine Manager dismisses some PRO tips automatically if the condit

ion that generated the alert, and the resulting PRO tip recommended action,

is no longer an issue.

PARAMETERS

-PROTip

Specifies a PRO tip object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Dismiss the first active PRO tip.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $AllPROTips = Get-PROTip

PS C:\> Dismiss-PROTip -PROTip $AllPROTips[0]

The first command connects to VMMServer1 and gets the server object from th

e VMM database. The following commands use this server by default.

The second command gets all active PRO tip objects from the VMM database an

d stores the PRO tip objects in variable $AllPROTips (an object array).

The last command dismisses the first tip ($AllPROTips[0]).

REMARKS

For more information, type: "get-help Dismiss-PROTip -detailed".

For technical information, type: "get-help Dismiss-PROTip -full".

### Get-PROTip

SYNOPSIS

Gets Performance and Resource Optimization tip (PRO tip) objects from the V

irtual Machine Manager database.

SYNTAX

Get-PROTip -VMHost [<String Host>] [-PROTipID <Guid>] [-VMMServer [<String

ServerConnection>]] [<CommonParameters>]

Get-PROTip -VM [<String VM>] [-PROTipID <Guid>] [-VMMServer [<String Server

Connection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent Performance and Resource Optimizati

on tip (PRO tip) objects from the Virtual Machine Manager database.

If PRO is enabled, a PRO tip recommends an action in response to an alert g

enerated by System Center Operations Manager 2007 for hosts that are member

s of a host group or for hosts configured in a host cluster, as well as for

the virtual machines deployed on those hosts. A recommended action might b

e to move a VM to a new host or to add a CPU to the VM.

PRO provides workload- and application-aware resource optimization within h

ost groups or host clusters that are managed by both Virtual Machine Manage

r and Operations Manager. To receive PRO tips for these hosts, you must fir

st configure PRO for Virtual Machine Manager. This includes deploying Opera

tions Manager, which generates the PRO tips based on monitors provided by P

RO-enabled management packs. PRO tip recommendations are based on policies

implemented through Operations Manager.

You can use the Get-PROTip cmdlet (and the other PROTip cmdlets) only on Hy

per-V or VMware hosts that belong to a host group or that are configured in

a host cluster.

If you use the Set-VMHostCluster cmdlet to specify that VMM will implement

PRO tips automatically:

\* For Hyper-V hosts configured in a host cluster, VMM can use the

Windows 2008 Cluster Migration feature (sometimes called Quick

Migration) to move VMs transparently between nodes in the cluster.

\* For VMware hosts configured in a host cluster, VMM can use the VMware

Live Migration feature (VMotion) to move VMs transparently between

nodes in the cluster.

For more information about Operations Manager and PRO tips, type:

Get-Help about\_VMM\_2008\_Ops\_Mgr\_Integration

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all active PRO tips for all hosts and all VMs managed by VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-PROTip

The first command connects to VMMServer1 and gets the server object from th

e VMM database. The next command uses this server by default.

The second command gets all active PRO tips from the VMM database and displ

ays information about them to the user.

2: Get all active PRO tips for a specific host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> Get-PROTip -VMHost $VMHost

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost02

and stores the host object in $VMHost.

The last command gets all active PRO tips that belong to VMHost02 and displ

ays information about each tip to the user.

3: Get all active PRO tips for a VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> Get-PROTip -VM $VM

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM03 and stores the virtual machine object in variable $VM.

The last command gets all active PRO tips targeted at VM03 and displays inf

ormation about them to the user.

REMARKS

For more information, type: "get-help Get-PROTip -detailed".

For technical information, type: "get-help Get-PROTip -full".

### Invoke-PROTip

SYNOPSIS

Performs the action recommended by a PRO tip. You can use this cmdlet to ma

nually invoke the action recommended by a PRO tip that is not set to be imp

lemented automatically.

SYNTAX

Invoke-PROTip [-PROTip] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsy

nchronously] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Performs the action recommended by a PRO tip. You can use this cmdlet to ma

nually invoke the action recommended by a PRO tip that is not set to be imp

lemented automatically.

For more information about PRO tips, type:

Get-Help Get-PROTip -detailed

PARAMETERS

-PROTip

Specifies a PRO tip object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Invoke the first active PRO tip.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $AllPROTips = Get-PROTip

PS C:\> Invoke-PROTip -PROTip $AllPROTips[0]

The first command connects to VMMServer1 and gets the server object from th

e VMM database. The following commands use this server by default.

The second command gets all active PRO tips from the VMM database and store

s the PRO tip objects in variable $AllPROTips (an object array).

The last command implements the suggested action from the first tip ($AllPR

OTips[0]) returned from the preceding command.

REMARKS

For more information, type: "get-help Invoke-PROTip -detailed".

For technical information, type: "get-help Invoke-PROTip -full".

### Set-PROTip

SYNOPSIS

Sets the status of a PRO tip object.

SYNTAX

Set-PROTip -PROTipID <Guid> [-ActionDetails <String>] [-ActionDetailsOpsMgr

String <String[]>] [-ActionScript <String>] [-ActionSummary <String>] [-Act

ionSummaryOpsMgrString <String>] [-JobVariable <String>] [-LastError <Strin

g>] [-LastErrorOpsMgrString <String>] [-RunAsynchronously] [-TipStatus] [-V

MMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Sets the status of a PRO tip object. This cmdlet, which is called by PRO ti

p implementation actions and is for use in building PRO Packs, is used by V

irtual Machine Manager to update the status of a PRO tip while performing t

he action recommended by the PRO tip.

As shown in the example, you can also use this cmdlet to manually update th

e status of an individual PRO tip or an array of PRO tips.

For more information about PRO tips, type:

Get-Help Get-PROTip -detailed

PARAMETERS

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-ActionDetails <String>

Provides a detailed description of what implementing this PRO tip will

do.

-ActionDetailsOpsMgrString <String[]>

Specifies an array of strings used to provide localized ActionDetails t

ext. The first element of the array should be the GUID of the OpsMgr st

ring and the following elements should be the parameters for string for

matting.

-ActionScript <String>

Specifies the script that will run by implementing this PRO tip.

-ActionSummary <String>

Provides a summary description of what implementing this PRO tip will d

o.

-ActionSummaryOpsMgrString <String>

Specifies an array of strings used to provide localized ActionSummary t

ext. The first element of the array should be the GUID of the OpsMgr st

ring and the following elements should be the parameters for string for

matting.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LastError <String>

Specifies the error text of a runtime error from a PRO tip script.

-LastErrorOpsMgrString <String>

Specifies an array of strings used to provide localized error text. Th

e first element of the array should be the GUID of the OpsMgr string an

d the following elements should be the parameters for string formatting

.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-TipStatus

Specifies the current status of a PRO tip object.

VALID VALUE DESCRIPTION

----------- -----------

Active The user can invoke the tip's recommended action.

Initialized The tip has been invoked; any incomplete jobs are queued.

Running The tip has been invoked; its jobs are running

Resolved The implementation of the tip has completed successfully.

Failed The implementation of the tip has failed.

Dismissed The user has chosen to ignore the tip.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Set the status of a PRO tip.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $AllPROTips = Get-PROTip

PS C:\> Set-PROTip -PROTipID $AllPROTips[0].Id -TipStatus Running

The first command connects to VMMServer1 and gets the server object from th

e VMM database. The following commands use this server by default.

The second command gets all objects that represent active PRO tips from the

VMM database and stores the PRO tip objects in variable $AllPROTips (an ob

ject array).

The last command updates the first tip ($AllPROTips[0]) returned from the p

receding command to the status "Running".

REMARKS

For more information, type: "get-help Set-PROTip -detailed".

For technical information, type: "get-help Set-PROTip -full".

## Script

### Get-Script

SYNOPSIS

Gets script objects from the Virtual Machine Manager library. With appropri

ate permissions, you can also use Get-Script to view or edit any script, or

to view, edit, or run a Windows PowerShell script.

SYNTAX

Get-Script [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [<CommonPa

rameters>]

Get-Script [-All] [-VMMServer [<String ServerConnection>]] [<CommonParamete

rs>]

DETAILED DESCRIPTION

Gets script objects from the Virtual Machine Manager library. The script fi

le that a script object represents is stored in the file system on a librar

y server. Typically, these scripts are either Windows PowerShell scripts or

answer file scripts (including Sysprep.inf and Unattend.xml files, which c

ontain the inputs required for the Windows Setup program).

As illustrated in the examples, you can use Get-Script not only to retrieve

script objects but also (if you have appropriate permissions) to view the

contents of a script or to edit a script. In addition, if the script is a W

indows PowerShell script, if scripting is enabled on your server, and if yo

u have appropriate permissions, you can also run the script (see the exampl

es).

IMPORTANT: For information about enabling Windows PowerShell scripting on y

our server, type:

Get-Help about\_Signing

Get-Help Get-ExecutionPolicy -detailed

Get-Help Set-ExecutionPolicy -detailed

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all scripts stored on all VMM library servers.

PS C:\> Get-Script -VMMServer VMMServer1.Contoso.com

Gets from the VMM library on VMMServer1 all objects that represent scripts

stored in library shares on library servers, and displays information about

these scripts to the user.

2: Get the specified information about all scripts on a specific VMM librar

y server.

PS C:\> Get-Script -VMMServer VMMServer1.Contoso.com | where { $\_.LibrarySe

rver.Name -eq "FileServer01.Contoso.com" } | Format-List -property Name, Li

braryServer, SharePath

Gets from the library on VMMServer1 all objects that represent scripts stor

ed on library server FileServer01 and displays the specified information (n

ame, library server, and share path) about these scripts to the user.

3: Get all scripts with a specific name on any VMM library server.

PS C:\> Get-Script -VMMServer VMMServer1.Contoso.com | where { $\_.Name -eq

“Sysprep.inf” }

Gets from the library on VMMServer1 all objects that represent answer file

scripts named Sysprep.inf that are stored on any library server.

NOTE: In VMM 2008, by default, the name of a script object in the VMM libra

ry is the same name (including the file extension) as the name of the actua

l script file on the library server.

4: View a script that is stored in the VMM library.

PS C:\> $SummaryScript = Get-Script | where { $\_.Name -eq "SummarizeVMMInfo

rmation.ps1"}

PS C:\> Notepad.exe $SummaryScript.SharePath

The first command gets the object that represents a script named "Summarize

VMMInformaton.ps1" from the VMM library and stores the script object in $Su

mmaryScript.

The second command uses Notepad to open the script so that you can view its

contents (if you have the appropriate permissions to read the script).

NOTE: If you have appropriate write permissions, you can also edit the scri

pt and save the new version.

5: Run a Windows PowerShell script that is stored in the VMM library.

PS C:\> $SummaryScript = Get-Script | where { $\_.Name -eq "SummarizeVMMInfo

rmation.ps1" }

PS C:\> &$SummaryScript.SharePath

The first command gets the object that represents a script named "Summarize

VMMInformaton.ps1" from the VMM library and stores the script object in $Su

mmaryScript.

The second command uses the "&" operator to run the script stored in $Summa

ryScript.

To run a Windows PowerShell script stored in a VMM 2008 library share:

\* You must have read and execute permissons on the script file.

\* You must be a member of the VMM Administrators user role.

For more information, type: Get-Help Set-VMMUserRole

\* You must have permissions to access the VMM library share.

For VMM 2008, the default location of the VMM Library share in the

file system on a server running Windows Server 2008 is:

<Drive>:\ProgramData\Virtual Machine Manager Library Files

\* If Windows PowerShell scripting is disabled, you must enable it:

1. Open Windows PowerShell - Virtual Machine manager by

right-clicking it and then clicking Run as administrator.

2. Use the Set-ExecutionPolicy cmdlet to set the execution

policy to the appropriate level for your environment.

For more information, type:

Get-Help about\_Signing

Get-Help Get-ExecutionPolicy -detailed

Get-Help Set-ExecutionPolicy -detailed

REMARKS

For more information, type: "get-help Get-Script -detailed".

For technical information, type: "get-help Get-Script -full".

### Remove-Script

SYNOPSIS

Removes a script object from Virtual Machine Manager.

SYNTAX

Remove-Script [-Script] <Script> [-Confirm] [-Force] [-JobVariable <String>

] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents one or more scripts from the Virtual Mach

ine Manager library and deletes the corresponding script file on the librar

y server.

If the script is attached to a template or hardware profile (and if you do

not use the Force parameter), VMM lists the container that contains the scr

ipt and prompts you to confirm that you want to remove the script:

\* If you reply Yes, VMM removes the association between the script and

the container to which it is attached, and then deletes the script

object from VMM.

\* If you reply No, the operation is cancelled.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-Script <Script>

Specifies a VMM script object.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a script object and delete the corresponding script file.

PS C:\> $Scripts = @(Get-Script -VMMServer VMMServer1.Contoso.com | where {

$\_.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -eq "Add

Host.ps1"} )

PS C:\> $Scripts.Count

PS C:\> $Scripts | select Name,SharePath | Format-List

PS C:\> Remove-Script -Script $Scripts[1] -Force

The first command retrieves from the VMM library on VMMServer1 the object t

hat represents the script file AddHost.ps1 (stored on library server FileSe

rver01) and stores the script object in variable $Script. Using the "@" sym

bol and parentheses ensures that the cmdlet stores the returned script obje

ct in an array (in case none exists or in case there is only one). More tha

n one file with the same name might exist if more than one container for sc

ripts exists on the specified library server.

The second command counts the number of scripts in $Script and displays the

results to the user.

The third command passes each script object in $Script to "select" (the Sel

ect-Object cmdlet), which selects the name and share path for each script i

n the array. The command then passes these results to the Format-List cmdle

t to display each script name, with its share path, to the user.

The last command uses the Remove-Script cmdlet to delete the second object

in the array ($Script[1]) and switches on the Force parameter to ensure tha

t the object that represents the script file is removed from the VMM databa

se. The command also deletes the corresponding script file from the file sy

stem on the library server.

2: Remove multiple scripts from the library.

PS C:\> $Scripts = Get-Script -VMMServer VMMServer1.Contoso.com | where { $

\_.Name -match “Sysprep” }

PS C:\> $Scripts | Remove-Script -Confirm

The first command gets from VMMServer1 all objects that represent scripts w

hose names include the string “Sysprep” and stores these script objects in

$Scripts (an object array).

The second command passes each script object in $Scripts to Remove-Script,

which removes each script object from the library and deletes each correspo

nding script file from the file system on the library server on which that

script is stored.

The Confirm parameter prompts you to confirm that you do want to remove the

se scripts. You have the option to confirm the deletion of all scripts at o

nce ("Yes to All" or "No to All") or to confirm the deletion of each script

one-by-one ("Yes" or "No").

REMARKS

For more information, type: "get-help Remove-Script -detailed".

For technical information, type: "get-help Remove-Script -full".

### Set-Script

SYNOPSIS

Changes properties of a script stored in the Virtual Machine Manager librar

y.

SYNTAX

Set-Script [-Script] <Script> [-Description <String>] [-Enabled <Boolean>]

[-JobVariable <String>] [-Name <String>] [-Owner <String>] [-PROTipID <Guid

>] [-RunAsynchronously] [-SharePath <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a script stored in the Virtual Machine Ma

nager library.

Properties that you can change include:

- Description

- Enabled

- Name

- Owner

- SharePath

Script objects represent script files stored in a library share on a librar

y server. Typically, these scripts are either Windows PowerShell scripts or

answer files (such as a Sysprep.inf or an Unattend.xml file) that contain

the inputs required for the Windows Setup program.

For examples that show how to view, edit, or run a Windows PowerShell scrip

t stored in the library, type:

Get-Help Get-Script -example

PARAMETERS

-Script <Script>

Specifies a VMM script object.

-Description <String>

Specifies a description for the specified object.

-Enabled <Boolean>

Enables a library object (when set to TRUE) or disables a library objec

t (when set to FALSE). For example, if you want to upgrade software on

a virtual machine template, you can disable the template object in the

VMM library to temporarily prevent users from using that object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the description of a script.

PS C:\> $Script = Get-Script -VMMServer VMMServer1.Contoso.com | where { $\_

.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -eq "Syspre

p.inf" }

PS C:\> Set-Script -Script $Script -Description “Windows Server 2003 Syspre

p Answer File”

The first command retrieves from the library on VMMServer1 the object that

represents the answer file script named Sysprep.inf (whose file is stored o

n the library server named FileServer01) and stores the script object in va

riable $Script.

The second command changes the description of this script object to “Window

s Server 2003 Sysprep Answer File”.

2: Disable a Windows PowerShell script stored in the VMM library.

PS C:\> $Script = Get-Script -VMMServer VMMServer1.Contoso.com | where { $\_

.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -eq "AddHos

t.ps1" }

PS C:\> Set-Script -Script $Script -Enabled $FALSE

The first command gets the object that represents the PowerShell script nam

ed AddHost.ps1 (whose file is stored on the library server named FileServer

01) and stores the script object in $Script.

The second command disables the script represented by $Script.

3: Specify an owner for all scripts with an "Unknown" owner

PS C:\> Get-Script -VMMServer "VMMServer1.Contoso.com" | where {$\_.Owner -e

q "Unknown"} | Set-Script -Owner "Contoso\FrankZheng"

Gets all script objects from the VMM library, selects only those scripts wh

ose owner is "Unknown", and specifies an owner for each script object.

REMARKS

For more information, type: "get-help Set-Script -detailed".

For technical information, type: "get-help Set-Script -full".

## SshPublicKey

### Get-SshPublicKey

SYNOPSIS

Gets the public key object from a VMware ESX Server that you want Virtual M

achine Manager to manage.

SYNTAX

Get-SshPublicKey [-ComputerName] <String> [-TCPPort <Int32>] [-VMMServer [<

String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets the Secure Shell (SSH) public key object from a VMware ESX Server that

you want Virtual Machine Manager to manage. SSH is a network protocol that

uses public key cryptography to enable secure encrypted communications bet

ween a local and a remote computer.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get the SSH public key from a specific VMware ESX host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $PublicKey = Get-SshPublicKey -ComputerName "nnn.nnn.nnn.nnn"

PS C:\> $PublicKey

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets the SSH public key object from a VMware ESX Server

(specified by its IP address, indicated in this example by "nnn.nnn.nnn.nnn

") and stores the public key object in variable $PublicKey.

The last command displays the contents of $PublicKey.

NOTE: To see an example that uses the public key object to associate an ESX

Server with Virtual Machine Manager, type: Get-Help Associate-VMHost -exam

ple

REMARKS

For more information, type: "get-help Get-SshPublicKey -detailed".

For technical information, type: "get-help Get-SshPublicKey -full".

## Step

### Get-Step

SYNOPSIS

Gets the steps for the specified job on a Virtual Machine Manager server.

SYNTAX

Get-Step [-Job] [<String Task>] [-Name <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets the steps for the specified job or jobs on a Virtual Machine Manager s

erver.

A job is composed of one or more steps, each of which has its own status. A

n earlier step must complete or be skipped before the next step runs.

PARAMETERS

-Job [<String Task>]

Specifies a VMM job object.

-Name <String>

Specifies the name of a VMM object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all steps for the specified job.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Job = Get-Job | where { $\_.ResultName -eq "VM01" -and Status -eq "

Running" }

PS C:\> Get-Step -Job $Job

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command performs the following operations:

\* Retrieves from the VMM database all objects that represent VMM jobs.

\* Passes these job objects to "where" (the Where-Object cmdlet), which

selects from the job results only those jobs on VM01 that are

currently running.

\* Stores the retrieved job objects in variable $Job.

The last command gets all objects that represent the top-most step objects

for the jobs stored in $Job and displays information about these steps to t

he user.

REMARKS

For more information, type: "get-help Get-Step -detailed".

For technical information, type: "get-help Get-Step -full".

## Template

### Get-Template

SYNOPSIS

Gets virtual machine template objects from the Virtual Machine Manager libr

ary.

SYNTAX

Get-Template [-All] [-VMMServer [<String ServerConnection>]] [<CommonParame

ters>]

Get-Template [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [<Common

Parameters>]

DETAILED DESCRIPTION

Gets from the Virtual Machine Manager library all objects that represent vi

rtual machine templates.

For information about how virtual machine templates are used to create new

virtual machines, type:

Get-Help New-Template -detailed

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all templates stored in the library.

PS C:\> Get-Template -VMMServer VMMServer1.Contoso.com

Gets all objects that represent templates from the VMM library on VMMServer

1, and displays information about these templates to the user.

2: Get all templates stored in the library that have a similar name.

PS C:\> Get-Template -VMMServer VMMServer1.Contoso.com | where { $\_.Name -l

ike "Windows 2000\*" }

Gets from the VMM library on VMMServer1 all objects that represent template

s whose name begins with "Windows 2000" and displays information about thes

e templates to the user.

REMARKS

For more information, type: "get-help Get-Template -detailed".

For technical information, type: "get-help Get-Template -full".

### New-Template

SYNOPSIS

Creates a virtual machine template used to create virtual machines managed

by Virtual Machine Manager.

SYNTAX

New-Template [-Name] <String> [-AdminPasswordCredential <PSCredential>] [-A

nswerFile <Script>] [-ComputerName <String>] [-CPUCount <Int32>] [-CPUType

[<ProcessorType String>]] [-Description <String>] [-FullName <String>] [-Gu

estOSProfile [<GuestOSProfile String>]] [-GuiRunOnceCommands <String[]>] [-

HardwareProfile <HardwareProfile>] [-HighlyAvailable <Boolean>] [-JobGroup

<Guid>] [-JobVariable <String>] [-JoinDomain <String>] [-JoinDomainCredenti

al <PSCredential>] [-JoinWorkgroup <String>] [-MemoryMB <Int32>] [-NoCustom

ization] [-OperatingSystem <OperatingSystem>] [-OrgName <String>] [-Owner <

String>] [-ProductKey <String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>

] [-RunAsynchronously] [-TimeZone <Int32>] [-VMMServer [<String ServerConne

ction>]] [<CommonParameters>]

New-Template [-Name] <String> -Template [<Template String>] [-AdminPassword

Credential <PSCredential>] [-AnswerFile <Script>] [-ComputerName <String>]

[-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-Description <Stri

ng>] [-FullName <String>] [-GuestOSProfile [<GuestOSProfile String>]] [-Gui

RunOnceCommands <String[]>] [-HardwareProfile <HardwareProfile>] [-HighlyAv

ailable <Boolean>] [-JobGroup <Guid>] [-JobVariable <String>] [-JoinDomain

<String>] [-JoinDomainCredential <PSCredential>] [-JoinWorkgroup <String>]

[-MemoryMB <Int32>] [-OperatingSystem <OperatingSystem>] [-OrgName <String>

] [-Owner <String>] [-ProductKey <String>] [-PROTipID <Guid>] [-RelativeWei

ght <Int32>] [-RunAsynchronously] [-TimeZone <Int32>] [<CommonParameters>]

New-Template [-Name] <String> -LibraryServer [<String LibraryServer>] -Shar

ePath <String> -VM [<String VM>] [-AdminPasswordCredential <PSCredential>]

[-AnswerFile <Script>] [-BootVirtualHardDisk <Boolean>] [-ComputerName <Str

ing>] [-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-Description

<String>] [-FullName <String>] [-GuestOSProfile [<GuestOSProfile String>]]

[-GuiRunOnceCommands <String[]>] [-HardwareProfile <HardwareProfile>] [-Hi

ghlyAvailable <Boolean>] [-JobGroup <Guid>] [-JobVariable <String>] [-JoinD

omain <String>] [-JoinDomainCredential <PSCredential>] [-JoinWorkgroup <Str

ing>] [-MemoryMB <Int32>] [-NoCustomization] [-OperatingSystem <OperatingSy

stem>] [-OrgName <String>] [-Owner <String>] [-ProductKey <String>] [-PROTi

pID <Guid>] [-RelativeWeight <Int32>] [-RunAsynchronously] [-SkipRearm] [-T

imeZone <Int32>] [<CommonParameters>]

New-Template [-Name] <String> -VirtualHardDisk <VirtualHardDisk> [-AdminPas

swordCredential <PSCredential>] [-AnswerFile <Script>] [-ComputerName <Stri

ng>] [-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-Description

<String>] [-FullName <String>] [-GuestOSProfile [<GuestOSProfile String>]]

[-GuiRunOnceCommands <String[]>] [-HardwareProfile <HardwareProfile>] [-Hig

hlyAvailable <Boolean>] [-JobGroup <Guid>] [-JobVariable <String>] [-JoinDo

main <String>] [-JoinDomainCredential <PSCredential>] [-JoinWorkgroup <Stri

ng>] [-MemoryMB <Int32>] [-NoCustomization] [-OperatingSystem <OperatingSys

tem>] [-OrgName <String>] [-Owner <String>] [-ProductKey <String>] [-PROTip

ID <Guid>] [-RelativeWeight <Int32>] [-RunAsynchronously] [-TimeZone <Int32

>] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual machine template that administrators or self-service user

s can use easily to create virtual machines managed by Virtual Machine Mana

ger. A template stores hardware configuration and guest operating system in

formation so that a specific template can be used repeatedly to create new

virtual machines. The New-Template cmdlet stores the new template object in

the Virtual Machine Manager library.

You can create a template based on an existing virtual hard disk, based on

an existing template, or from a virtual machine currently deployed on a vir

tual machine host. If you specify no parameters, Virtual Machine Manager cr

eates a default template object.

VMM 2008 TEMPLATE REQUIREMENTS

------------------------------

The following are the requirements for creating a template in VMM 2008:

\* TEMPLATE FROM VHD. If you create a template from a virtual hard

disk, the virtual hard disk must meet the following requirements:

- OS. You must install a supported Windows operating system on the

virtual hard disk before you use it to create the template.

Supported operating systems include:

Windows Server 2008

Windows Server 2003

Windows 2000 Server

Windows 2000 Advanced Server

Windows Vista

Windows XP Professional

- SYSPREP. You must run the System Preparation (Sysprep.exe) tool on

the virtual hard disk to ensure that every copy of the operating

system is unique when you distribute it to multiple virtual machines.

You can find Sysprep on the system CD of the Windows operating

systems listed earlier. Navigate to the Support\Tools folder, and

then open Deploy.cab.

- LOCAL ADMIN PASSWORD. The local Administrator password of the guest

operating system on a virtual hard disk that will be used to create a

new template must be blank before you run Sysprep.exe on the virtual

hard disk. A blank local Administrator password is required to enable

you to have the option to specify the local Administrator password

when you customize the guest operating system on the template.

\* TEMPLATE FROM VIRTUAL MACHINE. If you create a template from a virtual

machine, the virtual machine will be destroyed during the process of

converting it to a template. If you want to keep the virtual machine

and also use it to create a template, you can use the New-P2V cmdlet

(or the "Clone virtual machine" action in the Administrator Console)

to clone the virtual machine before you create a template.

\* TEMPLATE FOR SELF-SERVICE USERS. If a self-service user role includes

permission to use a template, the self-service user cannot change any

hardware profile settings. The only settings that a self-service user

can change when using a template to create a virtual machine are

computer name and, if the user has appropriate privileges, password

and product ID number.

VMM 2008 SUPPORT FOR CUSTOMIZABLE OR NON-CUSTOMIZABLE TEMPLATES

---------------------------------------------------------------

Optionally, if you specify the NoCustomization parameter with the New-Templ

ate cmdlet to create a template from a virtual hard disk, from a virtual ma

chine, from a template, or from a blank virtual hard disk, you do not need

to add a guest operating system profile to the template as you create the t

emplate. Without a guest operating system profile, Virtual Machine Manager

will not require Sysprep to run within the guest operating system when a vi

rtual machine that is created by using this template is deployed on a host

One possible scenario is that you have a manually sysprepped virtual hard d

isk that contains a non-Windows operating system (or that contains an opera

ting system that Virtual Machine Manager cannot automatically sysprep) and

have embedded an answer file in that virtual hard disk that contains the ap

propriate settings for that operating system. You can use the New-Template

cmdlet to create a template from this virtual hard disk and specify that th

e template does not allow customization of the guest operating system. To d

o so, use the NoCustomization flag when you create the new template.

In another scenario, you might import a VMware-based template that contains

a Windows-based operating system into Virtual Machine Manager. By default,

Virtual Machine Manager imports a VMware-based template as customizable if

Virtual Machine Manager knows how to customize the guest operating system.

Otherwise, the template is imported as non-customizable (this is true, for

example, for Linux guest operating systems). If you want to create a non-c

ustomizable Windows-based template from this imported, customizable VMware-

based template, you can use the New-Template cmdlet to create a new templat

e, point to the disks that are attached to the imported template, and then

specify the –NoCustomization flag on the new template.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VirtualHardDisk <VirtualHardDisk>

Specifies a virtual hard disk object.

-VM [<String VM>]

Specifies a virtual machine object.

-AdminPasswordCredential <PSCredential>

Specifies the password for the local Administrator account. Specifying

a password (on a new or existing template, on a new or existing guest

operating system profile, or on a new virtual machine) overrides any ex

isting Administrator password.

-AnswerFile <Script>

Specifies a script object stored in the VMM library to use as an answer

file. The name of the answer file script depends on the operating syst

em that you want to install on a virtual machine:

ANSWER FILE GUEST OS TO INSTALL ON VM

----------- -------------------------

Sysprep.inf Windows XP, Windows Server 2000, or Windows Server 2003

Unattend.xml Windows Vista or Windows Server 2008

Note: In VMM 2007, this parameter was named SysPrepFile.

-BootVirtualHardDisk <Boolean>

Specifies the virtual hard disk file that contains the operating system

file to use when you create a new template.

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-Description <String>

Specifies a description for the specified object.

-FullName <String>

Specifies the name of the person in whose name a virtual machine is reg

istered.

-GuestOSProfile [<GuestOSProfile String>]

Specifies a guest operating system profile object.

-GuiRunOnceCommands <String[]>

Specifies one or more commands to add to the [GuiRunOnce] section of an

unattended answer file (such as SysPrep.inf or Unattend.xml). Use sing

le quotes around each string enclosed in double quotes.

Example:

-GuiRunOnceCommands '"C:\APF\APFPostSysPrepCopy.cmd PARAMS1"', '"C:\APF

\APFPostSysPrepCopy.cmd PARAMS1"'

For information about how Windows PowerShell uses quotes, type: Get-Hel

p about\_Quoting\_Rules

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-HighlyAvailable <Boolean>

Specifies that a virtual machine will be placed on a Hyper-V host that

is part of a host cluster. Configure this setting on a virtual machine,

or on a template or hardware profile that will be used to create virtu

al machines.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-JoinDomain <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the do

main to which you want to join a virtual machine. You can use this para

meter to override the existing value on a template or on a guest operat

ing system profile. You can join a VM to a domain only if a virtual net

work adapter is configured for the VM.

-JoinDomainCredential <PSCredential>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the user name and

password of an account with permission to join a virtual machine to the

domain. A limited rights account should be used for joining machines t

o the domain.

-JoinWorkgroup <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the wo

rkgroup to which you want to join a virtual machine. You can use this p

arameter to override the existing value on a template or on a guest ope

rating system profile.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-NoCustomization

Specifies that guest operating system settings on this template cannot

be customized.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-OrgName <String>

Specifies the name of the organization of the person in whose name a vi

rtual machine is registered.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-ProductKey <String>

Specifies the product key to use for the operating system to be install

ed on a virtual machine. The product key is a 25-digit number that iden

tifies the product license.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SkipRearm

Skips running the Windows Software Licensing Rearm program. This progra

m restores the Windows operating system to its original, out-of-box lic

ensing state.

IMPORTANT: Please refer to your licensing agreements with regard to act

ivation and the Rearm program. Using this parameter might violate your

licensing agreements.

-TimeZone <Int32>

Specifies a number (an "index") that identifies a geographical region t

hat shares the same standard time. For a list of time zone indexes, see

"Microsoft Time Zone Index Values" at: http://go.microsoft.com/fwlink/

?LinkId=120935. If no time zone is specified, the default time zone use

d for a virtual machine is the same time zone setting that is on the vi

rtual machine host.

Example: To specify the GMT Standard Time zone, type: -TimeZone 085

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a default template from a virtual hard disk.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $OS = Get-OperatingSystem -VMMServer "VMMServer1.Contoso.com" | whe

re {$\_.Name -eq "64-bit Edition of Windows Server 2008 Datacenter"}

PS C:\> $VHD = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | wher

e { $\_.Name -eq “VHD01” -and $\_.LibraryServer.Name -eq "FileServer01.Conto

so.com” }

PS C:\> New-Template -Name "NewTemplate1" -VirtualHardDisk $VHD -OperatingS

ystem $OS -NoCustomization

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents a specific operating sys

tem (64-bit edition of Windows Server 2008 Datacenter) from the VMM databas

e and stores the operating system object in variable $OS.

The third command gets the object that represents the virtual hard disk fil

e named VHD01 (stored on library server FileServer01) from the VMM library

and stores the virtual hard disk object in variable $VHD.

The last command creates a template named NewTemplate1 from VHD01 and speci

fies the name of the operating system. No customization is made to the oper

ating system.

NOTE: This example assumes that VHD01 is a sysprepped virtual hard disk on

which the 64-bit edition of the Windows Server 2008 Datacenter operating sy

stem is installed. You can, optionally, install virtualization guest servic

es (that is, Integration Components on a Hyper-V host or Virtual Machine Ad

ditions on a Virtual Server host) on the virtual machine, or Virtual Machin

e Manager will install them automatically when the virtual machine is deplo

yed on a Windows-based host.

2: Create a template from an existing virtual machine.

PS C:\> $LibraryServer = Get-LibraryServer -VMMServer "VMMServer1.Contoso.c

om" | where {$\_.Name -eq "FileServer01.Contoso.com"}

PS C:\> $VM = Get-VM -VMMServer "VMMServer1.Contoso.com" -Name "VM01" | whe

re {$\_.VMHost.Name -eq "VMHost02.Contoso.com"}

PS C:\> $OperatingSystem = Get-OperatingSystem -VMMServer "VMMServer1.Conto

so.com" | where {$\_.Name -eq "Windows Vista"}

PS C:\> New-Template -Name "Template1" -RunAsynchronously -VM $VM -Owner "C

ontoso\User1" -LibraryServer $LibraryServer -SharePath "\\FileServer01.Cont

oso.com\MSSCVMMLibrary" -OperatingSystem $OperatingSystem -NoCustomization

The first command gets the object that represents the library server called

FileServer01 from VMMServer1 and stores the library server object in $Libr

aryServer.

The second command gets the object that represents the virtual machine name

d VM01 (which is currently deployed on VMHost02) and stores the virtual mac

hine object in $VM.

NOTE: VM01, which is the virtual machine that will be converted to a templa

te, will be destroyed during the conversion process. (If you want to retain

the virtual machine used to create a template, you can use the New-VM cmdl

et to clone the virtual machine before you create the template.)

The fourth command gets the object that represents a specific operating sys

tem (Windows Vista) and stores the operating system object in $OS.

The last command creates a template named Template1 from VM01. It specifies

the owner of the new template; specifies the library server and share wher

e you want to store the new template; and specfies the name of the operatin

g system. The -RunAsynchronously parameter returns control to the shell imm

ediately (before the command completes) without any customization to the op

erating system.

3: Create a template from a virtual hard disk with the specified characteri

stics.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $JobGroupId1 = [Guid]::NewGuid().ToString()

PS C:\> New-VirtualNetworkAdapter -JobGroup $JobGroupID1 -PhysicalAddressTy

pe Dynamic -VirtualNetwork "Internal Network"

PS C:\> New-VirtualSCSIAdapter -JobGroup $JobGroupID1 -AdapterID 6 -Shared

$FALSE

PS C:\> New-VirtualDVDDrive -JobGroup $JobGroupID1 -Bus 1 -LUN 0

PS C:\> New-HardwareProfile -Name "NewProfile1" -Owner "CONTOSO\Nicholas" -

Description "Temporary hardware profile used to create a VM/Template" -Memo

ryMB 512 -JobGroup $JobGroupID1

PS C:\> $JobGroupId2 = [Guid]::NewGuid().ToString()

PS C:\> $VHD = Get-VirtualHardDisk | where {$\_.Location -eq "\\VMHost01Shar

e\VHDs\Template.vhd"} | where {$\_.HostName -eq "VMHost01.Contoso.com"}

PS C:\> New-VirtualDiskDrive -IDE -Bus 0 -LUN 0 -JobGroup $JobGroupID2 -Vir

tualHardDisk $VHD

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "NewProfile1

" }

PS C:\> $OS = Get-OperatingSystem | where {$\_.Name -eq "Windows Server 2003

Enterprise x64 Edition"}

PS C:\> New-Template -Name "NewTemplate2" -Owner "CONTOSO\Nicholas" -Hardwa

reProfile $HWProfile -JobGroup $JobGroupID2 -ComputerName "\*" -JoinWorkgrou

p "WORKGROUP" -OperatingSystem $OS -RunAsynchronously

The first command connects to VMMServer1.

The second command generates a globally unique identifier (GUID) and stores

the GUID string in variable $JobGroupID1. The job group ID functions as an

identifier that groups subsequent commands that include $JobGroupID1 into

a single job group.

The third command will create a virtual network adapter but uses the JobGro

up parameter to specify that the network adapter is not created until just

before the New-HardwareProfile cmdlet runs (in the sixth command). The New-

VirtualNetworkAdapter cmdlet sets the physical address type (MAC address ty

pe) to dynamic and specifies that the new virtual network adapter will conn

ect to a virtual network called "Internal Network."

The fourth command will create a virtual SCSI adapter but uses the JobGroup

parameter to specify that the SCSI adapter is not created until just befor

e the New-HardwareProfile cmdlet runs (in the sixth command). The New-Virtu

alSCSIAdapter cmdlet sets the adapter ID to 6 and sets the Shared parameter

to $FALSE so that the adapter will not be shared (as it would have had to

be if you wanted to use it in guest clustering).

The fifth command will create a virtual DVD drive but uses the JobGroup par

ameter to specify that the DVD drive is not created until just before the N

ew-HardwareProfile cmdlet runs (in the sixth command). The New-VirtualDVDDr

ive cmdlet specifies Bus 1 and LUN 0 to attach the virtual DVD drive to Sec

ondary Channel (0) on the IDE bus.

The sixth command creates a hardware profile named NewProfile1, sets the ow

ner to Nicholas (whose account is in the Contoso domain), specifies a descr

iption, and specifies that the amount of memory on the host that a virtual

machine (created by using this template) will use is 512 MB. The New-Hardwa

reProfile cmdlet uses the JobGroup parameter to specify that all preceding

commands that include variable $JobGroupID1 will run just before New-Hardwa

reProfile creates the new hardware profile. After New-VirtualNetworkAdapter

, New-VirtualSCSIAdapter, and New-VirtualDVDDrive run, the resulting object

s that are created are automatically associated with the new hardware profi

le.

The seventh command generates a new GUID and stores it in $JobGroupID2. Thi

s job group ID will be used to identify any subsequent commands that includ

e this ID and will delay the execution of those commands until just before

the last command that specifies $JobGroupID2 runs.

The eighth command uses the Get-VirtualHardDisk cmdlet to get the object th

at represents the file named Template.vhd (stored in the VHDs folder under

VMHost01Share on VMHost01) and stores the virtual hard disk object in $VHD.

The ninth command will create a new virtual disk drive and attach the virtu

al hard disk stored in $VHD (Template.vhd) to this new virtual disk drive.

The command specifies Bus 0 and LUN 0 on the IDE Bus so that Template.vhd w

ill be attached to the first slot (0) of the Primary Channel (0) on the IDE

bus of the new virtual disk drive. The command uses the JobGroup parameter

to specify that the new virtual disk drive is not created until just befor

e the New-Template cmdlet runs (in the eleventh command).

The tenth command gets the object that represents the hardware profile name

d "NewProfile1" from the VMM library and stores the hardware profile object

in $HWProfile.

The eleventh command gets the object that represents a specific operating s

ystem (Windows Server 2003 Enterprise x64 Edition) and stores the operating

system object in variable $OS.

The last command creates a template named NewTemplate2, sets the owner to N

icholas (whose account is in the Contoso domain), specifies that this templ

ate will use the hardware profile named NewProfile1, sets the computer name

to be randomly generated (indicated by the asterisk “\*”), and specifies th

at any virtual machine created by using this template will be joined to the

workgroup called WORKGROUP. The New-Template cmdlet uses the JobGroup para

meter to specify that all preceding commands that include variable $JobGrou

pID2 will run before New-Template creates the new template. After Add-Virtu

alHardDisk (the only command before this command that includes $JobGroupID2

) runs, the resulting virtual hard disk object that is created is automatic

ally associated with the new template.

REMARKS

For more information, type: "get-help New-Template -detailed".

For technical information, type: "get-help New-Template -full".

### Remove-Template

SYNOPSIS

Removes a template object from Virtual Machine Manager and deletes all file

s associated with this template.

SYNTAX

Remove-Template [-Template] [<Template String>] [-Confirm] [-Force] [-JobVa

riable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters

>]

DETAILED DESCRIPTION

Removes an object that represents a template from the Virtual Machine Manag

er library and deletes all files associated with this template.

The types of file that can be associated with a template include virtual ha

rd disk files (Windows-based .vhd files or VMware-based .vmdk files), virtu

al floppy disk files (Windows-based .vfd files or VMware-based .flp files),

and script files (Windows PowerShell .ps1 script files or answer file scri

pts, including Sysprep.inf and Unattend.xml files).

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific templateobject from the library and delete the corresp

onding files on the library server.

PS C:\> $Template = Get-Template -VMMServer VMMServer1.Contoso.com | where

{ $\_.Name -eq "Template1" }

PS C:\> Remove-Template -Template $Template

The first command gets the object that represents the template named Templa

te1 from the library on VMMServer1 and stores the template object in variab

le $Template.

The second command removes the Template1 object from the library and delete

s all files associated with this template from the file system on the libra

ry server.

2: Remove all templates from the library.

PS C:\> $Templates = Get-Template -VMMServer VMMServer1.Contoso.com

PS C:\> $Templates | Remove-Template -Confirm

The first command gets all the template objects from VMMServer1 and stores

the template objects in $Templates (an object array).

The second command passes each template object in $Templates to Remove-Temp

late, which removes each template object from the VMM library. The command

also deletes all files associated with each template from the file system o

n the library server on which that template is stored. The Confirm paramete

r prompts you to confirm whether you want to delete each template.

REMARKS

For more information, type: "get-help Remove-Template -detailed".

For technical information, type: "get-help Remove-Template -full".

### Set-Template

SYNOPSIS

Changes properties of a template used in Virtual Machine Manager.

SYNTAX

Set-Template [-Template] [<Template String>] -JobGroup <Guid> [-AdminPasswo

rdCredential <PSCredential>] [-AnswerFile <Script>] [-BootOrder <BootDevice

[]>] [-ComputerName <String>] [-CostCenter <String>] [-CPUCount <Int32>] [-

CPUMax <Int32>] [-CPUReserve <Int32>] [-CPUType [<ProcessorType String>]] [

-Custom1 <String>] [-Custom10 <String>] [-Custom2 <String>] [-Custom3 <Stri

ng>] [-Custom4 <String>] [-Custom5 <String>] [-Custom6 <String>] [-Custom7

<String>] [-Custom8 <String>] [-Custom9 <String>] [-Description <String>] [

-DiskIO <Int32>] [-Enabled <Boolean>] [-ExcludeFromPRO <Boolean>] [-Expecte

dCPUUtilization <Int32>] [-FullName <String>] [-GuiRunOnceCommands <String[

]>] [-HighlyAvailable <Boolean>] [-JobVariable <String>] [-JoinDomain <Stri

ng>] [-JoinDomainCredential <PSCredential>] [-JoinWorkgroup <String>] [-Lim

itCPUFunctionality <Boolean>] [-MemoryMB <Int32>] [-Name <String>] [-Networ

kUtilization <Int32>] [-NumLock] [-OperatingSystem <OperatingSystem>] [-Org

Name <String>] [-Owner <String>] [-ProductKey <String>] [-PROTipID <Guid>]

[-QuotaPoint <Int32>] [-RelativeWeight <Int32>] [-RunAsynchronously] [-Tag

<String>] [-TimeZone <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a template used in a Virtual Machine Mana

ger environment.

Properties that you can change include settings for the name of the templat

e, its description, owner, organization name, time zone, operating system,

and product key; BIOS boot order; CPU number and characteristics; the amoun

t memory on the host that is assigned to a virtual machine; the amount of b

andwidth on the host's network available to a virtual machine; whether or n

ot a virtual machine created with this template will be highly available (t

hat is, deployed on a host that is part of a host cluster); values for doma

in or workgroup as well as domain credentials; an optional point quota that

specifies how many new virtual machines self-service users can create; cus

tom fields, and other options.

Changes that you make to a template affect only the template. Changes do no

t affect any existing virtual machines that were created earlier by using t

his template.

If you want to change the properties of a a virtual disk drive, virtual flo

ppy drive, virtual DVD drive, virtual network adapter, virtual SCSI adapter

or virtual COM port associated with a specific template, you can use Set-V

irtualDiskDrive, Set-VirtualFloppyDrive, Set-VirtualDVDDrive, Set-VirtualNe

tworkAdapter, Set-VirtualSCSIAdapter, or Set-VirtualCOMPort, respectively.

For more information about templates, type:

Get-Help New-Template -detailed

PARAMETERS

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-AdminPasswordCredential <PSCredential>

Specifies the password for the local Administrator account. Specifying

a password (on a new or existing template, on a new or existing guest

operating system profile, or on a new virtual machine) overrides any ex

isting Administrator password.

-AnswerFile <Script>

Specifies a script object stored in the VMM library to use as an answer

file. The name of the answer file script depends on the operating syst

em that you want to install on a virtual machine:

ANSWER FILE GUEST OS TO INSTALL ON VM

----------- -------------------------

Sysprep.inf Windows XP, Windows Server 2000, or Windows Server 2003

Unattend.xml Windows Vista or Windows Server 2008

Note: In VMM 2007, this parameter was named SysPrepFile.

-BootOrder <BootDevice[]>

Specifies the order of devices that a virtual machine on a Hyper-V host

uses to start up.

Valid values: CD, IDEHardDrive, PXEBoot, or Floppy.

Example: -BootOrder PXEBoot,IDEHardDrive,CD,Floppy

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-CostCenter <String>

Specifies the cost center for a virtual machine so that you can collect

data about the allocation of virtual machines (or resources allocated

to virtual machines) to make use of in your billing system.

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUMax <Int32>

Specifies the highest percentage of the total resources of a single CPU

on the host that can be used by a specific virtual machine at any give

n time.

Example: -CPUMax 80 (to specify 80 per cent)

-CPUReserve <Int32>

Specifies the minimum percentage of the resources of a single CPU on th

e host to allocate to a virtual machine. The percentage of CPU capacity

that is available to the virtual machine is never less than this perce

ntage.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-Custom1 <String>

Specifies a custom property on a VMM object.

-Custom10 <String>

Specifies a custom property on a VMM object.

-Custom2 <String>

Specifies a custom property on a VMM object.

-Custom3 <String>

Specifies a custom property on a VMM object.

-Custom4 <String>

Specifies a custom property on a VMM object.

-Custom5 <String>

Specifies a custom property on a VMM object.

-Custom6 <String>

Specifies a custom property on a VMM object.

-Custom7 <String>

Specifies a custom property on a VMM object.

-Custom8 <String>

Specifies a custom property on a VMM object.

-Custom9 <String>

Specifies a custom property on a VMM object.

-Description <String>

Specifies a description for the specified object.

-DiskIO <Int32>

Specifies the number of disk input/output operations per second (IOPS)

on the host that can be used by a specific virtual machine.

Example: -DiskIO 1500 (to specify 1500 IOPS).

-Enabled <Boolean>

Enables a library object (when set to TRUE) or disables a library objec

t (when set to FALSE). For example, if you want to upgrade software on

a virtual machine template, you can disable the template object in the

VMM library to temporarily prevent users from using that object.

-ExcludeFromPRO <Boolean>

Excludes (when set to TRUE) this virtual machine from being changed by

implementing host-targeted PRO tips.

-ExpectedCPUUtilization <Int32>

Specifies (as a percentage) the amount of CPU on the host that you expe

ct this virtual machine to use. This value is used only when VMM determ

ines a suitable host for the virtual machine.

-FullName <String>

Specifies the name of the person in whose name a virtual machine is reg

istered.

-GuiRunOnceCommands <String[]>

Specifies one or more commands to add to the [GuiRunOnce] section of an

unattended answer file (such as SysPrep.inf or Unattend.xml). Use sing

le quotes around each string enclosed in double quotes.

Example:

-GuiRunOnceCommands '"C:\APF\APFPostSysPrepCopy.cmd PARAMS1"', '"C:\APF

\APFPostSysPrepCopy.cmd PARAMS1"'

For information about how Windows PowerShell uses quotes, type: Get-Hel

p about\_Quoting\_Rules

-HighlyAvailable <Boolean>

Specifies that a virtual machine will be placed on a Hyper-V host that

is part of a host cluster. Configure this setting on a virtual machine,

or on a template or hardware profile that will be used to create virtu

al machines.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-JoinDomain <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the do

main to which you want to join a virtual machine. You can use this para

meter to override the existing value on a template or on a guest operat

ing system profile. You can join a VM to a domain only if a virtual net

work adapter is configured for the VM.

-JoinDomainCredential <PSCredential>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the user name and

password of an account with permission to join a virtual machine to the

domain. A limited rights account should be used for joining machines t

o the domain.

-JoinWorkgroup <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the wo

rkgroup to which you want to join a virtual machine. You can use this p

arameter to override the existing value on a template or on a guest ope

rating system profile.

-LimitCPUFunctionality <Boolean>

Enables running an older operating system (such as Windows NT 4.0) on a

virtual machine deployed on a Hyper-V host or on a VMware ESX host by

providing only limited CPU functionality for the virtual machine.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-Name <String>

Specifies the name of a VMM object.

-NetworkUtilization <Int32>

Specifies, in megabits per second (Mb/s), the amount of bandwidth on th

e host's network that can be used by a specific virtual machine.

Example: -NetworkUtilization 10 (to specify 10 Mb/s)

-NumLock

Enables the BIOS value for NumLock on a virtual machine (or on a templa

te or hardware profile that is used to create virtual machines) on a Hy

per-V host. This parameter does not apply to virtual machines on Virtua

l Server hosts or on VMware ESX hosts.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-OrgName <String>

Specifies the name of the organization of the person in whose name a vi

rtual machine is registered.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-ProductKey <String>

Specifies the product key to use for the operating system to be install

ed on a virtual machine. The product key is a 25-digit number that iden

tifies the product license.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-QuotaPoint <Int32>

Specifies a quota that limits the number of virtual machines self-servi

ce users can create.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-Tag <String>

Associates a word or phrase with a virtual machine (or a template used

to create virtual machines) so that you can search for all virtual mach

ines with that tag as a set.

-TimeZone <Int32>

Specifies a number (an "index") that identifies a geographical region t

hat shares the same standard time. For a list of time zone indexes, see

"Microsoft Time Zone Index Values" at: http://go.microsoft.com/fwlink/

?LinkId=120935. If no time zone is specified, the default time zone use

d for a virtual machine is the same time zone setting that is on the vi

rtual machine host.

Example: To specify the GMT Standard Time zone, type: -TimeZone 085

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Specify an amount of memory for an existing template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where {$\_.Name -eq "NewTemplate1"}

PS C:\> Set-Template -Template $Template -MemoryMB 1024

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the template named NewTemplate1 from the VMM librar

y and stores the template object in variable $Template.

The last command changes the memory value for NewTemplate1 to 1024 MB. This

is the amount of memory on the host that a virtual machine (created by usi

ng this template) will use.

2: Specify a new owner for multiple highly available templates.

PS C:\> Get-VMMServer "VMMServer01.contoso.com"

PS C:\> $Templates = Get-Template | where {$\_.IsHighlyAvailable}

PS C:\> foreach ($Template in $Templates) {Set-Template $Template -Owner "C

ontoso\<NewOwnerUserName>"}

The first command connects to VMMServer1.

The second command gets a list of template objects that match the search cr

iteria (in this example, highly available templates) and stores the templat

e objects in $Templates (an object array).

The third command uses a foreach loop to specify a new owner for each of th

e templates in the array.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

3: Specify an owner for all templates with an "Unknown" owner.

PS C:\> Get-Template -VMMServer "VMMServer1.Contoso.com" | where {$\_.Owner

-eq "Unknown"} | Set-Template -Owner "Contoso\DavidYalovsky"

Gets all template objects from the VMM library, selects only those objects

whose owner is "Unknown," and specifies an owner for each template object.

REMARKS

For more information, type: "get-help Set-Template -detailed".

For technical information, type: "get-help Set-Template -full".

## V2V

### New-V2V

SYNOPSIS

Converts a virtual machine created on a VMware ESX Server host to a virtual

machine deployed on a Windows-based host (Hyper-V or Virtual Server) manag

ed by Virtual Machine Manager.

SYNTAX

New-V2V -VMHost [<String Host>] -VMXMachineConfig <VMXMachineConfig> [-CPUC

ount <Int32>] [-CPUType [<ProcessorType String>]] [-DelayStart <Int32>] [-D

escription <String>] [-JobGroup <Guid>] [-JobVariable <String>] [-LibrarySe

rver [<String LibraryServer>]] [-MemoryMB <Int32>] [-Name <String>] [-Netwo

rkLocation <String>] [-NetworkTag <String>] [-NoConnection] [-OverridePatch

Path <String>] [-Owner <String>] [-Path <String>] [-PhysicalAddress <String

>] [-PhysicalAddressType <String>] [-PROTipID <Guid>] [-RelativeWeight <Int

32>] [-RunAsSystem] [-RunAsUserCredential <PSCredential>] [-RunAsynchronous

ly] [-SkipInstallVirtualizationGuestServices] [-SourceNetworkConnectionID <

String>] [-StartAction <String>] [-StartVM] [-StopAction <String>] [-Trigge

r] [-UseHardwareAssistedVirtualization] [-VirtualNetwork <VirtualNetwork>]

[-VirtualNetworkAdapter <VirtualNetworkAdapter>] [-VLANEnabled] [-VLANID <I

nt32>] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-V2V -VM [<String VM>] -VMHost [<String Host>] [-CPUCount <Int32>] [-CPU

Type [<ProcessorType String>]] [-DelayStart <Int32>] [-Description <String>

] [-JobGroup <Guid>] [-JobVariable <String>] [-MemoryMB <Int32>] [-Name <St

ring>] [-NetworkLocation <String>] [-NetworkTag <String>] [-NoConnection] [

-OverridePatchPath <String>] [-Owner <String>] [-Path <String>] [-PhysicalA

ddress <String>] [-PhysicalAddressType <String>] [-PROTipID <Guid>] [-Relat

iveWeight <Int32>] [-RunAsSystem] [-RunAsUserCredential <PSCredential>] [-R

unAsynchronously] [-SkipInstallVirtualizationGuestServices] [-SourceNetwork

ConnectionID <String>] [-StartAction <String>] [-StartVM] [-StopAction <Str

ing>] [-Trigger] [-UseHardwareAssistedVirtualization] [-VirtualNetwork <Vir

tualNetwork>] [-VirtualNetworkAdapter <VirtualNetworkAdapter>] [-VLANEnable

d] [-VLANID <Int32>] [-VMMServer [<String ServerConnection>]] [<CommonParam

eters>]

New-V2V -VMHost [<String Host>] -VMXPath <String> [-CPUCount <Int32>] [-CPU

Type [<ProcessorType String>]] [-DelayStart <Int32>] [-Description <String>

] [-JobGroup <Guid>] [-JobVariable <String>] [-LibraryServer [<String Libra

ryServer>]] [-MemoryMB <Int32>] [-Name <String>] [-NetworkLocation <String>

] [-NetworkTag <String>] [-NoConnection] [-OverridePatchPath <String>] [-Ow

ner <String>] [-Path <String>] [-PhysicalAddress <String>] [-PhysicalAddres

sType <String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSystem]

[-RunAsUserCredential <PSCredential>] [-RunAsynchronously] [-SkipInstallVi

rtualizationGuestServices] [-SourceNetworkConnectionID <String>] [-StartAct

ion <String>] [-StartVM] [-StopAction <String>] [-Trigger] [-UseHardwareAss

istedVirtualization] [-VirtualNetwork <VirtualNetwork>] [-VirtualNetworkAda

pter <VirtualNetworkAdapter>] [-VLANEnabled] [-VLANID <Int32>] [-VMMServer

[<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Converts a virtual machine created on a VMware ESX Server host to a virtual

machine deployed on a Windows-based host (Hyper-V or Virtual Server) manag

ed by Virtual Machine Manager. You cannot specify a VMware ESX host as the

destination host for the new virtual machine.

VMM 2008 V2V CONVERSION REQUIREMENTS

------------------------------------

A V2V conversion requires that the host on which the new virtual machine wi

ll be deployed is a computer running either Windows Server 2008 with the Hy

per-V server role enabled, or a Windows-based server (typically, a computer

running Windows Server 2003) on which Virtual Server 2005 R2 SP1 version 1

.1.629.0 or later is installed. (For more information about virtual machine

hosts in VMM 2008, type "Get-Help Add-VMHost -detailed".)

The source for a V2V conversion of a VMware virtual machine performed by th

e New-V2V cmdlet is a set of files that you must store in the Virtual Machi

ne Manager library before you perform the conversion:

\* A .vmx file, which is a VMware virtual machine configuration file.

A .vmx file is approximately similar in function to the virtual

machine configuration file (.vmc file) used for a Windows-based

virtual machine.

A .vmx file is a text file that describes the properties and

structure of a virtual machine, including name, memory, disk

assignments, network parameters, and so on.

\* One or more .vmdk files. A .vmdk file is a VMware virtual hard

disk file, which is similar to the virtual hard disk file (.vhd

file) used for a Windows-based virtual machine.

The .vmdk files are not passed directly as input to the New-V2V

cmdlet but are listed in the .vmx file. A .vmdk file contains the

virtual machine's guest operating system, applications, and data.

Supported VMware virtual hard disk formats include:

- monolithicSparse

- monolithicFlat

- vmfs

- twoGbMaxExtentSparse

- twoGbMaxExtentFlat

VMM 2008 V2V CONVERSION PROCESS

-------------------------------

During the conversion process, the New-V2V cmdlet converts the .vmdk files

to .vhd files and makes the operating system on the new virtual machine com

patible with Microsoft virtualization technologies. The virtual machine cre

ated by New-V2V matches VMware virtual machine properties, including name,

description, memory, disk-to-bus assignment, and so on, unless these settin

gs are explicitly overridden by specifying different values for these setti

ngs. By default, the conversion process does not preserve network adapter s

ettings; however, you can explicitly set adapter settings on the target vir

tual machine.

VMM 2008 V2V CONVERSION OF THESE GUEST OPERATING SYSTEMS

--------------------------------------------------------

New-V2V supports the conversion of VMware virtual machines that are running

any of the following guest operating systems:

\* Microsoft Windows 2000 Server with Service Pack 4 (SP4) or later

\* Windows Server 2003 SP1 or later

\* Windows Server 2003 R2 or later

\* Windows Server 2008

\* Windows XP SP1 or later

\* Windows Vista

Some conversions of a VMware-based virtual machine whose guest operating sy

stem is Windows might require that additional system files and drivers be a

dded to the internal cache. You can use the Add-Patch cmdlet to add the req

uired files to the cache. To determine what patches you need to add, run Ne

w-V2V and let the cmdlet convert the .vmdk file to a .vhd file. If you need

patches, this process will put the V2V conversion into a failed state and

will produce a list of required patches. Next, use the Add-Patch cmdlet to

add the patches to the internal cache, and then restart the failed V2V job.

The V2V process will continue and will not need to re-do the disk conversi

on.

If you use New-V2V to convert a VMware-based virtual machine running any ot

her operating system to a Hyper-V or Virtual Server-based virtual machine,

the virtual machine might not start up or might not function correctly. To

ensure a successful conversion, you must first modify the guest operating s

ystem to one of the listed supported operating systems.

FOR MORE INFORMATION

--------------------

\* About how Virtual Machine Manager can convert VMDK files directly, type:

Get-Help Copy-VMDK

\* About how to add required files to the internal cache, type:

Get-Help Add-Patch

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-VMXMachineConfig <VMXMachineConfig>

Specifies a VMX machine configuration for a VMware-based virtual machin

e. VMX machine configuration includes information about the virtual mac

hine's hardware, disks, and operating system.

Note: In VMM 2007, this parameter, when used with the New-V2V and Remov

e-VMXMachineConfig cmdlets, was named MachineConfig.

-VMXPath <String>

Specifies the full UNC path to the .vmx file of a VMware virtual machin

e.

Example format: \\ServerName\VolumeName\DirectoryName\VMwareVM.vmx

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-DelayStart <Int32>

Specifies the number of seconds to wait after the virtualization servic

e starts before automatically starting a virtual machine. Used to stagg

er the startup time of multiple virtual machines to help reduce the dem

and on the physical computer’s resources. A typical setting might be 30

to 60 seconds.

TYPE OF HOST MAXIMUM CONFIGURABLE DELAY

------------ --------------------------------

Hyper-V 1000000000 seconds (277777 hours)

Virtual Server 86400 seconds (24 hours)

VMware ESX 65535 seconds (18 hours)

-Description <String>

Specifies a description for the specified object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-Name <String>

Specifies the name of a VMM object.

-NetworkLocation <String>

Specifies the network location for a physical network adapter or for a

virtual network adapter, or changes the default network location of a h

ost's physical network adapter.

Example formats:

-NetworkLocation $NetLoc ($NetLoc might contain "Corp.Contoso.com")

-OverrideNetworkLocation $TRUE –NetworkLocation "HostNICNewLocation.Co

ntoso.com"

-NetworkTag <String>

Specifies a word or phrase to associate with a virtual network adapter

that is configured to connect to a specific internal or external networ

k on the host. The NetworkTag identifies all VMs with the same NetworkT

ag as members of the same network. VMM uses a NeworkTag (if one exists)

when it evaluates hosts as possible candidates on which to deploy a VM

. If the host does not include VMs on the network with the same Network

Tag as the VM to be placed, the host receives zero stars in the placeme

nt process.

-NoConnection

Disconnects a virtual network adapter from a virtual network.

-OverridePatchPath <String>

For internal use only (not for use in your code).

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PhysicalAddress <String>

Specifies the physical address (MAC address) of a physical or virtual n

etwork adapter.

Note: In VMM 2007, this parameter was named EthernetAddress.

-PhysicalAddressType <String>

Specifies the type of physical address (MAC address) to use for a virtu

al network adapter:

Valid values: Static, Dynamic

Note: In VMM 2007, this parameter was named EthernetAddressType.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsSystem

Specifies that a virtual machine on a Virtual Server host will run unde

r the local system account. If specified, Virtual Server will not autom

atically start the virtual machine when the Virtual Server service star

ts. (This parameter does not apply to virtual machines on Hyper-V or VM

ware ESX hosts because Hyper-V and VMware run a virtual machine under t

he local system account by default; you cannot change this setting on t

hose virtualization platforms.)

-RunAsUserCredential <PSCredential>

Specifies the guest account (domain\account) that a virtual machine on

a Virtual Server host runs under. If specified, Virtual Server will aut

omatically start a virtual machine when the Virtual Server service star

ts. For enhanced security, create a special account with limited permis

sions:

FILE TYPE MINIMUM REQUIRED PERMISSIONS FOR GUEST ACCOUNT

----------- ----------------------------------------------

.vmc file Read Data, Write Data, Execute File

.vmc folder List Folder, Write/Create File (required to save VM state)

.vhd file Read Data, Read Attributes, Read Extended Attributes,

Write Data

.vnc file Execute File, Read Data, Read Attributes, Read

(required if VM connects to a virtual network)

Note: This parameter does not apply to virtual machines on Hyper-V or V

Mware ESX hosts.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SkipInstallVirtualizationGuestServices

Skips the installation of virtualization guest services on a Windows-ba

sed virtual machine. By default, this parameter is set to FALSE and

VMM installs the appropriate virtualization guest service automatically

. For a virtual machine on a Hyper-V host, the virtualization guest ser

vice is called Integration Components (VMGuest.iso). For a virtual mach

ine on a Virtual Server host, the virtualization guest service is calle

d Virtual Machine Additions (VMAdditions.iso). Virtual machines on a VM

ware ESX host do not use a virtualization guest service.

-SourceNetworkConnectionID <String>

Specifies the MAC address or network name of the physical network adapt

er to be converted into a virtual network adapter in the virtual machin

e.

-StartAction <String>

Specifies the behavior of a virtual machine when the virtualization ser

vice (Hyper-V, Virtual Server, or VMware) starts. To specify that a vir

tual machine deployed on a Virtual Server host starts automatically, us

e the -RunAsUserCredential parameter to specify an account with appropr

iate permissions (otherwise, the StartAction reverts to NeverAutoTurnOn

VM).

Valid values: AlwaysAutoTurnOnVM, NeverAutoTurnOnVM, TurnOnVMIfRunningW

henVSStopped

-StartVM

Specifies that the virtual machine starts when it arrives at the destin

ation host.

-StopAction <String>

Specifies the behavior of the virtual machine when the virtualization s

ervice (Hyper-V, Virtual Server, or VMware) stops.

Valid values: SaveVM, TurnOffVM, ShutdownGuestOS

-Trigger

Starts the execution a job group for a physical-to-virtual (P2V) conver

sion, a virtual-to-virtual (V2V) conversion, or the conversion of a phy

sical hard disk to a virtual hard disk.

-UseHardwareAssistedVirtualization

Specifies that, for a virtual machine deployed on a Virtual Server host

, hardware-assisted virtualization is used if it is available (when set

to TRUE). The Virtual Server host must support AMD Virtualization (AMD

-V) or Intel Virtualization Technology (Intel-VT) hardware virtualizati

on. This parameter does not apply to virtual machines on Hyper-V hosts

or VMware ESX hosts.

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-VirtualNetworkAdapter <VirtualNetworkAdapter>

Specifies a virtual network adapter object for a virtual machine.

TYPE OF HOST NUMBER OF VIRTUAL NETWORK ADAPTERS

------------ ----------------------------------

Virtual Server Up to 4 emulated adapters per VM.

Hyper-V Up to 4 emulated adapters per VM.

Up to 8 synthetic adapters per VM.

(Exception: no driver available for an emulated

network adapter on a Windows Server 2003 x64 guest.)

VMware ESX Up to 4 emulated adapters per VM.

-VLANEnabled

Enables a virtual LAN (a VLAN) for use by virtual machines on a Hyper-V

or VMware ESX host.

Example format for a single VLAN:

-VLANEnabled -VLANMode "Access" -VLANID 35

Example format for multiple VLANs:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

-VLANID <Int32>

Assigns a numerical identifier in the range 1-4094 to a virtual network

adapter on a virtual machine or to a physical network adapter on a Hyp

er-V or VMware ESX host.

Configure a VLANID on a Hyper-V or VMware ESX host:

\* On an externally bound physical network adapter.

Configure a VLANID on a virtual network adapter of a virtual machine:

\* Bound to a physical network adapter on the host, or

\* Bound to an internal virtual network on the host.

Example format:

-VLANEnabled

-VLANMode "Access" -VLANID 35

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Convert a VMware-based VM deployed on an ESX host to a VM deployed on a

Windows-based host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $ESXHost = Get-VMHost -ComputerName "ESXHost01"

PS C:\> $VMHost = Get-VMHost -ComputerName "HyperVHost01.Contoso.com"

PS C:\> $VM = Get-VM -VMHost $ESXHost –Name "SourceVM"

PS C:\> New-V2V -VM $VM -VMHost $VMHost -Name "DestinationVM" -Path "C:\MyV

Ms" -MemoryMB 256 –RunAsynchronously

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database; the following commands use t

his server by default.

The second command gets the object that represents the host named ESXHost01

from the VMM database and stores the host object in variable $ESXHost.

The third command gets the object that represents the host named HyperVHost

01 (in the Contoso.com domain) and stores the host object in variable $VMHo

st.

The fourth command gets the object that represents the VM named SourceVM on

ESXHost01 and stores the VM object in variable $VM.

In the last command, New-V2V performs the following operations:

\* Creates a Windows-based virtual machine named DestinationVM from

the source VMware virtual machine named SourceVM. The command

deploys the new virtual machine, now named DestinationVM, onto

HyperVHost01, storing the virtual machine files in the folder C:\MyVMs

on HyperVHost01.

\* Assigns 256 MB of memory on HyperVHost01 for use by the new

virtual machine.

\* Uses the -RunAsynchronously parameter to return control to the

command shell immediately (before the command completes).

All of the virtual disks on the source virtual machine will be converted an

d attached to the new virtual machine.

2: Convert a VMware-based VM stored in the VMM library to a VM deployed on

a Windows-based host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $LibServ = Get-LibraryServer -ComputerName "FileServer02.Contoso.co

m"

PS C:\> $VMHost = Get-VMHost -ComputerName "VirtualServerHost02.Contoso.com

"

PS C:\> New-V2V -LibraryServer $LibServ -VMXPath "\\FileServer02\MSSCVMMLib

rary\VMware\VMSource.vmx" -VMHost $VMHost -Name "VM02" -Path "C:\MyVMs" -Me

moryMB 256 -RunAsynchronously

The first command connects to VMMServer1.

The second command gets the object that represents the library server named

FileServer02 and stores the library server object in $LibServ.

The third command gets the object that represents the host named VirtualSer

verHost02 and stores the host object in $VMHost.

In the last command, New-V2V performs the following operations:

\* Creates a Windows-based virtual machine named VM02 from

the source VMware file (VMSource.vmx) stored at the

specified path on FileServer02, and then deploys the new

virtual machine (VM02) onto VirtualServerHost02. The command

stores the virtual machine files in the folder C:\MyVMs on

VirtualServerHost02.

\* Assigns 256 MB of memory on VirtualServerHost02 for use by the new

virtual machine.

\* Uses the -RunAsynchronously parameter to return control to the

command shell immediately (before the command completes).

REMARKS

For more information, type: "get-help New-V2V -detailed".

For technical information, type: "get-help New-V2V -full".

## VirtualCOMPort

### Get-VirtualCOMPort

SYNOPSIS

Gets Virtual Machine Manager virtual communication (COM) port objects from

a virtual machine, template, or hardware profile.

SYNTAX

Get-VirtualCOMPort -VM [<String VM>] [<CommonParameters>]

Get-VirtualCOMPort -HardwareProfile <HardwareProfile> [<CommonParameters>]

Get-VirtualCOMPort -All [-VMMServer [<String ServerConnection>]] [<CommonPa

rameters>]

Get-VirtualCOMPort -Template [<Template String>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or both objects that represent Virtual Machine Manager virtual COM

ports from a virtual machine object, from a template object, or from a har

dware profile object.

A virtual COM port can connect to a physical port on a virtual machine host

server, to a text file, or to a named pipe. Each virtual machine, template

, and hardware profile contains exactly two COM ports.

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get COM ports from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> Get-VirtualCOMPort -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database; the following commands use t

his server by default.

The second command gets from the VMM database the object that represents th

e virtual machine named VM01 and stores the virtual machine object in varia

ble $VM.

The last command retrieves both objects that represent virtual COM ports on

VM01 and displays information about these virtual COM ports to the user.

2: Get COM ports from a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> Get-VirtualCOMPort -Template $Template

The first command connects to VMMServer1.

The second command gets the object that represents the template named Templ

ate1 and stores the template object in $Template.

The last command gets both objects that represent virtual COM ports on Temp

late1 and displays information about these virtual COM ports to the user.

3: Get COM ports from a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> Get-VirtualCOMPort -HardwareProfile $HWProfile

The first command connects to VMMServer1.

The second command gets the object that represents the hardware profile nam

ed HardwareProfile and stores the hardware profile object in $HWProfile.

The last command gets both objects that represent virtual COM ports on Hard

wareProfile1 and displays information about these virtual COM ports to the

user.

REMARKS

For more information, type: "get-help Get-VirtualCOMPort -detailed".

For technical information, type: "get-help Get-VirtualCOMPort -full".

### Set-VirtualCOMPort

SYNOPSIS

Changes properties of a virtual COM port associated with a virtual machine,

template, or hardware profile used in Virtual Machine Manager.

SYNTAX

Set-VirtualCOMPort [-VirtualCOMPort] <VirtualCOMPort> -NoAttach [-JobGroup

<Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<C

ommonParameters>]

Set-VirtualCOMPort -GuestPort <Byte> -NamedPipe <String> [-JobGroup <Guid>]

[-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServe

r [<String ServerConnection>]] [<CommonParameters>]

Set-VirtualCOMPort -GuestPort <Byte> -VMHostCOMPort <Int32> [-JobGroup <Gui

d>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMSe

rver [<String ServerConnection>]] [-WaitForModem <Boolean>] [<CommonParamet

ers>]

Set-VirtualCOMPort [-VirtualCOMPort] <VirtualCOMPort> -VMHostCOMPort <Int32

> [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynch

ronously] [-WaitForModem <Boolean>] [<CommonParameters>]

Set-VirtualCOMPort [-VirtualCOMPort] <VirtualCOMPort> -TextFile <String> [-

JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchrono

usly] [<CommonParameters>]

Set-VirtualCOMPort -GuestPort <Byte> -NoAttach [-JobGroup <Guid>] [-JobVari

able <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String

ServerConnection>]] [<CommonParameters>]

Set-VirtualCOMPort [-VirtualCOMPort] <VirtualCOMPort> -NamedPipe <String> [

-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchron

ously] [<CommonParameters>]

Set-VirtualCOMPort -GuestPort <Byte> -TextFile <String> [-JobGroup <Guid>]

[-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer

[<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual communications (COM) port assoc

iated with a virtual machine, template, or hardware profile used in a Virtu

al Machine Manager environment.

CONNECTING A VIRTUAL COM PORT

-----------------------------

Depending on the type of host on which a virtual machine is (or will be) de

ployed, you can use this cmdlet to connect a virtual COM port to a physical

COM port on a host server, to a text file, or to a named pipe, or you can

use it to disconnect a virtual COM port. Connecting a virtual COM port on a

virtual machine to a physical COM port on its host lets the virtual machin

e use the physical COM port for input and output.

Type of Host Available Virtual COM Port Connection Types

------------ -------------------------------------------

Hyper-V Connects to a named pipe only

Virtual Server Connects to a physical COM port, text file, or named pipe

VMware ESX Connects to a physical COM port, text file, or named pipe

HOW THE WAITFORMODEM PARAMETER WORKS

------------------------------------

You can use the Set-VirtualCOMPort cmdlet with the WaitForModem parameter t

o specify whether a virtual COM port on a virtual machine will connect imme

diately to a physical COM port on the host when the virtual machine starts,

or not.

If WaitForModem is set to TRUE, the virtual machine attempts to capture (co

nnect to) the physical COM port on the host only when a program running on

the virtual machine sends a modem command to the physical COM port. If the

COM port on the host is already connected, the virtual machine cannot conne

ct to it. If the virtual machine successfully connects to the physical COM

port, the virtual machine will later release the physical COM port back to

the host operating system if the program on the virtual machine that uses t

he COM port stops using the COM port.

If WaitForModem is set to FALSE, the virtual machine attempts to capture th

e physical COM port on the host as soon as the virtual machine starts. If t

he COM port on the host is already captured, the virtual machine cannot con

nect to it (same behavior for TRUE or FALSE). If the virtual machine succes

sfully connects to the physical COM port, the virtual machine will not rele

ase the physical COM port back to the host operating system until the virtu

al machine is shut down (behavior for FALSE differs from behavior for TRUE)

.

PARAMETER SETS THAT USE VIRTUALCOMPORT VERSUS GUEST PORT

--------------------------------------------------------

In VMM 2008, the Set-VirtalCOMPort cmdlet uses the VirtualCOMPort parameter

and the GuestPort parameter as follows:

\* -VirtualCOMPort<VirtualCOMPort>

Used with four parameter sets of Set-VirtualCOMPort to specify

a VirtualCOMPort object.

\* -GuestPort<Byte>

Used with four other parameters sets of Set-VirtualCOMPort to specify

a virtual COM port by ID (0 or 1).

Review the SYNTAX information for Set-VirtualCOMPort to see which parameter

sets use the VirtualCOMPort parameter and which use the GuestPort paramete

r.

PARAMETERS

-VirtualCOMPort <VirtualCOMPort>

Specifies a virtual COM port object. VMM supports configuring two COM p

orts on a virtual machine, template, or hardware profile.

Note: In VMM 2007, the VirtualCOMPort parameter was named GuestPort. In

VMM 2008, the GuestPort parameter is used to specify a virtual COM por

t by a numerical identifier (0 or 1).

-GuestPort <Byte>

Specifies a virtual COM port on a virtual machine by a numerical identi

fier.

Valid values: 0 or 1.

-NamedPipe <String>

Specifies a named pipe to which to connect a virtual COM port. Typical

uses include creating a connection between a virtual machine and a debu

gging program on the host (if the debugger supports the use of named pi

pes), or creating a virtual null modem cable between two virtual machin

es.

Example named pipe path: \\.\Contoso\Pipe\PipeName

-NoAttach

Specifies that no physical COM port on a host, named pipe, or file will

be connected to a virtual COM port; or disconnects a virtual COM port

that is already connected to a physical COM port, to a named pipe, or t

o a text file.

-TextFile <String>

Specifies a text file on the host to which to connect a virtual COM por

t on a virtual machine so that output from the virtual COM port can be

sent to that text file. The text file can be on any valid disk drive on

the host.

Example format: -TextFile "D:\ComPort.txt"

-VMHostCOMPort <Int32>

Specifies a physical COM port object on a host server to which you can

connect a virtual COM port.

Note: In VMM 2007, this parameter was named HostPort.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-WaitForModem <Boolean>

Specifies that a virtual COM port will wait to connect to a physical CO

M port on the host (when set to TRUE), or that the virtual COM port wil

l connect immediately to a physical COM port on the host as soon as the

virtual machine starts (when set to FALSE). For more information, type

: Get-Help Set-VirtualCOMPort.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Connect a virtual COM port to a physical COM port.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Set-VirtualCOMPort -VirtualCOMPort $VM.VirtualCOMPorts[0] -VMHostCO

MPort 1 -WaitForModem $FALSE

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The last command connects the first virtual COM port on VM01 ($VM.VirtualCO

MPorts[0]) to the specified physical COM port (COM1) on the host. Setting t

he WaitForModem parameter to $FALSE connects the host COM port as soon as t

he virtual machine starts.

2: Connect a virtual COM port to a named pipe.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $COM1 = Get-VirtualCOMPort -VM $VM | where {$\_.Name -eq "COM1"}

PS C:\> Set-VirtualCOMPort -VirtualCOMPort $COM1 -NamedPipe "\\Contoso\Pipe

\PipeName"

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third commands gets the virtual COM port named COM1 from VM02 and store

s the virtual COM port object in $COM1.

The last command connects the virtual COM port to the named pipe \\Contoso\

Pipe\PipeName.

3: Connect a virtual COM port to a text file.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> Set-VirtualCOMPort -VirtualCOMPort $VM.VirtualCOMPorts[0] -TextFile

"D:\ComPort.txt

The first command connects to VMMServer1.

The second command gets the object that represents VM03 and stores it in $V

M.

The last command connects the first virtual COM port on VM03 ($VM.VirtualCO

MPorts[0]) to the text file D:\ComPort.txt on the host.

4: Disconnect a virtual COM port.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM04"

PS C:\> $COM1 = Get-VirtualCOMPort -VM $VM | where {$\_.Name -eq "COM1"}

PS C:\> Set-VirtualCOMPort -VirtualCOMPort $COM1 -NoAttach

The first command connects to VMMServer1.

The second command gets the object that represents VM04 and stores it in $V

M.

The third command gets the virtual COM port on VM04 named COM1 and stores t

he virtual COM port object in $COM1.

The last command disconnects the virtual COM port object in $ComPort by spe

cifying the NoAttach parameter.

NOTE: You can use this command to disconnect a virtual COM port that is cur

rently connected to a physical COM port on a host, to a named pipe, or to a

text file.

5: Specify settings for each virtual COM port on a new VM, configure other

settings, and store the new VM in the VMM library.

PS C:\> $JobGroupId = [Guid]::NewGuid().ToString()

PS C:\> Set-VirtualCOMPort -TextFile "D:\text" -VMMServer VMMServer1.Contos

o.com -GuestPort 1 -JobGroup $JobGroupId

PS C:\> Set-VirtualCOMPort -NamedPipe "\\.\pipe\name" -GuestPort 2 -JobGrou

p $JobGroupId

PS C:\> $CPUType = Get-CPUType | where {$\_.Name -eq "1.20 GHz Athlon MP"}

PS C:\> New-HardwareProfile -Owner "Contoso\User1" -CPUType $CPUType -Name

"Profile1" -Description "" -CPUCount 1 -MemoryMB 512 -ExpectedCPUUtilizatio

n 20 -DiskIO 0 -NetworkUtilization 10 -RelativeWeight 100 -HighlyAvailable

$FALSE -NumLock $FALSE -BootOrder CD,IdeHardDrive,PxeBoot,Floppy -LimitCPUF

unctionality $FALSE -JobGroup $JobGroupId

PS C:\> $VHD = Get-VirtualHardDisk | where {$\_.Location -eq "\\FileServer1.

Contoso.com\MSSCVMMLibrary\VHDs\Large.vhd"} | where {$\_.HostName -eq "FileS

erver1.Contoso.com"}

PS C:\> New-VirtualDiskDrive -IDE -Bus 0 -LUN 0 -JobGroup $JobGroupId -Virt

ualHardDisk $VHD -Filename "VM01\_Large.vhd"

PS C:\> $LibraryServer = Get-LibraryServer | where {$\_.Name -eq "FileServer

1.Contoso.com"}

PS C:\> $HardwareProfile = Get-HardwareProfile | where {$\_.Name -eq "Profil

e1"}

PS C:\> New-VM -Name "VM05" -Description "" -Owner "Contoso\User1" -Library

Server $LibraryServer -SharePath "\\FileServer1.Contoso.com\VMs" -HardwareP

rofile $HardwareProfile -JobGroup $JobGroupId -RunAsynchronously -RunAsSyst

em -StartAction NeverAutoTurnOnVM -UseHardwareAssistedVirtualization $FALSE

-StopAction SaveVM

The first command generates a globally unique identifier (GUID) and stores

the GUID string in $JobGroupID. The job group ID functions as an identifier

that groups subsequent commands that include this identifier into a single

job group.

The second command connects to VMMServer1 and specifies that the first virt

ual COM port (for the virtual machine to be created in the last command) wi

ll use a text file. Using $JobGroupID specifies that the virtual COM port i

s not created until just before the New-VM cmdlet runs in the last command.

The third command specifies that the second virtual COM port (for the virtu

al machine to be created in the last command) will use a named pipe. The co

mmmand uses $JobGroupID to delay the execution of this cmdlet until just be

fore the New-VM cmdlet runs in the last command.

The fourth command gets a specific CPU type by name and stores the CPU type

object in $CPUType.

The fifth command will create a new hardware profile but uses $JobGroupID t

o specify that the hardware profile is not created until just before the Ne

w-VM cmdlet (in the last command) runs. The New-HardwareProfile cmdlet name

s the hardware profile "Profile1" and specifies an owner; sets the specifie

d values for CPU, memory,disk, and network settings; specifies that a virtu

al machine created from this hardware profile is not highly available (that

is, will not be placed on a host in a host cluster); disables NumLock and

sets limited CPU functionality to FALSE (to indicate that the operating sys

tem on a virtual machine created from this hardware profile is not an older

operating system that requires limited CPU functionality); and specifies t

he BIOS boot order for a virtual machine created from this hardware profile

.

The sixth command gets an existing VHD called Large.vhd from the library an

d stores the VHD object in $VHD.

The seventh command will create a new virtual disk drive and will attach th

e virtual hard disk object retrieved in the preceding command to the first

slot of the first IDE channel on the new virtual disk drive but uses $JobGr

oupID to delay the execution of this cmdlet until just before the New-VM cm

dlet runs in the last command.

The eighth command gets the object that represents the library server (File

Server1) on which the VM will be stored and stores the library server objec

t in $LibraryServer.

The ninth command gets the hardware profile object named Profile1 created i

n an earlier step and stores the hardware profile object in $HardwareProfil

e.

The last command uses the New-VM cmdlet, and the commands grouped into a se

t by $JobGroupID, to create a new VM named VM05. The command specifies an o

wner for the virtual machine and a share in the library where the new VM wi

ll be stored; specifies that the command will run asynchronously, that the

virtual machine will run under the local system account (which indicates th

at the virtual machine will be deployed on a Virtual Server host), and that

hardware-assisted virtualization is not used (which indicates that the vir

tual machine will be deployed on a Virtual Server that does not support eit

her AMD-V) or Intel-VT hardware virtualization); and specifies start and st

op actions for the virtual machine.

REMARKS

For more information, type: "get-help Set-VirtualCOMPort -detailed".

For technical information, type: "get-help Set-VirtualCOMPort -full".

## VirtualDiskDrive

### Compress-VirtualDiskDrive

SYNOPSIS

Compresses a dynamically expanding virtual hard disk attached to a virtual

disk drive object on a stopped virtual machine on a Windows-based host mana

ged by Virtual Machine Manager to reduce the size of the virtual hard disk.

NOTE: In VMM 2007, this cmdlet was named Compact-VirtualHardDisk.

SYNTAX

Compress-VirtualDiskDrive [-VirtualDiskDrive] <VirtualDiskDrive> [-JobGroup

<Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<

CommonParameters>]

DETAILED DESCRIPTION

Compresses a dynamically expanding virtual hard disk attached to a virtual

disk drive object on a virtual machine on a Windows-based host managed by V

irtual Machine Manager to reduce the size of the virtual hard disk. The vir

tual machine must be stopped before you can compress the virtual hard disk.

You can use the Compress-VirtualDiskDrive cmdlet to compress only a Windows

-based virtual hard disk file (a .vhd file) attached to a virtual disk driv

e object on a virtual machine that is deployed on a Hyper-V host or on a Vi

rtual Server host.

A VMware-based virtual hard disk file (a .vmdk file) on a virtual machine d

eployed on an ESX Server 3.0 or 3.5 host is fixed (not dynamic), and you ca

nnot compress a fixed virtual hard disk.

PARAMETERS

-VirtualDiskDrive <VirtualDiskDrive>

Specifies a virtual disk drive object. You can attach either a virtual

hard disk (for a virtual machine on any host) or a pass-through disk (f

or a virtual machine on a Hyper-V host or an ESX host) to a virtual dis

k drive object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Compress a virtual hard disk attached to a virtual disk drive on a VM de

ployed on a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VDD = Get-VirtualDiskDrive -VM (Get-VM -Name "VM01")

PS C:\> Compress-VirtualDiskDrive -VirtualDiskDrive $VDD

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual disk drive a

ttached to a virtual machine named VM01 from the VMM database and stores th

e virtual disk drive object in variable $VDD.

NOTE: This example assumes the virtual machine has only one virtual disk dr

ive and that the virtual hard disk attached to the virtual disk drive is, c

urrently, a dynamic virtual hard disk.

The third command compresses the dynamically expanding virtual hard disk th

at is attached to the virtual disk drive on VM01.

NOTE: You can use the Compress-VirtualDiskDrive cmdlet to compress only a W

indows-based virtual hard disk file (a .vhd file) attached to a virtual dis

k drive on a virtual machine deployed on a Hyper-V host or on a Virtual Ser

ver host. A VMware-based virtual hard disk file (a .vmdk file) on a virtual

machine deployed on an ESX Server 3.0 or 3.5 host is fixed (not dynamic),

and you cannot compress a fixed virtual hard disk.

REMARKS

For more information, type: "get-help Compress-VirtualDiskDrive -detailed".

For technical information, type: "get-help Compress-VirtualDiskDrive -full"

.

### Convert-VirtualDiskDrive

SYNOPSIS

Converts an existing virtual hard disk attached to a virtual disk drive obj

ect from dynamic to fixed or from fixed to dynamic; or converts a pass-thro

ugh disk attached to a virtual disk drive object to a virtual hard disk.

SYNTAX

Convert-VirtualDiskDrive [-VirtualDiskDrive] <VirtualDiskDrive> -Dynamic [-

FileName <String>] [-JobGroup <Guid>] [-JobVariable <String>] [-Path <Strin

g>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Convert-VirtualDiskDrive [-VirtualDiskDrive] <VirtualDiskDrive> -Fixed [-Fi

leName <String>] [-JobGroup <Guid>] [-JobVariable <String>] [-Path <String>

] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Converts an existing virtual hard disk attached to a virtual disk drive obj

ect from dynamic to fixed or from fixed to dynamic; or converts a pass-thro

ugh disk attached to a virtual disk drive object to a virtual hard disk.

CONVERTING A VIRTUAL HARD DISK FROM DYNAMIC TO FIXED, OR VICE VERSA

----------------------------------------------------------------

You can use the Convert-VirtualDiskDrive cmdlet to convert a virtual hard d

isk, which is attached to a virtual disk drive object on a virtual machine,

from a fixed disk format to a dynamic disk format, or vice versa. The virt

ual machine on which the virtual hard disk is configured must be stopped be

fore you can convert the virtual hard disk from one format to the other.

You can convert the disk format of a Windows-based virtual hard disk file (

a .vhd file) on a virtual machine deployed on a Hyper-V host or on a Virtua

l Server host.

A VMware-based virtual hard disk file (a .vmdk file) on a virtual machine t

hat is deployed on an ESX Server 3.0 or 3.5 host is fixed in format and can

not be converted to a dynamic format.

CONVERTING A PASS-THROUGH DISK TO A VIRTUAL HARD DISK

--------------------------------------------------

You can use the Convert-VirtualDiskDrive cmdlet to convert a pass-through d

isk attached to a virtual disk drive on a virtual machine on a Hyper-V host

to a virtual hard disk. The virtual machine must be stopped before you can

convert the pass-through disk to a virtual hard disk.

A pass-through disk is a physical hard disk on the host that a virtual mach

ine can use instead of using a virtual hard disk.

PARAMETERS

-VirtualDiskDrive <VirtualDiskDrive>

Specifies a virtual disk drive object. You can attach either a virtual

hard disk (for a virtual machine on any host) or a pass-through disk (f

or a virtual machine on a Hyper-V host or an ESX host) to a virtual dis

k drive object.

-Dynamic

Specifies that a virtual hard disk can expand dynamically.

-Fixed

Specifies that a virtual hard disk is fixed in size.

-FileName <String>

Specifies the file name to use when you rename a virtual hard disk file

as you add it to a virtual machine.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Convert a pass-through disk on a virtual disk drive on a VM to a virtual

hard disk.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $VDD = Get-VirtualDiskDrive -VM $VM

PS C:\> Convert-VirtualDiskDrive $VDD -Fixed -Path "C:\"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM. This example assumes that VM01 is currently configured to use a pa

ss-through disk and that the virtual machine has only one passthrough disk.

The third command gets the virtual disk drive object on VM01 and stores thi

s object in variable $VDD.

The last command converts the pass-through disk drive represented in $VDD t

o a fixed virtual hard disk and moves the virtual hard disk to the destinat

ion folder "C:\".

NOTE: You can convert a pass-through disk attached to a virtual disk drive

on a virtual machine on a Hyper-V host to a virtual hard disk. This functio

nality does not apply to Virtual Server or Hyper-V hosts.

2: Convert one of several pass-through disks on a virtual disk drive on a V

M to a VHD.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $VDD = Get-VirtualDiskDrive -VM $VM

PS C:\> $VDD[2] | Convert-VirtualDiskDrive -Dynamic -Path "D:\"

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM. This example assumes

that VM02 has three virtual disk drive objects and that the first virtual d

isk drive is bound to a virtual hard drive whereas both the second and thir

d virtual disk drives are bound to pass-through disks.

The third command gets all virtual disk drive objects on VM02 and stores th

em in $VDD (an object array).

The last command converts the third pass-through disk ($VDD[2]) to a dynami

cally expanding virtual hard disk and moves this new virtual hard disk to t

he destination folder "D:\".

3: Convert a dynamic VHD attached to a virtual disk drive object on a VM to

a fixed format.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VDD = Get-VirtualDiskDrive -VM (Get-VM -Name "VM01")

PS C:\> Convert-VirtualDiskDrive -VirtualDiskDrive $VDD -Fixed

The first command connects to VMMServer1.

The second command gets the object that represents the only virtual disk dr

ive that is attached to a virtual machine named VM01 and stores the virtual

disk drive object in $VDD.

NOTE: This example assumes that the virtual machine has only one virtual di

sk drive object and that the virtual hard disk attached to the virtual disk

drive is, currently, a dynamic virtual hard disk.

The last command converts the virtual hard disk attached to the virtual dis

k drive object stored in $VDD to a fixed disk.

NOTE: You can use the Convert-VirtualDiskDrive cmdlet to convert only a Win

dows-based virtual hard disk file (a .vhd file) on a virtual machine deploy

ed on a Hyper-V host or on a Virtual Server host. A VMware-based virtual ha

rd disk file (a .vmdk file) on a virtual machine deployed on an ESX Server

3.0 or 3.5 host is fixed (not dynamic) and cannot be converted to dynamic.

REMARKS

For more information, type: "get-help Convert-VirtualDiskDrive -detailed".

For technical information, type: "get-help Convert-VirtualDiskDrive -full".

### Expand-VirtualDiskDrive

SYNOPSIS

Expands a virtual hard disk attached to a virtual disk drive object on a st

opped virtual machine deployed on a host managed by Virtual Machine Manager

.

SYNTAX

Expand-VirtualDiskDrive [-VirtualDiskDrive] <VirtualDiskDrive> -Size <Int32

> [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynch

ronously] [<CommonParameters>]

DETAILED DESCRIPTION

Expands a virtual hard disk attached to a virtual disk drive object on a vi

rtual machine deployed on a host managed by Virtual Machine Manager in orde

r to increase the total capacity of the virtual hard disk. The virtual mach

ine must be stopped before you can expand the virtual hard disk.

You can use the Expand-VirtualDiskDrive cmdlet to expand a Windows-based vi

rtual hard disk file (a .vhd file) attached to a virtual disk drive object

on a virtual machine deployed on a Hyper-V host. You can also use this cmdl

et to expand a VMware-based virtual hard disk file (a .vmdk file) on a virt

ual machine deployed on an ESX Server 3.0 or 3.5 host. You cannot use this

cmdlet to expand a virtual hard disk on a virtual machine deployed on a Vir

tual Server host.

PARAMETERS

-VirtualDiskDrive <VirtualDiskDrive>

Specifies a virtual disk drive object. You can attach either a virtual

hard disk (for a virtual machine on any host) or a pass-through disk (f

or a virtual machine on a Hyper-V host or an ESX host) to a virtual dis

k drive object.

-Size <Int32>

Specifies, in megabytes (MB), the size of a fixed virtual hard disk fil

e or the maximum possible size of a dynamically expanding virtual hard

disk file.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1. Expand a virtual hard disk attached to a virtual disk drive on a virtual

machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $VDD = Get-VirtualDiskDrive -VM $VM | where {$\_.Bus -eq 0 -and $\_.L

un -eq 0}

PS C:\> Expand-VirtualDiskDrive -VirtualDiskDrive $VDD -Size 45

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VM01 and stores the virt

ual machine object in variable $VM.

The third command gets the virtual disk drive located on the first controll

er ($\_.Bus -eq 0) and first slot of that controller ($\_.Lun -eq 0) of VM01,

and it stores the virtual disk drive object in variable $VDD.

The last command expands the size of the virtual hard disk attached to the

virtual disk drive to 45 GB.

REMARKS

For more information, type: "get-help Expand-VirtualDiskDrive -detailed".

For technical information, type: "get-help Expand-VirtualDiskDrive -full".

### Get-VirtualDiskDrive

SYNOPSIS

Gets virtual disk drive objects on templates or on virtual machines managed

by Virtual Machine Manager.

SYNTAX

Get-VirtualDiskDrive -Template [<Template String>] [<CommonParameters>]

Get-VirtualDiskDrive -VM [<String VM>] [<CommonParameters>]

Get-VirtualDiskDrive -All [-VMMServer [<String ServerConnection>]] [<Common

Parameters>]

DETAILED DESCRIPTION

Gets one or more Virtual Machine Manager virtual disk drive objects configu

red on templates stored in the library or configured on virtual machines (e

ither deployed on a host or stored in the library).

For more information about virtual disk drive objects, type:

Get-Help New-VirtualDiskDrive -detailed

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get a list of all available virtual disk drives in your VMM environment.

PS C:\> Get-VirtualDiskDrive -VMMServer VMMServer1.Contoso.com -All

Gets a list of all virtual disk drives bound to all virtual machines regist

ered to VMM from the VMM database on VMMServer1 . The command displays info

rmation about each virtual disk drive to the user.

2: Get virtual disk drives for a specific virtual machine.

PS C:\> $VM = Get-VM -VMMServer VMMServer1.Contoso.com | where {$\_.Name -eq

"VM02"}

PS C:\> $VDD = Get-VirtualDiskDrive -VM $VM

PS C:\> $VDD

The first command gets the object that represents the virtual machine named

VM02 and stores the virtual machine object in $VM.

The second command gets all objects that represent virtual disk drives on V

M02 and stores the virtual disk drive objects in $VDD. If, as this example

assumes, a virtual machine contains multiple virtual disk drives, each virt

ual disk drive has connected to it either a virtual hard disk or a pass-thr

ough disk.

The last command displays the properties of each virtual disk drive on VM02

to the user, including the name of any virtual hard disks and the path to

the physical drive on the host for any pass-through disks.

3. Count all virtual disk drives (except pass-through disks) on the second

slot for both IDE channels.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VDD = @(Get-VirtualDiskDrive -ALL | where {$\_.BusType -eq 'IDE' -a

nd $\_.PassThroughDisk -eq $null -and $\_.LUN -eq 1 -and ($\_.Bus -eq 0 -or $\_

.Bus -eq 1)})

PS C:\> $VDD.Count

The first command connects to VMMServer1.

The second command gets the objects that represent all virtual disk drives

(excluding pass-through disks) that are connected to the second slot of eit

her IDE channel. Using the '@' symbol and parentheses ensures that the comm

and stores the results in an array (in case the command returns a single ob

ject or a null value).

The last command counts the number of virtual disk drive objects that match

the filter criteria and displays the results to the user.

4: Get virtual disk drives for a specific template.

PS C:\> Get-VMMServer "VMMServer1.Contoso.com"

PS C:\> $Template = @(Get-Template)

PS C:\> $Template | ForEach-Object {Get-VirtualDiskDrive -Template $\_ | whe

re {$\_.BusType -eq "IDE"}} | Format-List Name,BusType,Bus,LUN

The first command connects to VMMServer1.

The second command gets all objects that represent templates and stores the

template objects in $Template. Using the "@" symbol and parentheses ensure

s that the command stores the results in an array (in case the command retu

rns a single object or a null value).

The last command performs the following operations:

\* Passes each template object in $Template to the ForEach-Object cmdlet.

\* Gets all virtual disk drive objects for each template in $Template.

\* Passes all virtual disk drive objects to the "where" filter (the

Where-Object cmdlet).

\* Selects only those virtual disk drive objects with an IDE bus type.

\* Displays the values for Name, BusType, Bus, and LUN for each selected

virtual disk drive.

NOTE: For more information about the standard Windows PowerShell ForEach-Ob

ject cmdlet (which uses "foreach" as its alias but is not the same as the W

indows PowerShell foreach loop statement), type: Get-Help ForEach-Object

REMARKS

For more information, type: "get-help Get-VirtualDiskDrive -detailed".

For technical information, type: "get-help Get-VirtualDiskDrive -full".

### New-VirtualDiskDrive

SYNOPSIS

Creates a virtual disk drive object on a virtual machine deployed on a host

managed by Virtual Machine Manager, or on a template in the Virtual Machin

e Manager library.

SYNTAX

New-VirtualDiskDrive -AnyHostDisk -Bus <Int32> -IDE <Boolean> -JobGroup <Gu

id> -LUN <Int32> [-BootVolume] [-JobVariable <String>] [-PROTipID <Guid>] [

-RunAsynchronously] [-SystemVolume] [-VMMServer [<String ServerConnection>]

] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -HostDisk <HostDisk> -IDE <Boolean> -LUN

<Int32> -VM [<String VM>] [-BootVolume] [-JobVariable <String>] [-PROTipID

<Guid>] [-RunAsynchronously] [-SystemVolume] [-VMMServer [<String ServerCon

nection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -Fixed -IDE <Boolean> -LUN <Int32> -Size

<Int32> -VM [<String VM>] [-BootVolume] [-FileName <String>] [-JobVariable

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-System

Volume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -AnyHostDisk -Bus <Int32> -JobGroup <Guid> -LUN <Int32

> -SCSI <Boolean> [-BootVolume] [-JobVariable <String>] [-PROTipID <Guid>]

[-RunAsynchronously] [-SystemVolume] [-VMMServer [<String ServerConnection>

]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -Fixed -JobGroup <Guid> -LUN <Int32> -SCS

I <Boolean> -Size <Int32> [-BootVolume] [-FileName <String>] [-JobVariable

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-System

Volume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -HostDisk <HostDisk> -LUN <Int32> -SCSI <

Boolean> -VM [<String VM>] [-BootVolume] [-FileName <String>] [-JobVariable

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Syste

mVolume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -Fixed -IDE <Boolean> -JobGroup <Guid> -L

UN <Int32> -Size <Int32> [-BootVolume] [-FileName <String>] [-JobVariable <

String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-SystemV

olume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -JobGroup <Guid> -LUN <Int32> -SCSI <Bool

ean> -VirtualHardDisk <VirtualHardDisk> [-BootVolume] [-FileName <String>]

[-JobVariable <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchrono

usly] [-SystemVolume] [-VMMServer [<String ServerConnection>]] [<CommonPara

meters>]

New-VirtualDiskDrive -Bus <Int32> -Dynamic -IDE <Boolean> -LUN <Int32> -Siz

e <Int32> -VM [<String VM>] [-BootVolume] [-FileName <String>] [-JobVariabl

e <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Syst

emVolume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -IDE <Boolean> -LUN <Int32> -Template [<T

emplate String>] -VirtualHardDisk <VirtualHardDisk> [-BootVolume] [-JobVari

able <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-SystemVolume] [-VM

MServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -LUN <Int32> -SCSI <Boolean> -Template [<

Template String>] -VirtualHardDisk <VirtualHardDisk> [-BootVolume] [-JobVar

iable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-SystemVolume] [-V

MMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -IDE <Boolean> -LUN <Int32> -VirtualHardD

isk <VirtualHardDisk> -VM [<String VM>] [-BootVolume] [-FileName <String>]

[-JobVariable <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchrono

usly] [-SystemVolume] [-VMMServer [<String ServerConnection>]] [<CommonPara

meters>]

New-VirtualDiskDrive -Bus <Int32> -LUN <Int32> -SCSI <Boolean> -VirtualHard

Disk <VirtualHardDisk> -VM [<String VM>] [-BootVolume] [-FileName <String>]

[-JobVariable <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchron

ously] [-SystemVolume] [-VMMServer [<String ServerConnection>]] [<CommonPar

ameters>]

New-VirtualDiskDrive -Bus <Int32> -Fixed -LUN <Int32> -SCSI <Boolean> -Size

<Int32> -VM [<String VM>] [-BootVolume] [-FileName <String>] [-JobVariable

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Syste

mVolume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -HostDisk <HostDisk> -IDE <Boolean> -JobG

roup <Guid> -LUN <Int32> [-BootVolume] [-JobVariable <String>] [-PROTipID <

Guid>] [-RunAsynchronously] [-SystemVolume] [-VMMServer [<String ServerConn

ection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -Dynamic -LUN <Int32> -SCSI <Boolean> -Si

ze <Int32> -VM [<String VM>] [-BootVolume] [-FileName <String>] [-JobVariab

le <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Sys

temVolume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -Dynamic -IDE <Boolean> -JobGroup <Guid>

-LUN <Int32> -Size <Int32> [-BootVolume] [-FileName <String>] [-JobVariable

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Syste

mVolume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -HostDisk <HostDisk> -JobGroup <Guid> -LU

N <Int32> -SCSI <Boolean> [-BootVolume] [-FileName <String>] [-JobVariable

<String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-System

Volume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -Dynamic -JobGroup <Guid> -LUN <Int32> -S

CSI <Boolean> -Size <Int32> [-BootVolume] [-FileName <String>] [-JobVariabl

e <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Syst

emVolume] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualDiskDrive -Bus <Int32> -IDE <Boolean> -JobGroup <Guid> -LUN <Int

32> -VirtualHardDisk <VirtualHardDisk> [-BootVolume] [-FileName <String>] [

-JobVariable <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronou

sly] [-SystemVolume] [-VMMServer [<String ServerConnection>]] [<CommonParam

eters>]

DETAILED DESCRIPTION

CCreates a virtual disk drive object on a virtual machine deployed on a hos

t managed by Virtual Machine Manager, or creates a virtual disk drive objec

t on a template in the Virtual Machine Manager library.

VMM 2008 introduces support for virtual disk drives, which were not used in

VMM 2007. You can attach a virtual hard disk or a pass-through disk (also

new in VMM 2008) to a virtual disk drive object on a virtual machine or on

a template that is used to create virtual machines.

A virtual hard disk file (a Windows-based .vhd file or a VMware-based.vmdk

file) that is stored on a Virtual Machine Manager library share, but is not

attached to a virtual disk drive, exists as a standalone object in the lib

rary.

A pass-through disk is a disk on a Hyper-V or VMware ESX host that a virtua

l machine on that host can use as an alternative to using a virtual hard di

sk. The corresponding term used by VMware for a pass-through disk is Raw De

vice Mapping, or RDM. The host disk can be either a hard disk on the host o

r a logical unit on a Storage Area Network (SAN). VMM lets the virtual mach

ine bypass the host's file system and access the pass-through disk directly

.

TYPE OF HOST PASS-THROUGH DISK SUPPORT

------------ -------------------------

Hyper-V Supports pass-through disks

Hyper-V Supports converting a pass-through disk to a VHD

Virtual Server Does not support pass-through disks

VMware ESX Supports pass-through disks (RDP), but not

disk conversion

NOTE: You cannot create a checkpoint (snapshot) of a pass-through disk beca

use checkpoint creation is designed to work with virtual hard disks.

For more information about VMM 2008 support for virtual disk drives, type:

Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements

PARAMETERS

-AnyHostDisk

Identifies a placeholder parameter that is used to indicate the creatio

n of pass-through disks within a new virtual machine job group.

-Bus <Int32>

Specifies the IDE bus to which to attach a virtual disk drive or virtua

l DVD drive, or the SCSI bus to which to attach a virtual disk drive.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter. For more information about how VMM 2008 impleme

nts IDE and SCSI buses, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enha

ncements.

-Dynamic

Specifies that a virtual hard disk can expand dynamically.

-Fixed

Specifies that a virtual hard disk is fixed in size.

-HostDisk <HostDisk>

Specifies a disk on a Hyper-V or VMware ESX host that a virtual machine

on that host can use instead of using a virtual hard disk. This disk i

s referrred to as a pass-through disk (the corresponding VMware term is

Raw Device Mapping, or RDM). The host disk is either a local hard disk

or a logical unit on a Storage Area Network (SAN). VMM lets the virtua

l machine bypass the host's file system and access the pass-through dis

k directly.

TYPE OF HOST PASS-THROUGH DISK SUPPORT

------------ -------------------------

Hyper-V Supports pass-through disks

Supports converting a pass-through disk to a VHD

VMware ESX Supports pass-through disks (RDP), but not disk conversi

on

Virtual Server Does not support pass-through disks

-IDE <Boolean>

Specifies IDE as the bus type to which to attach a virtual disk drive o

bject or a virtual DVD drive object configured on a virtual machine or

on a template. (For more information about how VMM 2008 implements the

IDE bus, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements.)

Example format: -IDE –Bus 0 –LUN 1

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-LUN <Int32>

Specifies the logical unit number (LUN) for a virtual disk drive object

or for a virtual DVD drive object on an IDE bus, or for a virtual disk

drive object on a SCSI bus.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

-SCSI <Boolean>

Specifies SCSI as the bus type to which to attach a virtual disk drive

object configured on a virtual machine or on a template.

Example format: -SCSI -Bus 0 -LUN 0

For information about the number of devices per controller on a SCSI bu

s that VMM 2008 supports for each virtualization platform (Hyper-V, Vir

tual Server, or VMware), type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enh

ancements.

-Size <Int32>

Specifies, in megabytes (MB), the size of a fixed virtual hard disk fil

e or the maximum possible size of a dynamically expanding virtual hard

disk file.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VirtualHardDisk <VirtualHardDisk>

Specifies a virtual hard disk object.

-VM [<String VM>]

Specifies a virtual machine object.

-BootVolume

Indicates that the volume attached to the VirtualDiskDrive is a boot vo

lume.

-FileName <String>

Specifies the file name to use when you rename a virtual hard disk file

as you add it to a virtual machine.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SystemVolume

Indicates that the volume attached to the VirtualDiskDrive is a system

volume.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a new virtual disk drive on a template and attach an existing vir

tual hard disk to the virtual disk drive.

PS C:\> Get-VMMServer -Computername "VMMServer1.contoso.com"

PS C:\> $VHD = Get-VirtualHardDisk | where {$\_.Name -eq "Blank Disk - Small

"}

PS C:\> $Template = Get-Template | where {$\_.Name -eq "Template01"}

PS C:\> New-VirtualDiskDrive -Template $Template -IDE -Bus 1 -Lun 1 -Virtua

lHardDisk $VHD

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command gets a virtual hard disk object from the VMM library by

name and stores the virtual hard disk object in variable $VHD.

The third command gets a template object from the library by name and store

s the template object in variable $Template.

The last command creates a new virtual disk drive on Template01 and attache

s the existing virtual hard disk represented by $VHD to the second channel

in the second slot of the IDE bus on the virtual disk drive.

2: Create a new virtual disk drive and add it to an existing virtual machin

e.

PS C:\> Get-VMMServer -Computername "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> New-VirtualDiskDrive -VM $VM -Dynamic -Filename "Test" -IDE -Size 2

00000 -Bus 0 -LUN 1

The first command connects to VMMServer1.

The second command gets the object for a virtual machine named VM02 and sto

res the virtual machine object in $VM.

The last command creates a new dynamic virtual disk drive on the first IDE

channel in the second slot of the virtual machine and specifies its size as

200 GB .

3: Create a new virtual disk drive from an existing VHD and attach it to a

new virtual machine.

PS C:\> $JobGroupID1 = [Guid]::NewGuid().ToString()

PS C:\> $VHD = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | wher

e {$\_.Location -eq "\\LibraryServer01.Contoso.com\MSSCVMMLibrary\VHDs\Blank

Disk - Large.vhd"}

PS C:\> New-VirtualDiskDrive -IDE -Bus 0 -LUN 1 -JobGroup $JobGroupId1 -Vir

tualHardDisk $VHD

PS C:\> $HwProfile = Get-HardwareProfile | where {$\_.Name -match "ServerPro

file"}

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -match "VMHost03"}

PS C:\> New-VM -Name "VM10" -Description "New Server VM10" -Owner "Contoso\

VMAdmin" -VMHost $VMHost -Path "D:\VMs" -HardwareProfile $HwProfile -JobGro

up $JobGroupID1 -RunAsynchronously -RunAsSystem -StartAction NeverAutoTurnO

nVM -UseHardwareAssistedVirtualization $FALSE -StopAction SaveVM

The first command generates a new globally unique identifier (GUID) and sto

res the GUID string in variable $JobGroupID1. The job group ID functions as

an identifier that groups subsequent commands that include $JobGroupID1 in

to a single job group.

The second command gets the object that represents a virtual hard disk from

the VMM library (saved in the file system at the location \\LibraryServer0

1.Contoso.com\MSSCVMMLibrary\VHDs\Blank Disk - Large.vhd) and stores the vi

rtual hard disk object in variable $VHD.

The third command performs the following operations:

\* Creates a new virtual disk drive object.

\* Attaches the virtual hard disk represented by $VHD to the new

virtual disk drive object.

\* Assigns the new virtual disk drive object to IDE Bus 0 and LUN 1.

\* Associates the new virtual disk drive object with JobGroupID1

(created in the first command).

The fourth command gets the object that represents an existing hardware pro

file (that contains the string "ServerProfile" in its name) and stores the

hardware profile object in $HwProfile.

The fifth command gets the object that represents a host named VMHost03 and

stores the host object in $VMHost.

The last command performs the following operations:

\* Creates a new VM named VM10 and deploys it on VMHost03.

\* Gives the new VM the hardware settings stored in $HwProfile, and

specifies a name, a description, an owner, and the path used to

store VM files.

\* Uses the -JobGroup parameter to specify that, before the New-VM

cmdlet creates the new virtual machine, any preceding cmdlet that

specifies the JobGroup GUID is executed. In this example,

New-VirtualDiskDrive will execute, and the resulting object is

automatically associated with the new VM.

\* Uses the -RunAsynchronously parameter to return control to the shell

immediately (before the command completes).

\* Uses the -RunAsSystem parameter to specify that the VM runs under the

local system account (it is only necessary to specify this for a VM on

a Virtual Server host because Hyper-V and VMware run a VM under the

local system account by default).

\* Specifies that the VM is not started automatically when the host starts

and that the VM is put into a saved state when the virtualization

service stops.

\* Sets hardware-assisted virtualization to an off state.

4. Create a new virtual disk drive using a host disk and attach it to an ex

isting VM.

clsPS C:\> Get-VMMServer "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM04"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost04.Contoso.com"

PS C:\> $HostDisk = @(Get-VMHostDisk -VMHost $VMHost | where {$\_.IsPassThro

ughCapable -eq $TRUE})

PS C:\> New-VirtualDiskDrive -VM $VM -HostDisk $HostDisk[0] -SCSI -Bus 0 -L

UN 1

The first command connects to VMMServer1.

The second command gets a virtual machine object by name (VM04) and stores

the virtual machine object in $VM.

The third command gets a host object by name and stores the host object in

$VMHost.

The fourth command gets from VMHost04 all objects that represents a host ha

rd disk drive that are pass-through capable and stores the hard disk drive

objects in $HostDisk. Using the "@" symbol and parentheses ensures that the

command stores the results in an array (in case the command returns a sing

le object or a null value).

The last command creates a new virtual disk drive object that is connected

to a physical host disk on VMHost04. The virtual disk drive is attached to

the second slot of the first SCSI bus on VM04. This example assumes the vir

tual machine already has a SCSI controller.

REMARKS

For more information, type: "get-help New-VirtualDiskDrive -detailed".

For technical information, type: "get-help New-VirtualDiskDrive -full".

### Remove-VirtualDiskDrive

SYNOPSIS

Removes a virtual disk drive object from a virtual machine or from a templa

te in a Virtual Machine Manager environment.

SYNTAX

Remove-VirtualDiskDrive [-VirtualDiskDrive] <VirtualDiskDrive> [-Force] [-J

obGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronou

sly] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more virtual disk drive objects from a virtual machine or fr

om a template in a Virtual Machine Manager environment.

PARAMETERS

-VirtualDiskDrive <VirtualDiskDrive>

Specifies a virtual disk drive object. You can attach either a virtual

hard disk (for a virtual machine on any host) or a pass-through disk (f

or a virtual machine on a Hyper-V host or an ESX host) to a virtual dis

k drive object.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the second virtual disk drive object from the specified virtual m

achine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.contoso.com"

PS C:\> $VM = Get-VM | where { $\_.VMHost.Name -eq "VMHost01.Contoso.com" -a

nd $\_.Name -eq "VM01" }

PS C:\> $VDD = @(Get-VirtualDiskDrive -VM $VM)

PS C:\> if($VDD.Count -gt 1){Remove-VirtualDiskDrive -VirtualDiskDrive $VDD

[1]}

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object from the VMM database that represents th

e VM named VM01 (deployed on VMHost01) and stores the VM object in variable

$VM.

The third command gets all virtual disk drive objects on VM01 and stores th

e retrieved objects in variable $VDD. Using the '@' symbol and parentheses

ensures that the command stores the results in an array (in case the comman

d returns a single object or a null value).

The last command returns the number of virtual disk drives associated with

the virtual machine and then, if more than one exists, the command removes

the second ([1]) virtual disk drive from the virtual machine.

2: Remove all pass-through disks attached to a VM.

PS C:\> $VMM = Get-VMMServer -ComputerName "VMMServer1.contoso.com"

PS C:\> $VM = Get-VM | where {$\_.Name -eq "VM02"}

PS C:\> $VDDs = @(Get-VirtualDiskDrive -VM $VM | where {$\_.IsVHD -eq $FALSE

})

PS C:\> if($VDDs.Count -gt 0){foreach($VDD in $VDDs){Remove-VirtualDiskDriv

e -Force -VirtualDiskDrive $VDD}}

The first command connects to VMMServer1.

The second command gets the object that represents a VM named VM02 and stor

es the VM object in $VM.

The third command gets all objects that represent virtual disk drives attac

hed to VM02 that are not virtual hard disks (that is, only objects that rep

resent pass-through disks are retrieved) and stores the pass-through disk o

bjects in variable $VDDs (an object array).

The last command uses an if statement to determine whether at least one pas

s-through virtual disk drive exists. If the result is one or more, the comm

and then uses the foreach statement to remove each virtual disk drive from

the object array. Using the Force parameter ensures the removal of each vir

tual disk drive from its virtual machine even if other VMM objects depend o

n that virtual disk drive.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

3: Remove virtual disk drives attached to a VM by name.

PS C:\> Get-VMMServer -Computername "VMMServer1.contoso.com"

PS C:\> $VMs = @(Get-VM | where {$\_.Name -match "WebSrvLOB"})

PS C:\> foreach($VM in $VMs){$VDDs = Get-VirtualDiskDrive -VM $VM; foreach(

$VDD in $VDDs){if($VDD.Name -match "LOBData"){Remove-VirtualDiskDrive -Virt

ualDiskDrive $VDD}}}

The first command connects to VMMServer1.

The second command gets all objects that represent VMs whose name match a s

pecific string (in this case, the command gets all Line of Business Web ser

vers, indicated by the name "WebSrvLOB"). The command stores the retrieved

VM objects in $VM (an object array).

The last command uses "foreach" (the ForEach-Object cmdlet) to iterate thro

ugh the array of VMs returned in the second command and performs the follow

ing operations:

\* Gets all virtual disk drive objects from each VM in the $VM object

array and stores these virtual disk drive objects in the object

array $VDDs.

\* Uses a second foreach loop to select all virtual disk drive objects

stored in $VDDs whose name contains the string LOBData, and then

uses the Remove-VirtualDiskDrive cmdlet to delete these objects

from the VMM database.

REMARKS

For more information, type: "get-help Remove-VirtualDiskDrive -detailed".

For technical information, type: "get-help Remove-VirtualDiskDrive -full".

### Set-VirtualDiskDrive

SYNOPSIS

Modifies settings on a virtual disk drive object on a virtual machine or on

a template in a Virtual Machine Manager environment.

SYNTAX

Set-VirtualDiskDrive -VirtualDiskDrive <VirtualDiskDrive> [-Bus <Int32>] [-

IDE <Boolean>] [-JobGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>] [-

PROTipID <Guid>] [-RunAsynchronously] [-SCSI <Boolean>] [<CommonParameters>

]

DETAILED DESCRIPTION

Modifies settings on a virtual disk drive object on a virtual machine or on

a template in a Virtual Machine Manager environment. You can use this cmdl

et to change the Bus type (IDE or SCSI), or to change the Bus and LUN setti

ngs to connect a virtual disk drive to a different location on the bus.

For more information about VMM 2008 support for virtual disk drives, type:

Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements

PARAMETERS

-VirtualDiskDrive <VirtualDiskDrive>

Specifies a virtual disk drive object. You can attach either a virtual

hard disk (for a virtual machine on any host) or a pass-through disk (f

or a virtual machine on a Hyper-V host or an ESX host) to a virtual dis

k drive object.

-Bus <Int32>

Specifies the IDE bus to which to attach a virtual disk drive or virtua

l DVD drive, or the SCSI bus to which to attach a virtual disk drive.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter. For more information about how VMM 2008 impleme

nts IDE and SCSI buses, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enha

ncements.

-IDE <Boolean>

Specifies IDE as the bus type to which to attach a virtual disk drive o

bject or a virtual DVD drive object configured on a virtual machine or

on a template. (For more information about how VMM 2008 implements the

IDE bus, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements.)

Example format: -IDE –Bus 0 –LUN 1

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LUN <Int32>

Specifies the logical unit number (LUN) for a virtual disk drive object

or for a virtual DVD drive object on an IDE bus, or for a virtual disk

drive object on a SCSI bus.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SCSI <Boolean>

Specifies SCSI as the bus type to which to attach a virtual disk drive

object configured on a virtual machine or on a template.

Example format: -SCSI -Bus 0 -LUN 0

For information about the number of devices per controller on a SCSI bu

s that VMM 2008 supports for each virtualization platform (Hyper-V, Vir

tual Server, or VMware), type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enh

ancements.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the IDE bus and LUN settings for a virtual disk drive on a virtua

l machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where {$\_.Name -eq "VM01"}

PS C:\> $VDD = @(Get-VirtualDiskDrive -VM $VM)

PS C:\> if($VDD.Count -eq 1 -and $VDD[0].Bus -eq 0 -and $VDD[0].Lun

-eq 1){Set-VirtualDiskDrive -VirtualDiskDrive $VDD[0] -Bus 0 -LUN 0}

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command gets the virtual machine object VM01 and stores it in va

riable $VM.

The third command gets the object that represents the virtual disk drive on

VM01 and stores the virtual disk drive object in variable $VDD. Using the

'@' symbol and parentheses ensures that the command stores the results in a

n array (in case the command returns a single object or a null value).

The last command sets the Bus value to 0 and sets the LUN value to 0 for th

e virtual disk drive on VM01 if the VM only has one virtual disk drive and

is located on the second slot of the first IDE channel.

2: Change the bus type for a virtual disk drive from SCSI to IDE.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $VDDs = Get-VirtualDiskDrive -VM $VM

PS C:\> Set-VirtualDiskDrive -VirtualDiskDrive $VDDs[1] -IDE -Bus 0 -LUN 1

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command gets all objects that represent virtual disk drives confi

gured for this virtual machine and stores the virtual disk drive objects in

$VDDs (an object array). This example assumes that the virtual disk drive

is on a SCSI bus.

The last command sets the Bus type to IDE and connects the second virtual d

isk drive (specified by $VDDs[1]) to Primary Channel (1) and slot 2 (specif

ied by -Bus 0 and LUN 1).

REMARKS

For more information, type: "get-help Set-VirtualDiskDrive -detailed".

For technical information, type: "get-help Set-VirtualDiskDrive -full".

## VirtualDVDDrive

### Get-VirtualDVDDrive

SYNOPSIS

Gets Virtual Machine Manager virtual DVD drive objects from a virtual machi

ne, template, or hardware profile.

SYNTAX

Get-VirtualDVDDrive -Template [<Template String>] [<CommonParameters>]

Get-VirtualDVDDrive -All [-VMMServer [<String ServerConnection>]] [<CommonP

arameters>]

Get-VirtualDVDDrive -HardwareProfile <HardwareProfile> [<CommonParameters>]

Get-VirtualDVDDrive -VM [<String VM>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual DVD drives in a Virtual Mac

hine Manager environment from a virtual machine object, from a template obj

ect, or from a hardware profile object.

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get virtual DVD drives from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Get-VirtualDVDDrive -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The last command gets all objects that represent virtual DVD drives on VM01

and displays information about these virtual DVD drives to the user.

2: Get virtual DVD drives from a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> Get-VirtualDVDDrive -Template $Template

The first command connects to VMMServer1.

The second command selects from all template objects stored in the VMM libr

ary the one named Template1 and stores this template object in $Template.

The last command gets all objects that represent virtual DVD drives on Temp

late1 and displays information about these virtual DVD drives to the user.

3: Get virtual DVD drives from a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> Get-VirtualDVDDrive -HardwareProfile $HWProfile

The first command connects to VMMServer1.

The second command selects from all hardware profile objects in the VMM lib

rary the one named HardwareProfile1 and stores this hardware profile object

in $HWProfile.

The last command gets the objects that represent all virtual DVD drives on

HardwareProfile1 and displays information about these virtual DVD drives to

the user.

REMARKS

For more information, type: "get-help Get-VirtualDVDDrive -detailed".

For technical information, type: "get-help Get-VirtualDVDDrive -full".

### New-VirtualDVDDrive

SYNOPSIS

Creates a virtual DVD drive on a virtual machine, template, or hardware pro

file used in Virtual Machine Manager.

SYNTAX

New-VirtualDVDDrive -Bus <Int32> -JobGroup <Guid> -LUN <Int32> [-AnyHostDri

ve] [-HostDrive <String>] [-ISO <ISO>] [-JobVariable <String>] [-Link] [-PR

OTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String ServerConnection>]

] [<CommonParameters>]

New-VirtualDVDDrive -Bus <Int32> -LUN <Int32> -VM [<String VM>] [-AnyHostDr

ive] [-HostDrive <String>] [-ISO <ISO>] [-JobVariable <String>] [-Link] [-P

ROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

New-VirtualDVDDrive -Bus <Int32> -HardwareProfile <HardwareProfile> -LUN <I

nt32> [-AnyHostDrive] [-HostDrive <String>] [-ISO <ISO>] [-JobVariable <Str

ing>] [-Link] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

New-VirtualDVDDrive -Bus <Int32> -LUN <Int32> -Template [<Template String>]

[-AnyHostDrive] [-HostDrive <String>] [-ISO <ISO>] [-JobVariable <String>]

[-Link] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual DVD drive on a virtual machine, template, or hardware pro

file used in a Virtual Machine Manager environment.

By default, the virtual DVD drive created by using New-VirtualDVDDrive is n

ot connected to any media. You can use the Set-VirtualDVDDrive cmdlet to co

nnect a virtual DVD drive (which you created by using New-VirtualDVDDrive)

to a physical DVD drive on a virtual machine host or to connect it to an IS

O image.

NOTE: You can connect a virtual DVD drive to an IDE device on a virtual mac

hine but you cannot connect a virtual DVD drive to a SCSI adapter on a virt

ual machine.

For more information about how to connect a virtual DVD drive, type:

Get-Help Set-VirtualDVDrive -detailed

PARAMETERS

-Bus <Int32>

Specifies the IDE bus to which to attach a virtual disk drive or virtua

l DVD drive, or the SCSI bus to which to attach a virtual disk drive.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter. For more information about how VMM 2008 impleme

nts IDE and SCSI buses, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enha

ncements.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-LUN <Int32>

Specifies the logical unit number (LUN) for a virtual disk drive object

or for a virtual DVD drive object on an IDE bus, or for a virtual disk

drive object on a SCSI bus.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-AnyHostDrive

Specifies that a virtual DVD or floppy drive will be connected to any c

orresponding physical drive on a host when you deploy a stored virtual

machine on a host, or when you use a template or hardware profile to cr

eate and deploy a virtual machine on a host.

-HostDrive <String>

Specifies a drive on a virtual machine host.

Example formats:

Hard Drive Floppy Drive

---------- ------------

Windows-based host "C:" "A:"

VMware ESX host "/dev/tools" "/dev/sda"

-ISO <ISO>

Specifies an ISO object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Link

Specifies that a resource should be linked to instead of copied.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a virtual DVD drive on a VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> New-VirtualDVDDrive -VM $VM -Bus 1 -LUN 1

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VM01 from the VMM databa

se and stores the virtual machine object in variable $VM.

The last command creates a virtual DVD drive on VM01 and attaches the virtu

al DVD drive to Secondary Channel (1) by specifying IDE Bus 1 and LUN 1.

2: Create a virtual DVD drive on a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> New-VirtualDVDDrive -Template $Template -Bus 1 -LUN 1

The first command connects to VMMServer1.

The second command gets the object that represents Template1 and stores the

template object in $Template.

The last command creates a virtual DVD drive on Template1 that, when the te

mplate is used to create a virtual machine, will attach the virtual DVD dri

ve to Secondary Channel (1) on the IDE bus.

3: Create a virtual DVD drive on a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> New-VirtualDVDDrive -HardwareProfile $HWProfile -Bus 1 -LUN 1

The first command connects to VMMServer1.

The second command gets the object that represents HardwareProfile1 and sto

res the hardware profile object in $HWProfile.

The last command creates a virtual DVD drive on HardwareProfile1 that, when

the hardware profile is used to create a virtual machine, will attach the

virtual DVD drive to Secondary Channel (1) on the IDE bus.

4. Create a VM with a virtual DVD drive that connects to any available phys

ical DVD drive on the host.

PS C:\> $JobGroupId = [Guid]::NewGuid().ToString()

PS C:\> New-VirtualDVDDrive -VMMServer "VMMServer1.contoso.com" -JobGroup $

JobGroupId -Bus 1 -LUN 0 -AnyHostDrive

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -match "VMHost04"}

PS C:\> New-VM -Name "VM04" -Description "This is my new VM with a DVD driv

e" -Owner "Contoso\User1" -VMHost $VMHost -Path "F:\" -StartVM -JobGroup $J

obGroupId

The first command creates a new GUID string and stores it to variable $JobG

roupID. This GUID is a job group ID that functions as an identifier that gr

oups subsequent commands that include this identifier into a single job gro

up.

The second command creates a new virtual DVD drive object and specifies tha

t this new virtual DVD drive can use any available physical DVD drive. The

command will attach the new virtual DVD drive to the first slot of the seco

nd IDE channel (IDE is the only bus type that a virtual DVD drive can be at

tached to). Using the job group ID specifies that that this command will no

t run until just before the final command that includes "–JobGroup $JobGrou

pID" runs.

The third command gets a host object by name and stores the host object in

$VMHost.

The last command creates a virtual machine, names it VM04, provides a descr

iption, assigns an owner, and specifies the path on the host to the locatio

n where the virtual machine will be stored. The command uses the job group

ID to run the New-VirtualDVDDrive command just before the New-VM command r

uns; the resulting virtual DVD drive object will be associated with the new

virtual machine.

5. Add a new virtual DVD drive to an existing VM and attach an ISO from the

library to it.

PS C:\> Get-VMMServer "VMMServer01.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM05"

PS C:\> $ISO = Get-ISO | where {$\_.Name -eq "Windows Server 2003.iso"}

PS C:\> New-VirtualDVDDrive -VM $VM -ISO $ISO -Bus 1 -LUN 1

The first command connects to VMMServer1.

The second command gets the object that represents VM05 and stores the obje

ct in $VM.

The third command gets the object for an ISO that contains the Windows Serv

er 2003 operating system and stores the ISO object in $ISO.

The last command creates a new virtual DVD drive on VM05 and attaches it to

the specified location on the IDE bus.

REMARKS

For more information, type: "get-help New-VirtualDVDDrive -detailed".

For technical information, type: "get-help New-VirtualDVDDrive -full".

### Remove-VirtualDVDDrive

SYNOPSIS

Removes a virtual DVD drive object from Virtual Machine Manager.

SYNTAX

Remove-VirtualDVDDrive [-VirtualDVDDrive] <VirtualDVDDrive> [-JobGroup <Gui

d>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<Commo

nParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent virtual DVD drives from a hardwa

re profile, virtual machine, or template used in a Virtual Machine Manager

environment. The cmdlet also deletes from the file system on the library se

rver any .iso file that the virtual DVD drive uses.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualDVDDrive <VirtualDVDDrive>

Specifies a virtual DVD drive object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific virtual DVD drive from a VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $DVDDrive = Get-VirtualDVDDrive -VM $VM | where { $\_.Bus –eq 0 -and

$\_.LUN –eq 1 }

PS C:\> Remove-VirtualDVDDrive -VirtualDVDDrive $DVDDrive

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command gets the object that represents the virtual DVD drive tha

t is located on Primary Channel 1 (specified by –Bus 0 and –LUN 1) on the I

DE bus on VM01. The command stores the virtual DVD drive object in variable

$DVDDrive.

The last command removes the virtual DVD drive object stored in $DVDDrive f

rom VM01 and deletes any .iso file that this virtual DVD drive uses from th

e file system on the library server.

2: Remove the third virtual DVD drive from a VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $DVDDrive = Get-VirtualDVDDrive -VM $VM

PS C:\> $DVDDrive[2] | Remove-VirtualDVDDrive

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02and stores the virtual machine object in $VM.

The third command gets all objects that represent virtual DVD drives connec

ted to VM02 and stores each virtual DVD drive object in $DVDDrive (an objec

t array). This example assumes that VM02 has three virtual DVD drives and t

herefore the array contains three elements (counting 0 to 2).

The last command passes the third virtual DVD drive (object [2]) stored in

$DVDDrive to the Remove-VirtualDVDDrive cmdlet, which removes this virtual

DVD drive object from VM01 and deletes any .iso file used by this virtual D

VD drive from the file system on the library server.

REMARKS

For more information, type: "get-help Remove-VirtualDVDDrive -detailed".

For technical information, type: "get-help Remove-VirtualDVDDrive -full".

### Set-VirtualDVDDrive

SYNOPSIS

Changes properties of a virtual DVD drive associated with a virtual machine

, template, or hardware profile used in Virtual Machine Manager.

SYNTAX

Set-VirtualDVDDrive [-VirtualDVDDrive] <VirtualDVDDrive> -NoMedia <Boolean>

[-Bus <Int32>] [-JobGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>] [

-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Set-VirtualDVDDrive [-VirtualDVDDrive] <VirtualDVDDrive> -HostDrive <String

> [-Bus <Int32>] [-JobGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>]

[-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Set-VirtualDVDDrive [-VirtualDVDDrive] <VirtualDVDDrive> [-AnyHostDrive] [-

Bus <Int32>] [-JobGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>] [-PR

OTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Set-VirtualDVDDrive [-VirtualDVDDrive] <VirtualDVDDrive> -ISO <ISO> [-Bus <

Int32>] [-JobGroup <Guid>] [-JobVariable <String>] [-Link] [-LUN <Int32>] [

-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Set-VirtualDVDDrive [-VirtualDVDDrive] <VirtualDVDDrive> [-Bus <Int32>] [-J

obGroup <Guid>] [-JobVariable <String>] [-LUN <Int32>] [-PROTipID <Guid>] [

-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual DVD drive associated with a vir

tual machine, template, or hardware profile used in a Virtual Machine Manag

er environment.

You can use this cmdlet to connect the virtual DVD drive to a physical DVD

drive on a virtual machine host server, to a different location on the IDE

bus, to an ISO image, or you can use it to disconnect the virtual DVD drive

.

Most settings that you can configure for a virtual DVD drive on a virtual m

achine are the same regardless of whether the virtualization platform of th

e host is Virtual Server, Hyper-V, or VMware. All support:

\* Connecting a virtual DVD drive to a primary or secondary channel on

a host.

\* Capturing information from a physical CD or DVD drive on the host

without specifying a drive letter.

\* Capturing information from an image file (FileName.iso) stored in

the VMM library.

\* Capturing "no media" (used to disconnect a virtual DVD drive from

the host drive or from an ISO to which it was connected earlier).

The only setting for a virtual machine managed by VMM that does vary for th

is cmdlet depending on the virtualization platform of the host is whether o

r not an ISO file can be used directly from the VMM library:

\* Virtual Server or Hyper-V host. If you configure a connection to an

ISO file in the VMM library, you can, optionally, choose to use the

ISO directly from the library instead of copying it to the host.

\* VMware ESX host. If you configure a connection to an ISO file in the

VMM library, you cannot use the ISO directly from the library but must

instead accept the default, which copies the ISO file to the host.

Note: If the virtual DVD drive is configured on a virtual machine that was

created by using the Virtual Machine wizard in the Hyper-V Manager Console

rather than in the VMM Administrator Console, you must specify a drive lett

er. That drive letter will appear in the Properties for that virtual machin

e in the VMM Administrator Console.

For more information about VMM 2008 support for virtual DVD drives, type:

Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements

PARAMETERS

-VirtualDVDDrive <VirtualDVDDrive>

Specifies a virtual DVD drive object.

-HostDrive <String>

Specifies a drive on a virtual machine host.

Example formats:

Hard Drive Floppy Drive

---------- ------------

Windows-based host "C:" "A:"

VMware ESX host "/dev/tools" "/dev/sda"

-ISO <ISO>

Specifies an ISO object.

-NoMedia <Boolean>

Disconnects a virtual DVD drive from the host drive or ISO to which it

was connected, or disconnects a virtual floppy drive from the host driv

e or virtual floppy disk to which it was connected.

-AnyHostDrive

Specifies that a virtual DVD or floppy drive will be connected to any c

orresponding physical drive on a host when you deploy a stored virtual

machine on a host, or when you use a template or hardware profile to cr

eate and deploy a virtual machine on a host.

-Bus <Int32>

Specifies the IDE bus to which to attach a virtual disk drive or virtua

l DVD drive, or the SCSI bus to which to attach a virtual disk drive.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter. For more information about how VMM 2008 impleme

nts IDE and SCSI buses, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enha

ncements.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Link

Specifies that a resource should be linked to instead of copied.

-LUN <Int32>

Specifies the logical unit number (LUN) for a virtual disk drive object

or for a virtual DVD drive object on an IDE bus, or for a virtual disk

drive object on a SCSI bus.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Connect a virtual DVD drive to a physical DVD drive.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $DVDDrive = Get-VirtualDVDDrive -VM $VM | where { $\_.Bus -eq 1 -and

$\_.LUN -eq 0 }

PS C:\> Set-VirtualDVDDrive -VirtualDVDDrive $DVDDrive -HostDrive "D:"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command gets the object that represents the virtual DVD drive tha

t is located on Secondary Channel 0 (specified by -Bus 1 and -LUN 0) on the

IDE bus on VM01. The command stores the virtual DVD drive object in variab

le $DVDDrive.

The last command connects the virtual DVD drive whose object is stored in $

DVDDrive to a physical drive on the host (the D: drive). It also deletes an

y ISO file that the virtual DVD drive used earlier if no other virtual mach

ine currently uses that ISO file.

2: Connect a virtual DVD drive to a different location on the IDE bus.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $DVDDrive = Get-VirtualDVDDrive -VM $VM | where { $\_.Bus -eq 1 -and

$\_.LUN -eq 0 }

PS C:\> Set-VirtualDVDDrive -VirtualDVDDrive $DVDDrive -Bus 1 -LUN 1

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command gets the object that represents the virtual DVD drive tha

t is located on Secondary Channel 0 (specified by -Bus 1 and -LUN 0) on the

IDE bus on VM02. The command stores the virtual DVD drive object in $DVDDr

ive.

The last command connects the virtual DVD drive whose object is stored in $

DVDDrive to a different position on the IDE bus by setting the logical unit

number (LUN) to 1.

3: Disconnect a virtual DVD drive.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> $DVDDrive = Get-VirtualDVDDrive -VM $VM | where { $\_.Bus -eq 1 -and

$\_.LUN -eq 0 }

PS C:\> Set-VirtualDVDDrive -VirtualDVDDrive $DVDDrive -NoMedia

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM03 and stores the virtual machine object in $VM.

The third command gets the object that represents the virtual DVD drive tha

t is located on Secondary Channel 0 (specified by -Bus 1 and -LUN 0) on the

IDE bus on VM03. The command stores the virtual DVD drive object in $DVDDr

ive.

The last command uses the NoMedia parameter to disconnect the virtual DVD d

rive whose object is stored in $DVDDrive from any host drive or ISO to whic

h it was connected earlier. It also deletes any ISO file that the virtual

DVD drive used earlier if no other virtual machine currently uses that ISO

file.

4. Connect a virtual DVD drive on an existing VM to any available physical

DVD drive.

PS C:\> Get-VMMServer "VMMServer01.contoso.com"

PS C:\> $VM = Get-VM -Name "VM04"

PS C:\> Set-VirtualDVDDrive -AnyHostDrive -VirtualDVDDrive (Get-VirtualDVDD

rive -VM $VM | where {$\_.Bus -eq 1 -and $\_.Lun -eq 0})

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM04 and stores the virtual machine object in $VM.

The last command gets the object that represents the virtual DVD drive that

is located on the first slot of the Secondary Channel (specified by -Bus 1

and -LUN 0) on the IDE bus on VM04. The command uses the Set-VirtualDVDDri

ve cmdlet with the AnyHostDrive parameter to connect the virtual DVD drive

to any available physical DVD drive on the host.

REMARKS

For more information, type: "get-help Set-VirtualDVDDrive -detailed".

For technical information, type: "get-help Set-VirtualDVDDrive -full".

## VirtualFloppyDisk

### Get-VirtualFloppyDisk

SYNOPSIS

Gets virtual floppy disk objects from the Virtual Machine Manager library.

SYNTAX

Get-VirtualFloppyDisk [-All] [-VMMServer [<String ServerConnection>]] [<Com

monParameters>]

Get-VirtualFloppyDisk [-ID <Guid>] [-VMMServer [<String ServerConnection>]]

[<CommonParameters>]

DETAILED DESCRIPTION

Gets from the Virtual Machine Manager library one or more objects that repr

esent Virtual Machine Manager virtual floppy disk files (either a Windows-b

ased .vfd file or a VMware-based .flp file). The virtual floppy disk file t

hat a virtual floppy disk object represents is stored on a library server.

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all virtual floppy disks on all VMM library servers.

PS C:\> Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.com

Gets all objects that represent virtual floppy disk files from the VMM libr

ary and displays information about these virtual floppy disk objects to the

user. The virtual floppy disk files themselves (which the objects represen

t) are stored in library shares on library servers.

2: Get all virtual floppy disks on a specific VMM library server.

PS C:\> Get-VirtualFloppyDisk –VMMServer VMMServer1.Contoso.com | where { $

\_.LibraryServer.Name –eq "FileServer01.Contoso.com" }

Gets all objects that represent virtual floppy disks stored on library serv

er FileServer01 and displays information about these virtual floppy disks t

o the user.

3: Get all virtual floppy disks with a specific name on any VMM library ser

ver.

PS C:\> Get-VirtualFloppyDisk | where { $\_.Name -eq "BootFloppy.vfd" }

Gets all objects that represent virtual floppy disks named BootFloppy.vfd t

hat are stored on any library server managed by VMM and displays informatio

n about these virtual floppy disk objects to the user.

NOTE: In VMM 2008, by default, the name of a virtual floppy disk object in

the library is the same name (including the extension) as the name of the a

ctual virtual floppy disk file on the library server.

REMARKS

For more information, type: "get-help Get-VirtualFloppyDisk -detailed".

For technical information, type: "get-help Get-VirtualFloppyDisk -full".

### Remove-VirtualFloppyDisk

SYNOPSIS

Removes a virtual floppy disk object from Virtual Machine Manager.

SYNTAX

Remove-VirtualFloppyDisk [-VirtualFloppyDisk] <VirtualFloppyDisk> [-Confirm

] [-Force] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously]

[<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents a virtual floppy disk from the Virtual Ma

chine Manager library and deletes the corresponding virtual floppy disk fil

e (a Windows-based .vfd file or a VMware-based .flp file) from the library

server.

If the virtual floppy disk is attached to a virtual machine, template, or h

ardware profile (and if you do not use the Force parameter), VMM lists the

container that contains the virtual floppy disk and prompts you to confirm

that you want to remove the virtual floppy disk:

\* If you reply Yes, VMM removes the association between the virtual

floppy disk and the container to which it is attached, and then

deletes the virtual floppy disk object from VMM.

\* If you reply No, the operation is cancelled.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualFloppyDisk <VirtualFloppyDisk>

Specifies a virtual floppy disk object.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a virtual floppy disk object from the library and delete the corr

esponding file.

PS C:\> $Vfd = Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.com | w

here { $\_.Name -eq “BootFloppy1.vfd” -and $\_.LibraryServer.Name -eq “FileSe

rver01.Contoso.com” }

PS C:\> Remove-VirtualFloppyDisk -VirtualFloppyDisk $Vfd

The first command gets the object that represents the virtual floppy disk f

ile named BootFloppy1.vfd (stored on library server FileServer01) and store

s the virtual floppy disk object in variable $Vfd.

The second command removes the BootFloppy1.vfd object from the library and

deletes the corresponding virtual floppy disk file from the library server.

In this example, the virtual floppy disk is a .vfd file rather than an .fl

p file.

NOTE: This example assumes that only one virtual floppy disk object named B

ootFloppy1.vfd exists. For an example that illustrates how to remove multip

le virtual floppy disk objects, see Example 2.

2: Remove multiple virtual floppy disks and their files.

PS C:\> $Vfds = Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.com | w

here { $\_.Name -match “Boot” }

PS C:\> $Vfds | Remove-VirtualFloppyDisk

The first command gets all objects that represent virtual floppy disk files

whose names include the string “Boot” and stores these virtual floppy disk

objects in $Vfds (an object array).

The second command passes each virtual floppy disk object in $Vfds to Remov

e-VirtualFloppyDisk, which removes each virtual floppy disk object from the

library. The command also deletes each corresponding file (a .vfd file or

a .flp file) from the library server on which that virtual floppy disk is s

tored.

REMARKS

For more information, type: "get-help Remove-VirtualFloppyDisk -detailed".

For technical information, type: "get-help Remove-VirtualFloppyDisk -full".

### Set-VirtualFloppyDisk

SYNOPSIS

Changes properties of a virtual floppy disk used in Virtual Machine Manager

.

SYNTAX

Set-VirtualFloppyDisk [-VirtualFloppyDisk] <VirtualFloppyDisk> [-Descriptio

n <String>] [-Enabled <Boolean>] [-JobVariable <String>] [-Name <String>] [

-Owner <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-SharePath <Strin

g>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual floppy disk used in a Virtual M

achine Manager environment. A virtual floppy disk file used in VMM 2008 is

either a Windows-based .vfd file or a VMware-based .flp file.

Properties that you can change include:

- Description

- Enabled

- Name

- Owner

- SharePath

You can store a virtual floppy disk file in the Virtual Machine Manager lib

rary, or you can add the virtual floppy disk to a virtual machine.

PARAMETERS

-VirtualFloppyDisk <VirtualFloppyDisk>

Specifies a virtual floppy disk object.

-Description <String>

Specifies a description for the specified object.

-Enabled <Boolean>

Enables a library object (when set to TRUE) or disables a library objec

t (when set to FALSE). For example, if you want to upgrade software on

a virtual machine template, you can disable the template object in the

VMM library to temporarily prevent users from using that object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the description of a virtual floppy disk.

PS C:\> $Vfd = Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.com | w

here { $\_.Name -eq “BootFloppy.vfd” -and $\_.LibraryServer.Name -eq “FileSer

ver01.Contoso.com” }

PS C:\> Set-VirtualFloppyDisk -VirtualFloppyDisk $Vfd -Description “Latest

Boot Floppy”

The first command gets from the VMM library on VMMServer1 the object that r

epresents the virtual floppy disk named BootFloppy.vfd whose file is stored

on the library server named FileServer01. The command stores the virtual f

loppy disk object in variable $Vfd.

The second command changes the description of this virtual floppy disk obje

ct to "Latest Boot Floppy."

2: Disable a virtual floppy disk.

PS C:\> $Vfd = Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.com | wh

ere { $\_.Name -eq “BootFloppy.vfd” -and $\_.LibraryServer.Name -eq “FileServ

er01.Contoso.com” }

PS C:\> Set-VirtualFloppyDisk -VirtualFloppyDisk $Vfd -Enabled $FALSE

The first command gets the object that represents the virtual floppy disk n

amed BootFloppy.vfd (whose file is stored on the library server named FileS

erver01) and stores the virtual floppy disk object in $Vfd.

The second command disables the virtual floppy disk in $Vfd.

3: Change the name of the object that represents a VMware-based virtual flo

ppy disk.

PS C:\> $FLP = Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.com | wh

ere { $\_.Name -eq “ESXBootFloppy.flp” -and $\_.LibraryServer.Name -eq “FileS

erver01.Contoso.com” }

PS C:\> Set-VirtualFloppyDisk -VirtualFloppyDisk $FLP -Name “ESXBootFloppy-

Beta.flp”

The first command gets the object that represents the virtual floppy disk n

amed ESXBootFloppy.flp on library server FileServer01. The command stores t

he virtual floppy disk object in variable $FLP.

The second command changes the name of this virtual floppy disk object to E

SXBootFloppy-Beta.flp.

NOTE: In VMM 2008, by default, the name of a virtual floppy disk object in

the library is the same name (including the extension) as the name of the a

ctual virtual floppy disk file on the library share (in this case, ESXBootF

loppy.flp). Changing the name of the virtual floppy disk object in the libr

ary to ESXBootFloppy-Beta.flp does not change the name of the actual virtua

l floppy disk file (ESXBootFloppy.flp) stored on FileServer01.

4: Specify an owner for all virtual floppy disks with an "Unknown" owner.

PS C:\> Get-VirtualFloppyDisk -VMMServer "VMMServer1.Contoso.com" | where {

$\_.Owner -eq "Unknown"} | Set-VirtualFloppyDisk -Owner "Contoso\IsabelMarti

ns"

Gets all virtual floppy disk objects from the VMM library, selects only tho

se virtual floppy disks whose owner is "Unknown," and specifies an owner fo

r each virtual floppy disk object.

REMARKS

For more information, type: "get-help Set-VirtualFloppyDisk -detailed".

For technical information, type: "get-help Set-VirtualFloppyDisk -full".

## VirtualFloppyDrive

### Get-VirtualFloppyDrive

SYNOPSIS

Gets Virtual Machine Manager virtual floppy drive objects from a virtual ma

chine, template, or hardware profile.

SYNTAX

Get-VirtualFloppyDrive -HardwareProfile <HardwareProfile> [<CommonParameter

s>]

Get-VirtualFloppyDrive -All [-VMMServer [<String ServerConnection>]] [<Comm

onParameters>]

Get-VirtualFloppyDrive -VM [<String VM>] [<CommonParameters>]

Get-VirtualFloppyDrive -Template [<Template String>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual floppy drives in a Virtual

Machine Manager environment from a virtual machine object, from a template

object, or from a hardware profile object.

In Virtual Machine Manager, each virtual machine, template, or hardware pro

file has one floppy drive. You cannot remove this floppy drive or add any a

dditional floppy drives.

By default, the virtual floppy drive is configured as attached to no media.

You can use the Set-VirtualFloppyDrive cmdlet to configure the virtual flo

ppy drive to use the physical floppy drive on the virtual machine host (typ

ically, drive A:) in order to read physical floppy disks. Alternatively, yo

u can configure the virtual floppy drive to read an existing virtual floppy

disk.

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get the virtual floppy drive from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> Get-VirtualFloppyDrive -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command selects from all virtual machine objects in the VMM data

base the object that represents VM01 and stores this object in variable $VM

.

The last command gets the virtual floppy drive object on VM01 and displays

information about this drive to the user.

2: Get the virtual floppy drive from a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> Get-VirtualFloppyDrive -Template $Template

The first command connects to VMMServer1.

The second command selects from all template objects in the VMM library the

object that represents Template1 and stores this object in variable $Templ

ate.

The last command gets the virtual floppy drive object on Template1 and disp

lays information about this drive to the user.

3: Get the virtual floppy drive from a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> Get-VirtualFloppyDrive -HardwareProfile $HWProfile

The first command connects to VMMServer1.

The second command selects from all hardware profile objects in the VMM lib

rary the object that represents HardwareProfile1 and stores this object in

variable $HWProfile.

The last command gets the virtual floppy drive object on HardwareProfile1 a

nd displays information about this drive to the user.

REMARKS

For more information, type: "get-help Get-VirtualFloppyDrive -detailed".

For technical information, type: "get-help Get-VirtualFloppyDrive -full".

### Set-VirtualFloppyDrive

SYNOPSIS

Changes properties of a virtual floppy drive associated with a virtual mach

ine, template, or hardware profile used in Virtual Machine Manager.

SYNTAX

Set-VirtualFloppyDrive -AnyHostDrive [-JobGroup <Guid>] [-JobVariable <Stri

ng>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String ServerCon

nection>]] [<CommonParameters>]

Set-VirtualFloppyDrive -HostDrive <String> [-JobGroup <Guid>] [-JobVariable

<String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String Ser

verConnection>]] [<CommonParameters>]

Set-VirtualFloppyDrive -NoMedia <Boolean> [-JobGroup <Guid>] [-JobVariable

<String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String Serv

erConnection>]] [<CommonParameters>]

Set-VirtualFloppyDrive [-VirtualFloppyDrive] <VirtualFloppyDrive> -VirtualF

loppyDisk <VirtualFloppyDisk> [-JobGroup <Guid>] [-JobVariable <String>] [-

PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Set-VirtualFloppyDrive -VirtualFloppyDisk <VirtualFloppyDisk> [-JobGroup <G

uid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMM

Server [<String ServerConnection>]] [<CommonParameters>]

Set-VirtualFloppyDrive [-VirtualFloppyDrive] <VirtualFloppyDrive> -AnyHostD

rive [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsy

nchronously] [<CommonParameters>]

Set-VirtualFloppyDrive [-VirtualFloppyDrive] <VirtualFloppyDrive> -HostDriv

e <String> [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-

RunAsynchronously] [<CommonParameters>]

Set-VirtualFloppyDrive [-VirtualFloppyDrive] <VirtualFloppyDrive> -NoMedia

<Boolean> [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-R

unAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual floppy drive associated with a

virtual machine, template, or hardware profile used in a Virtual Machine Ma

nager environment.

You can use the Set-VirtualFloppyDrive cmdlet to configure the virtual flop

py drive to:

\* Use a physical floppy drive (typically, drive A:) in

order to read physical floppy disks.

\* Read an existing virtual floppy disk.

\* Disconnect the virtual floppy disk.

PARAMETERS

-VirtualFloppyDrive <VirtualFloppyDrive>

Specifies a virtual floppy drive object.

-AnyHostDrive

Specifies that a virtual DVD or floppy drive will be connected to any c

orresponding physical drive on a host when you deploy a stored virtual

machine on a host, or when you use a template or hardware profile to cr

eate and deploy a virtual machine on a host.

-HostDrive <String>

Specifies a drive on a virtual machine host.

Example formats:

Hard Drive Floppy Drive

---------- ------------

Windows-based host "C:" "A:"

VMware ESX host "/dev/tools" "/dev/sda"

-NoMedia <Boolean>

Disconnects a virtual DVD drive from the host drive or ISO to which it

was connected, or disconnects a virtual floppy drive from the host driv

e or virtual floppy disk to which it was connected.

-VirtualFloppyDisk <VirtualFloppyDisk>

Specifies a virtual floppy disk object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Connect a virtual floppy drive to a physical floppy drive.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $FloppyDrive = @(Get-VirtualFloppyDrive -VM $VM)

PS C:\> trap{"Fail: Cannot set virtual floppy drive to the physical host fl

oppy drive for VM: $VM";continue} Set-VirtualFloppyDrive -VirtualFloppyDriv

e $FloppyDrive[0] -HostDrive "A:"

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command gets the object that represents the virtual floppy drive

on VM01 and stores the drive object in variable $FloppyDrive. Using the "@"

symbol and parentheses ensures that the command stores the results in an a

rray (in case the command returns a single object, which is likely for a vi

rtual floppy drive, or returns a null value).

The last command connects the virtual floppy drive object in $FloppyDrive t

o a physical drive (in this example, to the A: drive) on the host. The comm

and uses the trap statement to catch terminating exceptions. If the Set-Vir

tualFloppyDrive command fails, the string in the trap statement is displaye

d. Continue is used in the trap statement to continue execution instead of

exiting.

2: Connect a virtual floppy drive to a virtual floppy disk.

PS C:\> $FloppyDisk = Get-VirtualFloppyDisk -VMMServer VMMServer1.Contoso.c

om | where {$\_.Name -eq "BootDisk.vfd"}

PS C:\> $VM = Get-VM -Name VM02

PS C:\> $FloppyDrive = @(Get-VirtualFloppyDrive -VM $VM)

PS C:\> Set-VirtualFloppyDrive -VirtualFloppyDrive $FloppyDrive[0] -Virtual

FloppyDisk $FloppyDisk

The first command gets the virtual floppy disk named BootDisk.vfd from VMMS

erver1 and stores the virtual floppy disk object in $FloppyDisk.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command gets the object that represents the virtual floppy drive

on VM02 and stores the virtual floppy drive object in $FloppyDrive.

The last command connects BootDisk.vfd ($FloppyDisk) to the first virtual f

loppy drive (FloppyDrive[0]) on VM02.

3: Disconnect a virtual floppy drive.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> $FloppyDrive = @(Get-VirtualFloppyDrive -VM $VM)

PS C:\> Set-VirtualFloppyDrive -VirtualFloppyDrive $FloppyDrive -NoMedia

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM03 and stores the virtual machine object in $VM.

The third command gets the object that represents the virtual floppy drive

on VM03 and stores the drive object in $FloppyDrive.

The last command disconnects the virtual floppy drive whose object is store

d in $FloppyDrive from any host drive or virtual floppy disk to which it wa

s connected by specifying the NoMedia parameter. This command also deletes

any virtual floppy disk that the virtual floppy drive used earlier if no ot

her virtual machine currently uses that virtual floppy disk.

REMARKS

For more information, type: "get-help Set-VirtualFloppyDrive -detailed".

For technical information, type: "get-help Set-VirtualFloppyDrive -full".

## VirtualHardDisk

### Get-VirtualHardDisk

SYNOPSIS

Gets virtual hard disk objects from a virtual machine, from a template, or

as a standalone file stored in the Virtual Machine Manager library.

SYNTAX

Get-VirtualHardDisk [-All] [-VMMServer [<String ServerConnection>]] [<Commo

nParameters>]

Get-VirtualHardDisk -Template [<Template String>] [<CommonParameters>]

Get-VirtualHardDisk -VM [<String VM>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets objects that represent virtual hard disks from a virtual machine, from

a template, or as a standalone file stored in the Virtual Machine Manager

library.

In VMM 2008, a virtual hard disk can be a Windows-based .vhd file or a VMwa

re-based.vmdk file. A virtual hard disk might be stored as a standalone obj

ect in the Virtual Machine Manager library; attached to a virtual disk driv

e on a template; or attached to a virtual disk drive on a deployed or store

d virtual machine.

PARAMETERS

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get a virtual hard disk object from the library.

PS C:\> $VHD = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | wher

e { $\_.Name -eq “VHD01.vhd” -and $\_.LibraryServer.Name -eq "FileServer01.C

ontoso.com” }

Selects the object for the virtual hard disk named VHD01.vhd (stored on the

library server FileServer01) and stores its object in variable $VHD.

2: Get a virtual hard disk object from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VHD = Get-VM -Name "VM01" | Get-VirtualHardDisk | where { $\_.Name

-match "DataDisk" }

The first command connects to VMMServer1.

The second command gets the object for VM01, selects all virtual hard disks

on VM01 whose name includes the string "DataDisk," and stores the returned

virtual hard disk objects in $VHD.

3: Get a virtual hard disk object from a specific template.

PS C:\> Get-Template -VMMServer "VMMServer1.Contoso.com" | where {$\_.Name -

eq "Template03"} | Get-VirtualHardDisk

Gets the object that represents Template03 from the library and displays al

l virtual hard disk objects on that template to the user.

REMARKS

For more information, type: "get-help Get-VirtualHardDisk -detailed".

For technical information, type: "get-help Get-VirtualHardDisk -full".

### Move-VirtualHardDisk

SYNOPSIS

Moves a Windows-based virtual hard disk file (a .vhd file) from one locatio

n to another on the same host.

SYNTAX

Move-VirtualHardDisk -Bus <Int32> -JobGroup <Guid> -LUN <Int32> -Path <Stri

ng> -SCSI <Boolean> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchr

onously] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

Move-VirtualHardDisk -Bus <Int32> -IDE <Boolean> -JobGroup <Guid> -LUN <Int

32> -Path <String> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchro

nously] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

Move-VirtualHardDisk [-VirtualHardDisk] <VirtualHardDisk> -JobGroup <Guid>

-Path <String> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronous

ly] [<CommonParameters>]

DETAILED DESCRIPTION

Moves a Windows-based virtual hard disk file (a .vhd file) from one locatio

n to another on the same host.

For example:

\* If the host has multiple physical disk drives and the virtual machine

has two virtual hard disks (one might store the operating system and

the other might contain data), you can use this cmdlet to put one of

the virtual hard disks onto a different physical hard drive in order

to improve performance for both virtual hard disks.

\* If the virtual machine has one dynamically expanding virtual hard disk

and you discover that the virtual hard disk has grown so large that it

uses most of the space on its current physical hard disk on the host,

you can use this cmdlet to move the expanded virtual hard disk to a

larger physical hard disk if a larger physical hard disk is available

on the host.

The Move-VirtualHardDisk cmdlet moves a Windows-based .vhd file used by a v

irtual machine to another location on a Windows-based host (a Hyper-V or Vi

rtual Server host). You cannot use this cmdlet to move a VMware-based .vmdk

file to another location on an ESX host.

PARAMETERS

-VirtualHardDisk <VirtualHardDisk>

Specifies a virtual hard disk object.

-Bus <Int32>

Specifies the IDE bus to which to attach a virtual disk drive or virtua

l DVD drive, or the SCSI bus to which to attach a virtual disk drive.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter. For more information about how VMM 2008 impleme

nts IDE and SCSI buses, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enha

ncements.

-IDE <Boolean>

Specifies IDE as the bus type to which to attach a virtual disk drive o

bject or a virtual DVD drive object configured on a virtual machine or

on a template. (For more information about how VMM 2008 implements the

IDE bus, type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enhancements.)

Example format: -IDE –Bus 0 –LUN 1

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-LUN <Int32>

Specifies the logical unit number (LUN) for a virtual disk drive object

or for a virtual DVD drive object on an IDE bus, or for a virtual disk

drive object on a SCSI bus.

Example format: -IDE -Bus 1 -LUN 0

Example format: -SCSI -Bus 0 -LUN 1

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-SCSI <Boolean>

Specifies SCSI as the bus type to which to attach a virtual disk drive

object configured on a virtual machine or on a template.

Example format: -SCSI -Bus 0 -LUN 0

For information about the number of devices per controller on a SCSI bu

s that VMM 2008 supports for each virtualization platform (Hyper-V, Vir

tual Server, or VMware), type: Get-Help about\_VMM\_2008\_Disk\_and\_DVD\_Enh

ancements.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Move a virtual hard disk file from one location to another on the same h

ost.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $VHD = $VM.VirtualHardDisks[0]

PS C:\> Move-VirtualHardDisk -VirtualHardDisk $VHD -Path "C:\Test"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command stores the object for the first virtual hard disk on VM01

($VM.VirtualHardDisks[0]) in $VHD.

The last command (which assumes that a directory named "Test" already exist

s on volume "C:\") moves the virtual hard disk represented by $VHD to the f

older "C:\Test".

REMARKS

For more information, type: "get-help Move-VirtualHardDisk -detailed".

For technical information, type: "get-help Move-VirtualHardDisk -full".

### Remove-VirtualHardDisk

SYNOPSIS

Removes a virtual hard disk object from a virtual machine or template or fr

om the Virtual Machine Manager library.

SYNTAX

Remove-VirtualHardDisk [-VirtualHardDisk] <VirtualHardDisk> [-Confirm] [-Fo

rce] [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsy

nchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents a virtual hard disk from a virtual machin

e or template or from the Virtual Machine Manager library. The Remove-Virtu

alHardDisk cmdlet also deletes the corresponding virtual hard disk file (a

Windows-based .vhd file or a VMware-based .vmdk file) on the library server

.

If the virtual hard disk is attached to a virtual disk drive on a virtual m

achine or template (and if you do not use the Force parameter), VMM lists t

he container that contains the virtual hard disk and prompts you to confirm

that you want to remove the virtual hard disk:

\* If you reply Yes, VMM removes the association between the virtual hard

disk and the container to which it is attached, and then deletes the

virtual hard disk object from VMM.

\* If you reply No, the operation is cancelled.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualHardDisk <VirtualHardDisk>

Specifies a virtual hard disk object.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a virtual hard disk object from the library.

PS C:\> $VHD = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | wher

e { $\_.Name -eq “VHD01.vhd” -and $\_.LibraryServer.Name -eq "FileServer01.C

ontoso.com” }

PS C:\> Remove-VirtualHardDisk -VirtualHardDisk $VHD

The first command gets the virtual hard disk object for the file VHD01.vhd

(stored on library server FileServer01) and stores the returned object in v

ariable $VHD.

The second command removes the VHD01 object from the library and deletes th

e corresponding file (a .vhd file or a .vmdk file) from the file system on

the library server.

2: Remove a virtual hard disk from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VHD = Get-VM -Name "VM01" | Get-VirtualHardDisk | where { $\_.Name

-match "DataDisk" }

PS C:\> $VHD | Remove-VirtualHardDisk

The first command connects to VMMServer1.

The second command gets the object for the virtual machine named VM01, gets

all virtual hard disks on VM01 whose name includes the string "DataDisk",

and stores these virtual hard disk objects in $VHD (an object array).

The third command removes each virtual hard disk object from the virtual ma

chine and deletes each corresponding file (a .vhd file or a .vmdk file) fro

m the file system on the library server.

REMARKS

For more information, type: "get-help Remove-VirtualHardDisk -detailed".

For technical information, type: "get-help Remove-VirtualHardDisk -full".

### Set-VirtualHardDisk

SYNOPSIS

Changes properties of a virtual hard disk object used in Virtual Machine Ma

nager.

SYNTAX

Set-VirtualHardDisk [-VirtualHardDisk] <VirtualHardDisk> [-Description <Str

ing>] [-Enabled <Boolean>] [-JobGroup <Guid>] [-JobVariable <String>] [-Nam

e <String>] [-OperatingSystem <OperatingSystem>] [-Owner <String>] [-PROTip

ID <Guid>] [-RunAsynchronously] [-SharePath <String>] [-VirtualizationPlatf

orm <VirtualizationPlatform>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual hard disk object used in a Virt

ual Machine Manager environment. A virtual hard disk file used in VMM 2008

is either a Windows-based .vhd file or a VMware-based .vmdk file.

Properties that you can change include:

- Description

- Enabled

- Name

- Operating System

- Owner

- SharePath

A virtual hard disk file might be stored in the VMM library, or it might be

attached to a virtual disk drive on a virtual machine or template.

NOTE: In VMM 2007, the Set-VirtualHardDisk cmdlet was used to change the lo

cation of a virtual hard disk on a virtual machine by changing the Bus and

LUN settings on an IDE bus or SCSI bus. By contrast, in VMM 2008, a virtual

hard disk is attached to a virtual disk drive. If you want to change the B

us and LUN settings for a virtual disk drive, use the Set-VirtualDiskDrive

cmdlet.

PARAMETERS

-VirtualHardDisk <VirtualHardDisk>

Specifies a virtual hard disk object.

-Description <String>

Specifies a description for the specified object.

-Enabled <Boolean>

Enables a library object (when set to TRUE) or disables a library objec

t (when set to FALSE). For example, if you want to upgrade software on

a virtual machine template, you can disable the template object in the

VMM library to temporarily prevent users from using that object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

-VirtualizationPlatform <VirtualizationPlatform>

Specifies the virtualization platform of a virtual machine host managed

by VMM.

Valid values: VirtualServer, Hyper-V, or VMwareESX.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the description of a virtual hard disk.

PS C:\> $VHD = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | wher

e { $\_.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -eq "

WindowsServer2003BootVHD.vhd"}

PS C:\> Set-VirtualHardDisk -VirtualHardDisk $VHD -Description “Latest Wind

ows Server 2003 Boot VHD”

The first command retrieves from the library on VMMServer1 the object that

represents the virtual hard disk named "WindowsServer2003BootVHD.vhd", whos

e file is stored on library server FileServer01. The command stores the vir

tual hard disk object in variable $VHD.

The second command changes the description of this virtual hard disk object

to "Latest Windows Server 2003 Boot VHD".

2: Enable a VMware-based virtual hard disk in the library.

PS C:\> $VMDK = Get-VirtualHardDisk -VMMServer VMMServer1.Contoso.com | whe

re { $\_.LibraryServer.Name -eq “FileServer01.Contoso.com” -and $\_.Name -eq

"WindowsServer2003.vmdk"}

PS C:\> Set-VirtualHardDisk -VirtualHardDisk $VMDK -Enabled $TRUE

The first command gets the object that represents the virtual hard disk nam

ed WindowsServer2003.vmdk (whose file is stored on library server FileServe

r01) and stores the virtual hard disk object in $VMDK.

The second command enables the virtual hard disk object.

3: Specify an owner for all virtual hard disks with an "Unknown" owner.

PS C:\> Get-VirtualHardDisk -VMMServer "VMMServer1.Contoso.com" | where {$\_

.Owner -eq "Unknown"} | Set-VirtualHardDisk -Owner "Contoso\SanjayPatel"

Gets all virtual hard disk objects from the VMM library, selects only those

virtual hard disks whose owner is "Unknown," and specifies an owner for ea

ch virtual hard disk object.

REMARKS

For more information, type: "get-help Set-VirtualHardDisk -detailed".

For technical information, type: "get-help Set-VirtualHardDisk -full".

## VirtualizationManager

### Add-VirtualizationManager

SYNOPSIS

Adds a VMware VirtualCenter Server to Virtual Machine Manager.

SYNTAX

Add-VirtualizationManager [-ComputerName] <String> -Credential <PSCredentia

l> [-Certificate] [-Description <String>] [-JobVariable <String>] [-PROTipI

D <Guid>] [-RunAsynchronously] [-SecureMode <Boolean>] [-TCPPort <Int32>] [

-VMMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Adds a VMware VirtualCenter Management Server, also called a VirtualCenter

Server, to your Virtual Machine Manager environment so that Virtual Machine

Manager can connect to the VirtualCenter Server and import its data. After

you add the VirtualCenter Server to Virtual Machine Manager, Virtual Machi

ne Manager can manage VMware ESX hosts associated with the VirtualCenter Se

rver and the virtual machines deployed on those hosts.

Adding a VirtualCenter Server as a managed platform to Virtual Machine Mana

ger maps each VMware object to a corresponding Virtual Machine Manager obje

ct. This mapping enables that data to be managed as Windows PowerShell obje

cts from the command-line or presented in the same Administrator Console as

Microsoft virtual objects (in the Administration view).

The default port used to connect to a VMware VirtualCenter Server computer

is TCP port 443.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-Certificate

Specifies a security certificate object.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SecureMode <Boolean>

Specifies the trust level between VMM and VMware ESX hosts managed by a

VMware VirtualCenter Server.

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a VMware VirtualCenter Server to VMM.

PS C:\> $Credential = Get-Credential

PS C:\> $Cert= Get-Certificate -Computername "VirtMgrServer01.Contoso.com"

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Add-VirtualizationManager -ComputerName "VirtMgrServer01.Contoso.co

m" -Certificate $Cert -TCPPort 443 -Credential $Credential

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are either a local Administrator account

or a domain account with administrator rights on the computer (a VMware Vir

tualCenter Server) that you want to add as a virtualization manager to VMM.

The second command obtains the security certificate from "VirtMgrServer01.C

ontoso.com" and stores it in a variable $Cert.

The third command connects to the VMM server named VMMServer1 that is locat

ed in the Contoso domain and gets the server object from the VMM database p

rovided by VMMServer1.

The last command adds the virtualization manager object that represents the

VirtualCenter Server called VirtMgrServer01 to the VMM database, imports t

he security certificate object , and specifies that VMM will use TCP port 4

43 (the default port) to connect to that server. As the last command is pro

cessed, $Credential provides your credentials to Add-VirtualizationManager.

2: Add multiple VMware VirtualCenter Servers to VMM.

PS C:\> $Credential = Get-Credential

PS C:\> $ServerNames = "VirtMgrServer01.Contoso.com", "VirtMgrServer02.Cont

oso.com"

PS C:\> foreach( $ServerName in $ServerNames ) {$Cert = Get-Certificate -Co

mputername $Servername; Add-VirtualizationManager -ComputerName $ServerNam

e -Certificate $Cert -TCPPort 443 -Credential $Credential}

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password and stores your credentials in $Credential.

The second command stores in $ServerNames the strings "VirtMgrServer01.Cont

oso.com" and "VirtMgrServer02.Contoso.com", which are the names of two VMwa

re VirtualCenter Servers.

The last command adds the two servers to VMM and specifies that VMM will im

port the security certificates and use TCP port 443 (the default port) to

connect to the virtualization manager service on VirtMgrServer01 and VirtM

grServer02. As this command is processed, $Credential provides your credent

ials to Add-VirtualizationManager.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

REMARKS

For more information, type: "get-help Add-VirtualizationManager -detailed".

For technical information, type: "get-help Add-VirtualizationManager -full"

.

### Get-VirtualizationManager

SYNOPSIS

Gets objects that represent VMware VirtualCenter Servers managed by Virtual

Machine Manager from the Virtual Machine Manager database.

SYNTAX

Get-VirtualizationManager [[-ComputerName] <String>] [-VMMServer [<String S

erverConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent VMware VirtualCenter Servers manage

d by Virtual Machine Manager from the Virtual Machine Manager database. Vir

tualCenter Server is a virtualization manager that typically manages VMware

ESX Server virtual machine hosts and virtual machines deployed on those ho

sts.

If a VirtualCenter Server is connected to Virtual Machine Manager, you can

use this cmdlet to view the object properties of the VirtualCenter Server o

r to store its object in a variable for use by other cmdlets.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Display information about each VMware VirtualCenter Server managed by VM

M.

PS C:\> Get-VirtualizationManager -VMMServer VMMServer1.Contoso.com

The command retrieves all objects that represent virtualization managers cu

rrently associated with VMM from VMMServer1 and displays information about

the returned objects.

2: Get a specific VMware VirtualCenter Server managed by VMM.

PS C:\> Get-VirtualizatonManager -VMMServer VMMServer1.Contoso.com -Compute

rName "VirtMgrServer02"

Gets object for the VirtualCenter server called "VirtMgrServer02" and displ

ays information about the returned object.

3: Get all VMware VirtualCenter Servers that match specified criteria.

PS C:\> $VirtMgrServers = Get-VirtualizationManager -VMMServer "VMMServer1.

Contoso.com" | where {$\_.Name -match "Server"}

PS C:\> Write-Output $VirtMgrServers

The first command gets all virtualization manager objects whose name includ

es the string “Server” (such as VirtMgrServer01, VirtMgrServer02, and so on

) and stores the returned objects in $VirtMgrServers.

The second command displays information about each VirtualCenter Server obj

ect.

REMARKS

For more information, type: "get-help Get-VirtualizationManager -detailed".

For technical information, type: "get-help Get-VirtualizationManager -full"

.

### Refresh-VirtualizationManager

SYNOPSIS

Refreshes the properties of a VMware VirtualCenter Server so that the Virtu

al Machine Manager Administrator Console displays updated information about

entities in VirtualCenter Server.

SYNTAX

Refresh-VirtualizationManager [-VirtualizationManager] <VirtualizationManag

er> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<Commo

nParameters>]

DETAILED DESCRIPTION

Refreshes the properties of a VMware VirtualCenter Server managed by Virtua

l Machine Manager so that the Virtual Machine Manager Administrator Console

displays updated information about entities in VirtualCenter Server.

The Refresh-VirtualizationManager cmdlet imports changes that an administra

tor makes on VirtualCenter Server into the Virtual Machine Manager database

so that Virtual Machine Manager can then manage the new or modified object

s.

For example, if an administrator adds a set of ESX hosts to VirtualCenter S

erver and then, on the Virtual Machine Manager server, runs the Refresh-Vir

tualizationManager cmdlet, Virtual Machine Manager discovers the new hosts

(and the virtual machines deployed on those hosts). These ESX hosts and the

ir virtual machines can now be managed by Virtual Machine Manager.

Similarly, if an administrator changes the hierarchical structure of folder

and datacenter objects (which are child objects under the VirtualCenter ro

ot folder), running the Refresh-VirtualizationManager cmdlet propagates tho

se changes into Virtual Machine Manager.

PARAMETERS

-VirtualizationManager <VirtualizationManager>

Specifies a virtualization manager object currently managed by VMM. VMM

2008 supports the following non-Microsoft virtualization managers:

-VMware VirtualCenter 2.0: manages hosts running ESX Server 3.0 or 3.5

-VMware VirtualCenter 2.5: manages hosts running ESX Server 3.0, 3.5, o

r 3i

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Refresh all VMware VirtualCenter Servers managed by VMM.

PS C:\> $VirtManagers = Get-VirtualizationManager -VMMServer VMMServer1.Con

toso.com

PS C:\> foreach ($VirtMan in $VirtManagers) { Refresh-VirtualizationManager

-VirtualizationManager $VirtMan }

The first command retrieves all virtualizaton manager objects from the VMM

database on VMMServer1 and stores the returned objects in variable $VirtMan

.

The second command uses a foreach loop statement to refreshe the properties

of the objects stored in $VirtMan so that current information about these

virtualization managers displays in the Administrator Console.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

2: Refresh a specific VMware VirtualCenter Server managed by VMM.

PS C:\> $VirtMan = Get-VirtualizationManager -VMMServer "VMMServer1.Contoso

.com" -Computername "VMwareVC1"

PS C:\> Refresh-VirtualizationManager -VirtualizationManager $VirtMan

The first command gets the object that represents a specific virtualizaton

manager named "VMwareVC1" from VMMServer1 and stores the returned object in

$VirtMan.

The second command refreshes the properties for the object stored in $VirtM

an so that current information about this virtualization manager displays i

n the Administrator Console.

REMARKS

For more information, type: "get-help Refresh-VirtualizationManager -detail

ed".

For technical information, type: "get-help Refresh-VirtualizationManager -f

ull".

### Remove-VirtualizationManager

SYNOPSIS

Removes a VMware VirtualCenter Server from Virtual Machine Manager.

SYNTAX

Remove-VirtualizationManager [-VirtualizationManager] <VirtualizationManage

r> [-Confirm] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronousl

y] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent VMware VirtualCenter Servers fro

m the Virtual Machine Manager database. This cmdlet deletes the VirtualCent

er Server object from the Virtual Machine Manager database and also removes

all imported ESX host objects and virtual machine objects associated with

the VirtualCenter Server from the Virtual Machine Manager database.

When you remove a VirtualCenter Server, the cmdlet does not make any change

s within VirtualCenter Server and does not remove any hosts or virtual mach

ines from the VirtualCenter Server. What has changed after you run the Remo

ve-VirtualizationManager cmdlet is that the VirtualCenter server is no long

er managed by Virtual Machine Manager.

PARAMETERS

-VirtualizationManager <VirtualizationManager>

Specifies a virtualization manager object currently managed by VMM. VMM

2008 supports the following non-Microsoft virtualization managers:

-VMware VirtualCenter 2.0: manages hosts running ESX Server 3.0 or 3.5

-VMware VirtualCenter 2.5: manages hosts running ESX Server 3.0, 3.5, o

r 3i

-Confirm

Prompts for confirmation before running the command.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a VMware VirtualCenter Server from VMM.

PS C:\> $VirtMgrServer = Get-VirtualizationManager -VMMServer VMMServer1.Co

ntoso.com -ComputerName "VirtMgrServer01.Contoso.com"

PS C:\> Remove-VirtualizationManager -VirtualizationManager $VirtMgrServer

The first command gets the object for a VirtualCenter Server named VirtMgrS

erver01 from VMMServer1 and stores the object in $VirtMgrServer.

The second command removes the VirtualCenter Server object, as well as all

associated host and virtual machine objects, from VMM.

NOTE: This example assumes that there is only one VirtualCenter Server.

2: Remove a set of VMware VirtualCenter Servers from VMM.

PS C:\> $VirtMgrServers = Get-VirtualizationManager -VMMServer VMMServer1.C

ontoso.com | where { $\_.Name -match "Server" }

PS C:\> foreach( $VirtMgr in $VirtMgrServers) { Remove-VirtualizationManage

r -VirtualizationManager $VirtMgr }

The first command gets all virtualization manager objects whose name includ

es the string “Server” and stores the returned objects in $VirtMgrServers.

The second command removes each object in $VirtMgrServers from VMM, as well

as all associated host and virtual machine objects.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

3: Remove all VMware VirtualCenter Servers from VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VirtualizationManager | foreach { Remove-VirtualizationManager

-VirtualizationManager $\_ -RunAsynchronously }

The first command connects to VMMServer1.

The second command removes all virtualization manager objects from VMM.

REMARKS

For more information, type: "get-help Remove-VirtualizationManager -detaile

d".

For technical information, type: "get-help Remove-VirtualizationManager -fu

ll".

### Set-VirtualizationManager

SYNOPSIS

Changes properties of a VMware VirtualCenter Server that is managed by Virt

ual Machine Manager.

SYNTAX

Set-VirtualizationManager [-VirtualizationManager] <VirtualizationManager>

[-Certificate] [-Credential <PSCredential>] [-JobVariable <String>] [-PROTi

pID <Guid>] [-RunAsynchronously] [-SecureMode <Boolean>] [-TCPPort <Int32>]

[<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a VMware VirtualCenter Server that is con

nected to and managed by Virtual Machine Manager. A VirtualCenter Server ma

nages VMware ESX Servers that act as hosts for VMware-based virtual machine

s.

Properties that you can change include settings for the TCP port used to co

nnect to the VirtualCenter Server, credentials used to access the VirtualCe

nter Server, and updating a VMware VirtualCenter Server security certificat

e.

If a security certificate for a VMware VirtualCenter server expires or a se

lf-signed certificate is replaced by a certificate from a third-party Certi

fication Authority (CA), you must update both VMware VirtualCenter and VMM:

\* First, replace the current VMware VirtualCenter certificate with the

new certificate in VMware VirtualCenter. See the instructions in the

VMware "Developer's Setup Guide," available at:

http://www.vmware.com/support/developer/vc-sdk/

\* Next, update the certificate in VMM by importing the new certificate

into VMM. See example 3 for this cmdlet.

For more information about including a VMware VirtualCenter Server as a vir

tualization managers in a Virtual Machine Manager environment, type:

Get-Help Add-VirtualizationManager -detailed

PARAMETERS

-VirtualizationManager <VirtualizationManager>

Specifies a virtualization manager object currently managed by VMM. VMM

2008 supports the following non-Microsoft virtualization managers:

-VMware VirtualCenter 2.0: manages hosts running ESX Server 3.0 or 3.5

-VMware VirtualCenter 2.5: manages hosts running ESX Server 3.0, 3.5, o

r 3i

-Certificate

Specifies a security certificate object.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SecureMode <Boolean>

Specifies the trust level between VMM and VMware ESX hosts managed by a

VMware VirtualCenter Server.

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Specify new credentials for a virtualization manager object.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VirtMgr = Get-VirtualizationManager -ComputerName "VirtMgrServer01

"

PS C:\> $Credential = Get-Credential

PS C:\> Set-VirtualizationManager -VirtualizationManager $VirtMgr -Credenti

al $Credential

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object for the VMware VirtualCenter Server virt

ualization manager named VirtMgrServer01 from the VMM database and stores t

he returned object in variable $VirtMgr.

The third command prompts the user for new credentials and stores the crede

ntials in variable $Credential. The credentials required for this operation

are <VCServerName>\Administrator and the password for that account.

The last command changes the credentials stored for VirtMgrServer01 to be u

sed when VMM connects to the external service.

2: Modify the virtualization manager TCP port settings.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VirtMgr = Get-VirtualizationManager -ComputerName "VirtMgrServer02

"

PS C:\> Set-VirtualizationManager -VirtualizationManager $VirtMgr -TCPPort

980

The first command connects to VMMServer1.

The second command gets the object for the VMware VirtualCenter Server virt

ualization manager named VirtMgrServer02 and stores the returned object in

$VirtMgr.

The third command changes the TCP port setting used to connect to the Virtu

alCenter Server to "980"

3: Update the VirtualCenter Server security certificate by importing a new

certificate into VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\>$Virtmgr = Get-Virtualizationmanager -Computername “VirtMgrServer03.

Contoso.com”

PS C:\>$Cert = Get-Certificate -Computername “VirtMgrServer01.Contoso.com”

PS C:\>Set-VirtualizationManager -VirtualizationManager $VirtMgr -Certifica

te $Cert

The first command connects to VMMServer1.

The second command gets the object that represents the VMWare VirtualCenter

Server named VirtMgrServer03 (located in the Contoso.com domain) and store

s the returned object in $VirtMgr.

The third command obtains the security certficate object from VirtMgrServer

01 and stores the certificate object in $Cert.

The last command imports the security certificate object on VirtMgrServer01

into the local machine Certificate store on the VMM server.

The VMware certificate is located in the local machine Certificate store un

der Certificates (Local Computer) / Trusted People / Certificates. For info

rmation about installing and using MMC Certificates, see "How to: View Cert

ificates with the MMC Snap-in" (http://go.microsoft.com/fwlink/?LinkID=9436

9).

REMARKS

For more information, type: "get-help Set-VirtualizationManager -detailed".

For technical information, type: "get-help Set-VirtualizationManager -full"

.

## VirtualNetwork

### Get-VirtualNetwork

SYNOPSIS

Gets virtual network objects configured on a host managed by Virtual Machin

e Manager.

SYNTAX

Get-VirtualNetwork [[-Name] <String>] [-VMMServer [<String ServerConnection

>]] [<CommonParameters>]

Get-VirtualNetwork [[-Name] <String>] -VMHost [<String Host>] [<CommonParam

eters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual networks configured on a ho

st managed by Virtual Machine Manager.

For more information about virtual networks in Virtual Machine Manager, typ

e:

Get-Help New-VirtualNetwork -detailed

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all virtual networks in the VMM database.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VN = Get-VirtualNetwork

PS C:\> $VN

PS C:\> $VN | select Name,VMHost,VMHostNetworkadapters | format-list

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command all objects that represent virtual networks on all hosts

managed by VMM and stores the virtual network objects in $VN (an object ar

ray).

The third command displays a set of information about each virtual network

object in $VN.

The last command displays a subset of the information in $VN: the name of e

ach virtual network, the physical host on which each virtual network is con

figured, and the physical network adapters configured on the host for each

virtual network.

2: Get all virtual networks on a specific host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> $VN = Get-VirtualNetwork -VMHost $VMHost

PS C:\> $VN

The first command connects to VMMServer1.

The second command gets the object that represents VMHost02 and stores the

host object in $VMHost.

The third command gets from VMHost01 all objects that represent virtual net

works on this host and stores the virtual network objects in $VN.

The last command displays information about each virtual network on VMHost0

2.

3: Get a virtual network by name from a specific host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03.Contoso.com"

PS C:\> $VN = Get-VirtualNetwork -VMHost $VMHost -Name "Internal Network"

PS C:\> $VN | Get-Member

The first command connects to VMMServer1.

The second command gets the object that represents VMHost03 and stores the

host object in $VMHost.

The third command gets from VMHost03 the object that represents the virtual

network named "Internal Network" and stores the virtual network object in

$VN.

The last command passes each virtual network object in $VN to the Get-Membe

r cmdlet, which displays the .NET type for a virtual network object:

TypeName: Microsoft.SystemCenter.VirtualMachineManager.VirtualNetwork

The command also displays a list of methods and properties that are associa

ted with a VMM virtual network object.

REMARKS

For more information, type: "get-help Get-VirtualNetwork -detailed".

For technical information, type: "get-help Get-VirtualNetwork -full".

### New-VirtualNetwork

SYNOPSIS

Creates a virtual network object on a host managed by Virtual Machine Manag

er that enables virtual machines on that host to communicate over that virt

ual network.

SYNTAX

New-VirtualNetwork [-Name] <String> -VMHost [<String Host>] [-BoundToVMHost

<Boolean>] [-Description <String>] [-HostBoundVLanId] [-JobGroup <Guid>] [

-JobVariable <String>] [-NetworkTag <String>] [-Path <String>] [-PROTipID <

Guid>] [-RunAsynchronously] [-VirtualDHCPDefaultGatewayAddress <String>] [-

VirtualDHCPDNSServer <String>] [-VirtualDHCPEnabled <Boolean>] [-VirtualDHC

PEndingIPAddress <String>] [-VirtualDHCPIPAddressLeaseTime <Int32>] [-Virtu

alDHCPLeaseRebindingTime <Int32>] [-VirtualDHCPLeaseRenewalTime <Int32>] [-

VirtualDHCPNetworkAddress <String>] [-VirtualDHCPNetworkMask <String>] [-Vi

rtualDHCPServerAddress <String>] [-VirtualDHCPStartingIPAddress <String>] [

-VirtualDHCPWINSServer <String>] [-VMHostNetworkAdapters <HostNetworkAdapte

r[]>] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual network object on a host managed by Virtual Machine Manag

er that enables virtual machines on that host to communicate over that virt

ual network.

VMM 2008 supports the use of virtual switches to implement virtual networki

ng scenarios for all supported hosts: Virtual Server, Hyper-V, and VMware h

osts. These virtual switch-based scenarios are similar to the virtual netwo

rk scenarios supported in VMM 2007. You can connect, or bind, virtual machi

nes and hosts to a virtual network switch in a manner similar to the way th

at you connect physical computers to a physical network switch.

For Hyper-V hosts and VMware hosts, and the virtual machines deployed on th

ose hosts, VMM 2008 also supports the use of virtual switches to implement

virtual local area networks (VLANs). A VLAN is an independent logical virtu

al network configured within a physical LAN. If you create multiple VLANs o

n a physical LAN, these separate logical segments cannot exchange data with

each other.

In VMM 2008, you can easily move a virtual machine that is connected to a V

LAN from one host to another host and (assuming that both hosts are connect

ed to the same VLAN), the virtual machine in its new location is already co

nfigured to resume communicating over the VLAN without any additional admin

istrator effort. Moving a virtual machine to a new location on a VLAN does

not require software reconfiguration in the way that moving a physical comp

uter to a new location on a physical network requires hardware reconfigurat

ion.

VMM 2008 NETWORKING SCENARIOS

-----------------------------

The following three scenarios summarize VMM 2008 virtual networking configu

rations.

Scenario 1 – External Virtual Network

In this scenario, virtual machines deployed on a host use a virtual network

adapter to connect to a virtual switch on the host, and this virtual switc

h is, in turn, connected to a physical network adapter on the host. The hos

t is connected through a physical switch to other computers on its network.

This configuration gives the virtual machines access to the host itself, t

o the physical network to which the host is connected, and to other physica

l computers (or other physical devices) that are on the same physical netwo

rk as the host.

The virtual network can support external access though a VLAN if the physic

al adapter on the host that it is bound to has been configured appropriatel

y and if the virtual machines on that host are configured to use a VLAN. Fo

r more information, type:

Get-Help Add-VMHostNetworkAdapter -detailed

Get-Help New-VirtualNetworkAdapter -detailed

Scenario 2 – Internal Virtual Network

In this scenario, virtual machines deployed on a host use a virtual network

adapter to connect to a virtual switch on the host. In this scenario, the

virtual network is bound to the host but the virtual machines do not connec

t via the virtual switch to a physical network adapter on the host. This co

nfiguration establishes an internal virtual network that enables virtual ma

chines connected to that virtual switch to communicate with each other and

with services and applications on the host, but not with other computers co

nnected to the host’s physical network.

If you want to to configure an internal network that is separated into two

or more VLANs, you must set the VLAN IDs on a virtual network adapter confi

gured on the virtual machine object. For more information, type:

Get-Help New-VirtualNetworkAdapter -detailed

Get-Help Set-VirtualNetworkAdapter -detailed

Scenario 3 - Private Virtual Network

In this scenario, virtual machines deployed on a host use a virtual network

adapter to connect to a virtual switch on the host. As in scenario 2, a vi

rtual machine does not connect via that virtual switch to a physical networ

k adapter on the host. Unlike scenario 2, the virtual network is not bound

to the host. This configuration establishes a private virtual network that

virtual machines on the same host can use to communicate with each other, b

ut, in this case, they cannot communicate with services or applications on

the host nor with any physical computers connected to the host’s physical n

etwork.

FOR VIRTUAL SERVER HOSTS ONLY - OPTIONAL VIRTUAL DHCP SERVER

------------------------------------------------------------

The only type of host for which you can, optionally, configure a virtual DH

CP server is a Virtual Server host. By default, a new virtual network confi

gured on a Virtual Server host does not use a virtual DHCP server.

For virtual machines configured to communicate with the host’s external net

work, if the Virtual Server host computer is configured to obtain a dynamic

IP address from a DHCP server, the virtual machines will also automaticall

y obtain dynamic IP addresses from that DHCP server (alternatively, you can

configure static IP addresses on virtual machines).

For virtual machines on an isolated internal virtual network on a Virtual S

erver host, you might choose to use static IP addresses (if you have only a

few virtual machines on the internal virtual network). More typically, you

might prefer to enable virtual DHCP to provide dynamic IP addresses to the

virtual machines on that network.

You might, for example, specify:

- 10.237.0.0 as the network address (with a subnet mask 255.255.0.0)

- 10.237.0.1 as the IP address for the DCHP server

- 10.237.0.16 as the starting IP address for DHCP clients

- 10.237.255.254 as the ending IP address for DHCP clients

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-BoundToVMHost <Boolean>

Enables virtual machines connected to a virtual network on a Hyper-V ho

st to access the host operating system. This parameter does not apply t

o Virtual Server or VMware ESX hosts.

-Description <String>

Specifies a description for the specified object.

-HostBoundVLanId

Assigns a numerical identifier in the range 1-4094 to a virtual local a

rea network (a VLAN) configured on a Hyper-V host. This is the VLAN ID

that virtual machines use to access the host. This parameter does not a

pply to Virtual Server or VMware ESX hosts.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-NetworkTag <String>

Specifies a word or phrase to associate with a virtual network adapter

that is configured to connect to a specific internal or external networ

k on the host. The NetworkTag identifies all VMs with the same NetworkT

ag as members of the same network. VMM uses a NeworkTag (if one exists)

when it evaluates hosts as possible candidates on which to deploy a VM

. If the host does not include VMs on the network with the same Network

Tag as the VM to be placed, the host receives zero stars in the placeme

nt process.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VirtualDHCPDefaultGatewayAddress <String>

Specifies the IP address of the default DHCP gateway for a virtual netw

ork configured on a Virtual Server host. The default gateway is a local

IP router that forwards traffic beyond this virtual network. This para

meter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPDNSServer <String>

Specifies a set of IP addresses of WINS servers (as strings separated b

y the pipeline operator) for use by DHCP clients for NetBIOS name resol

ution on a virtual network configured on a Virtual Server host. This pa

rameter does not apply to Hyper-V hosts or VMware ESX hosts.

Example format:

-VirtualDHCPDNSServer "nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn"

-VirtualDHCPEnabled <Boolean>

Enables virtual DHCP on a virtual network on a Virtual Server host (whe

n set to TRUE), or disables virtual DHCP on the virtual network (when s

et to FALSE). This parameter does not apply to Hyper-V hosts or VMware

ESX hosts.

-VirtualDHCPEndingIPAddress <String>

Specifies the last IP address in the range of IP addresses that the vir

tual DHCP server maintains for assignment to virtual DHCP clients on a

virtual network configured on a Virtual Server host. This parameter doe

s not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPIPAddressLeaseTime <Int32>

Specifies, in minutes, the duration of the lease for IP addresses assig

ned to virtual DHCP clients by the virtual DHCP server on a virtual net

work configured on a Virtual Server host.

Default value: 2160 minutes (36 hours). Maximum value: 49,710 days.

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPLeaseRebindingTime <Int32>

Specifies, in minutes, the DHCP lease rebinding time used by virtual DH

CP clients of a virtual DHCP server on a virtual network configured on

a Virtual Server host.

Default value: 1620 minutes (27 hours). Maximum value: 49,710 days.

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPLeaseRenewalTime <Int32>

Specifies, in minutes, the amount of time that you want a virtual DHCP

client to wait before it tries to renew its current IP address with the

virtual DHCP server on a virtual network configured on a Virtual Serve

r host.

Default value: 1080 minutes (18 hours). Maximum value: up to 49,710 day

s.

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPNetworkAddress <String>

Specifies the network address for a virtual network configured on a Vir

tual Server host. The value specified must be a valid IP address and mu

st be the base of the subnet. This parameter does not apply to Hyper-V

hosts or VMware ESX hosts.

-VirtualDHCPNetworkMask <String>

Specifies the network mask for a virtual network configured on a Virtua

l Server host. The specified value must be a valid network mask. The de

fault value is 255.255.0.0. This parameter does not apply to Hyper-V ho

sts or VMware ESX hosts.

-VirtualDHCPServerAddress <String>

Specifies the IP address for a virtual DHCP server on a virtual network

configured on a Virtual Server host. This parameter does not apply to

Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPStartingIPAddress <String>

Specifies the first IP address in the range of IP addresses that a virt

ual DHCP server maintains for assignment to virtual DHCP clients on a v

irtual network configured on a Virtual Server host. The value specified

must be a valid IP address; must use the network address (IP address o

f this virtual network) and network mask (subnet mask for this virtual

network); and must start at 16 or higher. This parameter does not apply

to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPWINSServer <String>

Specifies a set of the IP addresses of WINS servers (as strings separat

ed by the pipeline operator) for use by DHCP clients on a virtual netwo

rk configured on a Virtual Server host. The array must contain a set of

valid IP addresses.

Example format:

-VirtualDHCPWINSServer "nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn

"

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VMHostNetworkAdapters <HostNetworkAdapter[]>

Specifies an array of one or more physical network adapter objects on a

host to which virtual machines deployed on that host can connect.

Example format: -VMHostNetworkAdapters $VMHostNICs

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create an external virtual network on a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> $HostAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost -Name "Host

LANAdapter01"

PS C:\> New-VirtualNetwork -Name "ExternalVirtualNetwork1" -VMHost $VMHost

-VMHostNetworkAdapter $HostAdapter

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from teh VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host named VMHost01

and stores the host object in $VMHost.

The third command gets the object that represents the physical host network

adapter on VMHost01 named HostLANAdapter01 and stores the adapter object i

n $HostAdapter.

The last command creates a virtual network on VMHost01, names it ExternalVi

rtualNetwork1, and connects the new virtual network to the host network ada

pter HostLANAdapter01.

This virtual network is an external virtual network. It is attached to the

physical network adapter on the host and can therefore access the LAN that

the host is attached to just as if it were another physical computer on tha

t LAN.

2: Create an internal host-bound virtual network.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> $NewVNet = New-VirtualNetwork -VMHost $VMHost -Name "Internal02" -D

escription "Internal Host-Bound Network" -NetworkTag "Internal02" -BoundToV

MHost

PS C:\> $NewVNet

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost02

and stores the host object in $VMHost.

The third command creates a virtual network on VMHost02, names it Internal0

2, specifies a description and network tag, and binds the virtual network t

o the physical host. The virtual network object created by the New-VirtualN

etwork cmdlet is stored in $NewVNet.

The last command displays the properties of the virtual network object in $

NewVNet.

This virtual network is an internal, host-bound virtual network. Because it

is not attached to a physical network adapter on the host, it cannot acces

s networks external to the host. Virtual machines that are connected to thi

s internal virtual network on this host can communicate only with each othe

r. Because the network is bound to the host, network communication from vir

tual machines to the host is also possible.

3: Create a private virtual network that is not bound to the host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> New-VirtualNetwork -Name "UnboundVirtualNetwork1" -VMHost $VMHost

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost01

and stores the host object in $VMHost.

The last command creates a virtual network on VMHost01, names it UnboundVir

tualNetwork1. Because the network is not attached to a physical network ad

apter on the host, it cannot access networks external to the host. VMs tha

t are connected to this internal virtual network on this host can communica

te only with each other. The virtual network is not bound to the host, so n

etwork communication to the host is not possible.

4: Create an internal DHCP-enabled virtual network on a Virtual Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VirtualServerHost01.Contoso.com"

PS C:\> $NewVNet = New-VirtualNetwork -VMHost $VMHost -Name "Internal01" -D

escription "Internal DHCP Network" -Path "C:\Public" -NetworkTag "Internal0

1" -VirtualDHCPEnabled $TRUE -VirtualDHCPNetworkAddress 192.168.1.0 -Virtua

lDHCPNetworkMask 255.255.255.0 -VirtualDHCPStartingIPAddress 192.168.1.100

-VirtualDHCPEndingIPAddress 192.168.1.200 -VirtualDHCPServerAddress 192.168

.1.10 -VirtualDHCPDefaultGatewayAddress 192.168.1.1 -VirtualDHCPDNSServer 1

92.168.1.10 -VirtualDHCPWINSServer 192.168.1.10 -VirtualDHCPIPAddressLeaseT

ime 2160 -VirtualDHCPLeaseRenewalTime 1080 -VirtualDHCPLeaseRebindingTime 1

620

PS C:\> $NewVNet

BEFORE YOU START: Note that DHCP-enabled virtual networks apply only to Vi

rtual Server hosts.

The first command connects to VMMServer1.

The second command gets the object that represents the host named VirtualSe

rverHost01 and stores the host object in variable $VMHost.

The third command creates a virtual network on VirtualServerHost01, names i

t Internal01, specifies the description, path, and network tag. Next, the c

ommand specifies the DCHP server settings. Finally, the command stores the

new virtual network object in $NewVNet.

The last command displays the properties of the virtual network object repr

esented by $NewVNet.

This virtual network is an internal virtual network. Because it is not atta

ched to a physical network adapter on the host, it cannot access networks e

xternal to the host. Virtual machines that are connected to this internal v

irtual network on this host can communicate with each other and can receive

DHCP leases from Virtual Server host's DHCP server.

REMARKS

For more information, type: "get-help New-VirtualNetwork -detailed".

For technical information, type: "get-help New-VirtualNetwork -full".

### Remove-VirtualNetwork

SYNOPSIS

Removes a virtual network object from a host managed by Virtual Machine Man

ager.

SYNTAX

Remove-VirtualNetwork [-VirtualNetwork] <VirtualNetwork> [-Confirm] [-JobGr

oup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously]

[<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects from the Virtual Machine Manager database that

represent virtual networks configured on a virtual machine host.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-Confirm

Prompts for confirmation before running the command.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific virtual network from a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> $Network = Get-VirtualNetwork -VMHost $VMHost -Name "Internal Netwo

rk"

PS C:\> Remove-VirtualNetwork -VirtualNetwork $Network

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object. The following commands use this server by default.

The second command gets the object that represents the host named VMHost01

from the VMM database and stores the host object in $VMHost.

The third command gets the object that represents the virtual network named

"Internal Network" that is configured on VMHost01 and stores the virtual n

etwork object in variable $Network.

The last command removes the virtual network called Internal Network from V

MHost01.

REMARKS

For more information, type: "get-help Remove-VirtualNetwork -detailed".

For technical information, type: "get-help Remove-VirtualNetwork -full".

### Set-VirtualNetwork

SYNOPSIS

Changes properties of a virtual network on a virtual machine host managed b

y Virtual Machine Manager.

SYNTAX

Set-VirtualNetwork [-VirtualNetwork] <VirtualNetwork> [-BoundToVMHost <Bool

ean>] [-Description <String>] [-HostBoundVLanId] [-JobGroup <Guid>] [-JobVa

riable <String>] [-Name <String>] [-NetworkTag <String>] [-PROTipID <Guid>]

[-RunAsynchronously] [-VirtualDHCPDefaultGatewayAddress <String>] [-Virtua

lDHCPDNSServer <String>] [-VirtualDHCPEnabled <Boolean>] [-VirtualDHCPEndin

gIPAddress <String>] [-VirtualDHCPIPAddressLeaseTime <Int32>] [-VirtualDHCP

LeaseRebindingTime <Int32>] [-VirtualDHCPLeaseRenewalTime <Int32>] [-Virtua

lDHCPNetworkAddress <String>] [-VirtualDHCPNetworkMask <String>] [-VirtualD

HCPServerAddress <String>] [-VirtualDHCPStartingIPAddress <String>] [-Virtu

alDHCPWINSServer <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes properties of a virtual network on a virtual machine host managed b

y Virtual Machine Manager.

Properties for a virtual network that you can change include:

\* ANY HOST - For a virtual network configured for virtual machines

deployed on any virtual machine host supported by Virtual Machine

Manager (a Hyper-V, Virtual Server, or VMware ESX host), you can set

or modify the name, description, or Network Tag.

\* HYPER-V HOST ONLY - If the host is a Hyper-V host, you can also

configure whether virtual machines are bound to the host (and can

thus access the host operating system), and you can specify a

numerical identifier for a virtual local area network (VLAN) on the

host.

\* VIRTUAL SERVER HOST ONLY - If the host is a Virtual Server host, you

can set or modify multiple properties related to Dynamic Host

Configuration Protocol (DHCP).

PARAMETERS

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-BoundToVMHost <Boolean>

Enables virtual machines connected to a virtual network on a Hyper-V ho

st to access the host operating system. This parameter does not apply t

o Virtual Server or VMware ESX hosts.

-Description <String>

Specifies a description for the specified object.

-HostBoundVLanId

Assigns a numerical identifier in the range 1-4094 to a virtual local a

rea network (a VLAN) configured on a Hyper-V host. This is the VLAN ID

that virtual machines use to access the host. This parameter does not a

pply to Virtual Server or VMware ESX hosts.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-NetworkTag <String>

Specifies a word or phrase to associate with a virtual network adapter

that is configured to connect to a specific internal or external networ

k on the host. The NetworkTag identifies all VMs with the same NetworkT

ag as members of the same network. VMM uses a NeworkTag (if one exists)

when it evaluates hosts as possible candidates on which to deploy a VM

. If the host does not include VMs on the network with the same Network

Tag as the VM to be placed, the host receives zero stars in the placeme

nt process.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VirtualDHCPDefaultGatewayAddress <String>

Specifies the IP address of the default DHCP gateway for a virtual netw

ork configured on a Virtual Server host. The default gateway is a local

IP router that forwards traffic beyond this virtual network. This para

meter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPDNSServer <String>

Specifies a set of IP addresses of WINS servers (as strings separated b

y the pipeline operator) for use by DHCP clients for NetBIOS name resol

ution on a virtual network configured on a Virtual Server host. This pa

rameter does not apply to Hyper-V hosts or VMware ESX hosts.

Example format:

-VirtualDHCPDNSServer "nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn"

-VirtualDHCPEnabled <Boolean>

Enables virtual DHCP on a virtual network on a Virtual Server host (whe

n set to TRUE), or disables virtual DHCP on the virtual network (when s

et to FALSE). This parameter does not apply to Hyper-V hosts or VMware

ESX hosts.

-VirtualDHCPEndingIPAddress <String>

Specifies the last IP address in the range of IP addresses that the vir

tual DHCP server maintains for assignment to virtual DHCP clients on a

virtual network configured on a Virtual Server host. This parameter doe

s not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPIPAddressLeaseTime <Int32>

Specifies, in minutes, the duration of the lease for IP addresses assig

ned to virtual DHCP clients by the virtual DHCP server on a virtual net

work configured on a Virtual Server host.

Default value: 2160 minutes (36 hours). Maximum value: 49,710 days.

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPLeaseRebindingTime <Int32>

Specifies, in minutes, the DHCP lease rebinding time used by virtual DH

CP clients of a virtual DHCP server on a virtual network configured on

a Virtual Server host.

Default value: 1620 minutes (27 hours). Maximum value: 49,710 days.

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPLeaseRenewalTime <Int32>

Specifies, in minutes, the amount of time that you want a virtual DHCP

client to wait before it tries to renew its current IP address with the

virtual DHCP server on a virtual network configured on a Virtual Serve

r host.

Default value: 1080 minutes (18 hours). Maximum value: up to 49,710 day

s.

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPNetworkAddress <String>

Specifies the network address for a virtual network configured on a Vir

tual Server host. The value specified must be a valid IP address and mu

st be the base of the subnet. This parameter does not apply to Hyper-V

hosts or VMware ESX hosts.

-VirtualDHCPNetworkMask <String>

Specifies the network mask for a virtual network configured on a Virtua

l Server host. The specified value must be a valid network mask. The de

fault value is 255.255.0.0. This parameter does not apply to Hyper-V ho

sts or VMware ESX hosts.

-VirtualDHCPServerAddress <String>

Specifies the IP address for a virtual DHCP server on a virtual network

configured on a Virtual Server host. This parameter does not apply to

Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPStartingIPAddress <String>

Specifies the first IP address in the range of IP addresses that a virt

ual DHCP server maintains for assignment to virtual DHCP clients on a v

irtual network configured on a Virtual Server host. The value specified

must be a valid IP address; must use the network address (IP address o

f this virtual network) and network mask (subnet mask for this virtual

network); and must start at 16 or higher. This parameter does not apply

to Hyper-V hosts or VMware ESX hosts.

-VirtualDHCPWINSServer <String>

Specifies a set of the IP addresses of WINS servers (as strings separat

ed by the pipeline operator) for use by DHCP clients on a virtual netwo

rk configured on a Virtual Server host. The array must contain a set of

valid IP addresses.

Example format:

-VirtualDHCPWINSServer "nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn|nnn.nnn.nnn.nnn

"

This parameter does not apply to Hyper-V hosts or VMware ESX hosts.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Disable virtual DHCP on a virtual network on a Virtual Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Interna

l Network"

PS C:\> Set-VirtualNetwork -VirtualNetwork $VirtualNetwork -VirtualDHCPEnab

led $FALSE

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the returned object in $VMHost.

The third command gets from VMHost01 the object that represents the virtual

network named "Internal Network" and stores the returned object in variabl

e $VirtualNetwork.

The last command disables (turns off) virtual DHCP on the virtual network.

2: Enable virtual DHCP on a virtual network on a Virtual Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Interna

l Network"

PS C:\> Set-VirtualNetwork -VirtualNetwork $VirtualNetwork -VirtualDHCPNetw

orkAddress "10.251.0.0" -VirtualDHCPNetworkMask "255.255.0.0" -VirtualDHCPS

tartingIPAddress "10.251.0.16" -VirtualDHCPEndingIPAddress "10.251.255.254"

-VirtualDHCPServerAddress "10.251.0.1" -VirtualDHCPDefaultGateway "192.168

.1.1" -VirtualDHCPDNSServer "192.168.1.2,192.168.1.3" -VirtualDHCPWINSServe

r "192.168.1.4,192.168.1.5" -VirtualDHCPIPAddressLeaseTime 1080 -VirtualDHC

PLeaseRenewalTime 500 -VirtualDHCPLeaseRebindingTime 900

The first command connects to VMMServer1.

The second command gets the object that represents VMHost02 and stores the

host object in $VMHost.

The third command gets from VMHost02 the object that represents the virtual

network named "Internal Network" and stores the virtual network object in

$VirtualNetwork.

The last command configures a series of settings that together enable virtu

al DHCP on that virtual network.

3: Unbind a virtual network from the host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03.Contoso.com"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Host Bo

und Network"

PS C:\> Set-VirtualNetwork -VirtualNetwork $VirtualNetwork -BoundToVMHost $

False

The first command connects to VMMServer1.

The second command gets the object that represents VMHost03 and stores the

host object in $VMHost.

The third command gets from VMHost03 the object that represents the virtual

network named "Host Bound Network" and stores the virtual network object i

n $VirtualNetwork.

The last command renames the virtual network to "Unbound Network" and sets

the -BoundToVMHost parameter to FALSE. This unbinds the virtual network fro

m the host, which prevents any virtual machines that are attached to this v

irtual network from accessing the host through this network.

REMARKS

For more information, type: "get-help Set-VirtualNetwork -detailed".

For technical information, type: "get-help Set-VirtualNetwork -full".

## VirtualNetworkAdapter

### Get-VirtualNetworkAdapter

SYNOPSIS

Gets Virtual Machine Manager virtual network adapter objects from a virtual

machine, template, or hardware profile.

SYNTAX

Get-VirtualNetworkAdapter -All [-VMMServer [<String ServerConnection>]] [<C

ommonParameters>]

Get-VirtualNetworkAdapter -Template [<Template String>] [<CommonParameters>

]

Get-VirtualNetworkAdapter -VM [<String VM>] [<CommonParameters>]

Get-VirtualNetworkAdapter -HardwareProfile <HardwareProfile> [<CommonParame

ters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual network adapters from a vir

tual machine object, from a template object, or from a hardware profile obj

ect in a Virtual Machine Manager environment .

For more information about virtual network adapters, type:

Get-Help New-VirtualNetworkAdapter -detailed

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get virtual network adapters from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> Get-VirtualNetworkAdapter -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command selects from all virtual machine objects in the VMM data

base the object that represents VM01 and stores this object in variable $VM

.

The last command gets all objects that represent virtual network adapters o

n VM01 and displays information about these adapters to the user.

2: Get virtual network adapters from a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> Get-VirtualNetworkAdapter -Template $Template

The first command connects to VMMServer1.

The second command selects from all template objects in the VMM library the

object that represents Template1 and stores this object in $Template.

The last command gets all objects that represent virtual network adapters o

n Template1 and displays information about these adapters to the user.

3: Get virtual network adapters from a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> Get-VirtualNetworkAdapter -HardwareProfile $HWProfile

The first command connects to VMMServer1.

The second command selects from all hardware profile objects in the VMM lib

rary the object that represents HardwareProfile1 and stores this object in

$HWProfile.

The last command gets all objects that represent virtual network adapters o

n HardwareProfile1 and displays information about these adapters to the use

r.

REMARKS

For more information, type: "get-help Get-VirtualNetworkAdapter -detailed".

For technical information, type: "get-help Get-VirtualNetworkAdapter -full"

.

### New-VirtualNetworkAdapter

SYNOPSIS

Creates a virtual network adapter on a virtual machine, template, or hardwa

re profile used in Virtual Machine Manager.

SYNTAX

New-VirtualNetworkAdapter [[-VirtualNetwork] <VirtualNetwork>] -VM [<String

VM>] [-JobVariable <String>] [-NetworkLocation <String>] [-NetworkTag <Str

ing>] [-NoConnection] [-PhysicalAddress <String>] [-PhysicalAddressType <St

ring>] [-PROTipID <Guid>] [-RunAsynchronously] [-Synthetic] [-VLANEnabled]

[-VLANID <Int32>] [<CommonParameters>]

New-VirtualNetworkAdapter [[-VirtualNetwork] <VirtualNetwork>] -JobGroup <G

uid> [-JobVariable <String>] [-NetworkLocation <String>] [-NetworkTag <Stri

ng>] [-NoConnection] [-PhysicalAddress <String>] [-PhysicalAddressType <Str

ing>] [-PROTipID <Guid>] [-RunAsynchronously] [-Synthetic] [-VLANEnabled] [

-VLANID <Int32>] [-VMMServer [<String ServerConnection>]] [<CommonParameter

s>]

New-VirtualNetworkAdapter [[-VirtualNetwork] <VirtualNetwork>] -Template [<

Template String>] [-JobVariable <String>] [-NetworkLocation <String>] [-Net

workTag <String>] [-NoConnection] [-PhysicalAddress <String>] [-PhysicalAdd

ressType <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Synthetic] [-V

LANEnabled] [-VLANID <Int32>] [<CommonParameters>]

New-VirtualNetworkAdapter [[-VirtualNetwork] <VirtualNetwork>] -HardwarePro

file <HardwareProfile> [-JobVariable <String>] [-NetworkLocation <String>]

[-NetworkTag <String>] [-NoConnection] [-PhysicalAddress <String>] [-Physic

alAddressType <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Synthetic

] [-VLANEnabled] [-VLANID <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual network adapter on a virtual machine or on a template or

hardware profile used to create virtual machines managed by Virtual Machine

Manager.

NETWORK LOCATION

----------------

You can, optionally, use the New-VirtualNetworkAdapter cmdlet to specify a

network location and connect the virtual network adapter to a virtual netwo

rk configured on the host when you create the adapter, or you can configure

those and other settings later by using the Set-VirtualNetworkAdapter cmdl

et.

STATIC OR DYNAMIC MAC ADDRESS

-----------------------------

Additionally, you can specify whether the virtual network adapter uses a st

atic or dynamic physical address (MAC address), and you can specify a stati

c MAC address.

EMULATED OR SYNTHETIC VIRTUAL NETWORK ADAPTERS

----------------------------------------------

You can use the New-VirtualNetworkAdapter cmdlet to create an adapter whose

type is either emulated (the default) or synthetic.

For virtual machines on any type of host (Hyper-V, Virtual Server, or VMwar

e), you can configure a virtual network adapter on the virtual machine that

emulates a specific physical network adapter.

For virtual machines on Hyper-V hosts, if the guest operating system instal

led on a virtual machine is a virtualization-aware operating system (for ex

ample, Windows Server 2008 and some versions of Linux), VMM 2008 lets you c

onfigure a high-performance synthetic virtual network adapter on the virtua

l machine to communicate with the physical hardware on the host. You must e

xplicitly specify that a virtual network adapter is synthetic by using the

Synthetic parameter.

VIRTUAL LOCAL AREA NETWORK

--------------------------

VMM 2008 extends virtual networking support to include support for configur

ing one or more virtual area networks (VLANs) on a host for use by virtual

machines deployed on that host. You can use the New-VirtualNetworkAdapter c

mdlet (or the Set-VirtualNetworkAdapter cmdlet) with the VLAN parameters to

attach the virtual network adapter on a virtual machine to a VLAN. To conf

igure corresponding VLAN settings on the host network adapter, use the Add-

VMHostNetworkAdapter cmdlet or the Set-VMHostNetworkAdapter cmdlet.

For an illustration of how to configure VLANs, see the examples for this cm

dlet, and see the examples for New-VMHostNetworkAdapter and Set-VMHostNetwo

rkAdapter.

For more information about VLANs, type:

Get-Help about\_VMM\_2008\_Virtual\_Networking

PARAMETERS

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-NetworkLocation <String>

Specifies the network location for a physical network adapter or for a

virtual network adapter, or changes the default network location of a h

ost's physical network adapter.

Example formats:

-NetworkLocation $NetLoc ($NetLoc might contain "Corp.Contoso.com")

-OverrideNetworkLocation $TRUE –NetworkLocation "HostNICNewLocation.Co

ntoso.com"

-NetworkTag <String>

Specifies a word or phrase to associate with a virtual network adapter

that is configured to connect to a specific internal or external networ

k on the host. The NetworkTag identifies all VMs with the same NetworkT

ag as members of the same network. VMM uses a NeworkTag (if one exists)

when it evaluates hosts as possible candidates on which to deploy a VM

. If the host does not include VMs on the network with the same Network

Tag as the VM to be placed, the host receives zero stars in the placeme

nt process.

-NoConnection

Disconnects a virtual network adapter from a virtual network.

-PhysicalAddress <String>

Specifies the physical address (MAC address) of a physical or virtual n

etwork adapter.

Note: In VMM 2007, this parameter was named EthernetAddress.

-PhysicalAddressType <String>

Specifies the type of physical address (MAC address) to use for a virtu

al network adapter:

Valid values: Static, Dynamic

Note: In VMM 2007, this parameter was named EthernetAddressType.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-Synthetic

Specifies that a device (such as a virtual network adapter) on a virtua

l machine deployed on a Hyper-V host is a high-performance synthetic de

vice. Requires a virtualization-aware guest operating system on the vir

tual machine.

-VLANEnabled

Enables a virtual LAN (a VLAN) for use by virtual machines on a Hyper-V

or VMware ESX host.

Example format for a single VLAN:

-VLANEnabled -VLANMode "Access" -VLANID 35

Example format for multiple VLANs:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

-VLANID <Int32>

Assigns a numerical identifier in the range 1-4094 to a virtual network

adapter on a virtual machine or to a physical network adapter on a Hyp

er-V or VMware ESX host.

Configure a VLANID on a Hyper-V or VMware ESX host:

\* On an externally bound physical network adapter.

Configure a VLANID on a virtual network adapter of a virtual machine:

\* Bound to a physical network adapter on the host, or

\* Bound to an internal virtual network on the host.

Example format:

-VLANEnabled

-VLANMode "Access" -VLANID 35

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a virtual network adapter on a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name VM01

PS C:\> New-VirtualNetworkAdapter -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VM01 from the VMM databa

se and stores the virtual machine object in variable $VM.

The last command creates a virtual network adapter on VM01.

2: Create a virtual network adapter on a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template2" }

PS C:\> New-VirtualNetworkAdapter -Template $Template

The first command connects to VMMServer1.

The second command gets the object that represents Template2 from the VMM l

ibrary and stores the template object in $Template.

The last command creates a virtual network adapter on Template2.

3: Create an emulated virtual network adapter and a synthetic virtual netwo

rk adapter on a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file3" }

PS C:\> New-VirtualNetworkAdapter -HardwareProfile $HWProfile

PS C:\> New-VirtualNetworkAdapter -HardwareProfile $HWProfile -Synthetic

The first command connects to VMMServer1.

The second command gets the object that represents HardwareProfile3 from th

e VMM library and stores the hardware profile object in $HWProfile.

The third command creates a virtual network adapter (a "native" or emulated

adapter) on HardwareProfile3.

The last command creates a synthetic virtual network adapter on HardwarePro

file3.

4: Create a virtual network adapter on a VM and assign it a unique MAC addr

ess.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MAC = New-PhysicalAddress -Commit

PS C:\> $VM = Get-VM -Name "VM04"

PS C:\> New-VirtualNetworkAdapter -VM $VM -PhysicalAddress $MAC

The first command connects to VMMServer1.

The second command get the next available physical address (MAC address) fr

om the address pool configured on VMMServer1; uses the -Commit parameter to

commit that address for use (for example, it can be assigned to a virtual

network adapter on a VM, a template, or a hardware profile); and stores the

MAC address object in $MAC.

The third command gets the object that represents the virtual machine named

VM04 and stores the virtual machine object in $VM.

The last command creates a new virtual network adapter on VM04 and assigns

the MAC address stored in $MAC to that virtual network adapter.

NOTE: This example assumes that a range of MAC addresses has already been c

onfigured for your VMM environment. For information about how to set the ra

nge of MAC addresses, type: Get-Help Set-VMMServer -example

5: Create a virtual network adapter with a static MAC address and a specifi

c VLAN ID.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM05" -and $\_.VMHost.Name -eq "

VMHost05.Contoso.com" }

PS C:\> $Loc = Get-NetworkLocation | where { $\_.Name -eq "Corp.Contoso.com"

}

PS C:\> $VNetwork = Get-VirtualNetwork | where { $\_.Name -match "External N

etwork1" -and $\_.VMHost -match "VMHost05" }

PS C:\> New-VirtualNetworkAdapter -VM $VM -VirtualNetwork $VNetwork -Physic

alAddress "00-00-00-11-12-1A" -PhysicalAddressType "Static" -NetworkLocatio

n $Loc -VLANEnabled $TRUE -VLANId 3

The first command connects to the VMM server.

The second command gets the object that represents the virtual machine name

d VM05 that is deployed on host server VMHost05 and stores the virtual mach

ine object in $VM.

The third command gets the object that represents the network location (nam

ed "Corp.Contoso.com") reported to VMM by all of the hosts that VMM manages

, and stores the virtual network object in $Loc.

The fourth command gets the object that represents a virtual network (named

"External Network1") from VMHost05 and stores the virtual network object i

n $VNetwork.

The last command creates a new virtual network adapter for VM05; connects t

he virtual network adapter to the virtual network represented by $VNetwork;

specifies a static physical address (MAC address) type for the virtual net

work adapter and sets its static physical address to 00-00-00-11-12-1A; ena

bles VLAN, and specifies a VLAN ID of 2,

CAUTION: This example assumes that that your host is already connected to a

VLAN or, if not, that your host has two network adapters. If your host has

a single network adapter (as might be the case if you are experimenting wi

th VMM cmdlets in a lab setting), assigning the adapter to a VLAN that is u

navailable to the VMM server will prevent VMM from managing the host. You c

an perform the steps in this example on a host that has only one network ad

apter if you first install the Microsoft Loopback Adapter on your server.

REMARKS

For more information, type: "get-help New-VirtualNetworkAdapter -detailed".

For technical information, type: "get-help New-VirtualNetworkAdapter -full".

### Remove-VirtualNetworkAdapter

SYNOPSIS

Removes a virtual network adapter object from Virtual Machine Manager.

SYNTAX

Remove-VirtualNetworkAdapter [-VirtualNetworkAdapter] <VirtualNetworkAdapte

r> [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsync

hronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent virtual network adapters from a

virtual machine, template, or hardware profile used in a Virtual Machine Ma

nager environment.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualNetworkAdapter <VirtualNetworkAdapter>

Specifies a virtual network adapter object for a virtual machine.

TYPE OF HOST NUMBER OF VIRTUAL NETWORK ADAPTERS

------------ ----------------------------------

Virtual Server Up to 4 emulated adapters per VM.

Hyper-V Up to 4 emulated adapters per VM.

Up to 8 synthetic adapters per VM.

(Exception: no driver available for an emulated

network adapter on a Windows Server 2003 x64 guest.)

VMware ESX Up to 4 emulated adapters per VM.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a virtual network adapter with the specified MAC address from a v

irtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $Adapter = Get-VirtualNetworkAdapter -VM $VM | where { $\_.PhysicalA

ddress -eq "12-34-56-78-90-12" }

PS C:\> Remove-VirtualNetworkAdapter -VirtualNetworkAdaptor $Adapter

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command gets the object that represents the virtual network adapt

er on VM01 that has the specified physical address (MAC address), 12-34-56-

78-90-12, and stores this virtual network adapter object in variable $Adapt

er.

The last command removes the virtual network adapter whose object is stored

in $Adapter from VM01.

2: Remove a virtual network adapter connected to a specific virtual network

from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $Adapter = Get-VirtualNetworkAdapter -VM $VM | where { $\_.VirtualNe

twork -eq "ContosoNet" }

PS C:\> Remove-VirtualNetworkAdapter -VirtualNetworkAdapter $Adapter

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command gets the object that represents the virtual network adapt

er on VM02 that is currently connected to the specified virtual network (Co

ntosoNet) and stores this virtual network adapter object in $Adapter.

The last command removes the virtual network adapter object stored in $Adap

ter from VM02.

3: Remove the only virtual network adapter from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> $Adapter = Get-VirtualNetworkAdapter -VM $VM

PS C:\> Remove-VirtualNetworkAdapter -VirtualNetworkAdapter $Adapter

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM03 and stores the virtual machine object in $VM.

The third command gets the object that represents the single virtual networ

k adapter on VM03 and stores this virtual network adapter object in $Adapte

r. (This example assumes that VM03 has only one virtual network adapter.)

The last command removes the object stored in $Adapter from VM03. Because $

Adapter contains only one element, you can use this command to pass the vi

rtual network adapter object directly as a parameter to the Remove-VirtualN

etworkAdapter cmdlet.

Compare the syntax used in this example that removes a single object with t

he syntax in the following examples that remove objects from multiple-eleme

nt arrays.

4: Remove all virtual network adapters from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM04"

PS C:\> $Adapters = Get-VirtualNetworkAdapter -VM $VM

PS C:\> $Adapters | Remove-VirtualNetworkAdapter

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM04 and stores the virtual machine object in $VM.

The third command gets all objects that represent virtual network adapters

on VM04 and stores these virtual network objects in $Adapters (an object ar

ray).

The last command passes each object stored in $Adapters to Remove-VirtualNe

tworkAdapter, which removes each virtual network adapter object from VM04.

5: Remove the second virtual network adapter from a virtual machine that ha

s three virtual network adapters.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM05"

PS C:\> $Adapters = Get-VirtualNetworkAdapter -VM $VM

PS C:\> $Adapters[1] | Remove-VirtualNetworkAdapter

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM05 and stores the virtual machine object in $VM.

The third command gets all objects that represent virtual network adapters

on VM05 and stores these virtual network objects in $Adapters (an object ar

ray). This example assumes that VM05 has three virtual network adapters and

therefore the array contains three elements (counting 0 to 2).

The last command passes the second virtual network adapter (object [1]) sto

red in $Adapters to the Remove-VirtualNetworkAdapter cmdlet, which removes

this virtual network adapter object from VM05.

REMARKS

For more information, type: "get-help Remove-VirtualNetworkAdapter -detaile

d".

For technical information, type: "get-help Remove-VirtualNetworkAdapter -fu

ll".

### Set-VirtualNetworkAdapter

SYNOPSIS

Changes properties of a virtual network adapter associated with a virtual m

achine, or with a template or hardware profile used to create virtual machi

nes in Virtual Machine Manager.

SYNTAX

Set-VirtualNetworkAdapter [-VirtualNetworkAdapter] <VirtualNetworkAdapter>

[-JobGroup <Guid>] [-JobVariable <String>] [-NetworkLocation <String>] [-Ne

tworkTag <String>] [-NoConnection] [-PhysicalAddress <String>] [-PhysicalAd

dressType <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-VirtualNetwor

k <VirtualNetwork>] [-VLANEnabled] [-VLANID <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual network adapter associated with

a virtual machine, or with a template or hardware profile used to create v

irtual machines in a Virtual Machine Manager environment.

You can use this cmdlet to:

\* Connect a virtual network adapter to a virtual network.

\* Disconnect a virtual network adapter from a virtual network.

\* Specify a network location and network tag on a virtual network adapter.

\* Specify a physical address (MAC address) on this virtual network adapter.

\* Enable the use of a virtual local area network (VLAN) and specify a

VLAN ID (numerical identifier) for that VLAN on this virtual network

adapter.

PARAMETERS

-VirtualNetworkAdapter <VirtualNetworkAdapter>

Specifies a virtual network adapter object for a virtual machine.

TYPE OF HOST NUMBER OF VIRTUAL NETWORK ADAPTERS

------------ ----------------------------------

Virtual Server Up to 4 emulated adapters per VM.

Hyper-V Up to 4 emulated adapters per VM.

Up to 8 synthetic adapters per VM.

(Exception: no driver available for an emulated

network adapter on a Windows Server 2003 x64 guest.)

VMware ESX Up to 4 emulated adapters per VM.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-NetworkLocation <String>

Specifies the network location for a physical network adapter or for a

virtual network adapter, or changes the default network location of a h

ost's physical network adapter.

Example formats:

-NetworkLocation $NetLoc ($NetLoc might contain "Corp.Contoso.com")

-OverrideNetworkLocation $TRUE –NetworkLocation "HostNICNewLocation.Co

ntoso.com"

-NetworkTag <String>

Specifies a word or phrase to associate with a virtual network adapter

that is configured to connect to a specific internal or external networ

k on the host. The NetworkTag identifies all VMs with the same NetworkT

ag as members of the same network. VMM uses a NeworkTag (if one exists)

when it evaluates hosts as possible candidates on which to deploy a VM

. If the host does not include VMs on the network with the same Network

Tag as the VM to be placed, the host receives zero stars in the placeme

nt process.

-NoConnection

Disconnects a virtual network adapter from a virtual network.

-PhysicalAddress <String>

Specifies the physical address (MAC address) of a physical or virtual n

etwork adapter.

Note: In VMM 2007, this parameter was named EthernetAddress.

-PhysicalAddressType <String>

Specifies the type of physical address (MAC address) to use for a virtu

al network adapter:

Valid values: Static, Dynamic

Note: In VMM 2007, this parameter was named EthernetAddressType.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-VLANEnabled

Enables a virtual LAN (a VLAN) for use by virtual machines on a Hyper-V

or VMware ESX host.

Example format for a single VLAN:

-VLANEnabled -VLANMode "Access" -VLANID 35

Example format for multiple VLANs:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

-VLANID <Int32>

Assigns a numerical identifier in the range 1-4094 to a virtual network

adapter on a virtual machine or to a physical network adapter on a Hyp

er-V or VMware ESX host.

Configure a VLANID on a Hyper-V or VMware ESX host:

\* On an externally bound physical network adapter.

Configure a VLANID on a virtual network adapter of a virtual machine:

\* Bound to a physical network adapter on the host, or

\* Bound to an internal virtual network on the host.

Example format:

-VLANEnabled

-VLANMode "Access" -VLANID 35

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Connect a virtual network adapter to a virtual network.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $Adapter = Get-VirtualNetworkAdapter -VM $VM | where { $\_.PhysicalA

ddress -eq "12-34-56-78-90-12" }

PS C:\> Set-VirtualNetworkAdapter -VirtualNetworkAdapter $Adapter -VirtualN

etwork "Network1"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command selects from all virtual network adapter objects on VM01

the object that represents the virtual network adapter with the physical ad

dress (MAC address) 12-34-56-78-90-12 and stores this virtual network adapt

er object in variable $Adapter.

The last command connects the virtual network adapter (whose object is stor

ed in $Adapter) to the virtual network named Network1 on the host that cont

ains VM01.

2: Specify a static physical address (MAC address) for a virtual network ad

apter.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM02" }

PS C:\> $Adapter = Get-VirtualNetworkAdapter -VM $VM | where { $\_.ID -eq "1

234-1234-

1234-1234-1234" }

PS C:\> Set-VirtualNetworkAdapter -VirtualNetworkAdapter $Adapter -Physical

AddressType "Static" -PhysicalAddress "00-03-FF-70-C1-F3"

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command selects from all virtual network adapters on VM02 the vir

tual network adapter with the specified ID and stores its object in $Adapte

r. This example assumes that, currently, this adapter has a dynamic physica

l address (MAC address).

The last command specifies that Adapter02 uses a static MAC address type an

d sets its static MAC address to 00-03-FF-70-C1-F3.

3: Specify a static MAC address and assign it to an existing virtual networ

k adapter.

PS C:\> $VMMServer = Get-VMMServer "VMMServer1.contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> if($VMMServer.PhysicalAddressRangeStart -ne $null){$StaticMac = New

-PhysicalAddress -Commit}

PS C:\> Set-VirtualNetworkAdapter -VirtualNetworkAdapter (Get-VirtualNetwor

kAdapter -VM $VM | where { $\_.ID -eq "1234-1234-1234-1234-1234"}) -Physical

AddressType "Static" -PhysicalAddress $StaticMac

The first command connects to VMMServer1 and stores the server object in $V

MMServer.

The second command gets the object that represents the virtual machine name

d VM03 and stores the virtual machine object in $VM.

The third command checks whether the static MAC pool has been configured; i

f it does exist, the command retrieves the object for the next available MA

C address from the pool and stores the MAC address object in $StaticMac.

The last command gets the object that represents an adapter on VM03 by ID,

specifies that the adapter uses a static address type, and sets the physica

l address (MAC address) to the value retrieved from the static MAC pool.

4. Disconnect the specified virtual network adapter from the virtal network

.

PS C:\> Get-VMMServer "VMMServer1.contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM04" -and $\_.VMHost.Name -eq "

VMHost04.Contoso.com"}

PS C:\> $Adapters = Get-VirtualNetworkAdapter -VM $VM

PS C:\> Set-VirtualNetworkAdapter -VirtualNetworkAdapter $Adapter[1] -NoCon

nection

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM04 deployed on the host named VMHost04 and stores the virtual machine o

bject in $VM.

The third command gets all virtual network adapters on VM04 and stores the

adapter objects in $Adapters. This example assumes that VM04 has at least 2

virtual network adapters.

The last command uses the NoConnection parameter to disconnect the second v

irtual network adapter (Adapters[1]) from any virtual network that it is co

nnected to.

REMARKS

For more information, type: "get-help Set-VirtualNetworkAdapter -detailed".

For technical information, type: "get-help Set-VirtualNetworkAdapter -full"

.

## VirtualSCSIAdapter

### Get-VirtualSCSIAdapter

SYNOPSIS

Gets Virtual Machine Manager virtual SCSI adapter objects from a virtual ma

chine, template, or hardware profile.

SYNTAX

Get-VirtualSCSIAdapter -HardwareProfile <HardwareProfile> [<CommonParameter

s>]

Get-VirtualSCSIAdapter -VM [<String VM>] [<CommonParameters>]

Get-VirtualSCSIAdapter -Template [<Template String>] [<CommonParameters>]

Get-VirtualSCSIAdapter -All [-VMMServer [<String ServerConnection>]] [<Comm

onParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual SCSI adapters used in a Vir

tual Machine Manager environment from a virtual machine object, from a temp

late object, or from a hardware profile object.

PARAMETERS

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get virtual SCSI adapters from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> Get-VirtualSCSIAdapter -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object. The following commands use this server by default.

The second command selects from all virtual machine objects in the VMM data

base the object that represents VM01 and stores this object in variable $VM

.

The last command gets all objects that represent virtual SCSI adapters on V

M01 and displays information about these adapters to the user.

2: Get virtual SCSI adapters from a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> Get-VirtualSCSIAdapter -Template $Template

The first command connects to VMMServer1.

The second command selects from all template objects in the VMM library the

object that represents Template1 and stores this object in $Template.

The last command gets all objects that represent virtual SCSI adapters on T

emplate1 and displays information about these adapters to the user.

3: Get virtual SCSI adapters from a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> Get-VirtualSCSIAdapter -HardwareProfile $HWProfile

The first command connects to VMMServer1.

The second command selects from all hardware profile objects in the VMM lib

rary the object that represents HardwareProfile1 and stores this object in

$HWProfile.

The last command gets all objects that represent virtual SCSI adapters on H

ardwareProfile1 and displays information about these adapters to the user.

REMARKS

For more information, type: "get-help Get-VirtualSCSIAdapter -detailed".

For technical information, type: "get-help Get-VirtualSCSIAdapter -full".

### New-VirtualSCSIAdapter

SYNOPSIS

Creates a virtual SCSI adapter on a virtual machine, template, or hardware

profile used in Virtual Machine Manager.

SYNTAX

New-VirtualSCSIAdapter [[-AdapterID] <Int32>] -Template [<Template String>]

[-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Shared <

Boolean>] [<CommonParameters>]

New-VirtualSCSIAdapter [[-AdapterID] <Int32>] -JobGroup <Guid> [-JobVariabl

e <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Shared <Boolean>] [-V

MMServer [<String ServerConnection>]] [<CommonParameters>]

New-VirtualSCSIAdapter [[-AdapterID] <Int32>] -HardwareProfile <HardwarePro

file> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Sha

red <Boolean>] [<CommonParameters>]

New-VirtualSCSIAdapter [[-AdapterID] <Int32>] -VM [<String VM>] [-JobVariab

le <String>] [-PROTipID <Guid>] [-RunAsynchronously] [-Shared <Boolean>] [<

CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual SCSI adapter on a virtual machine, template, or hardware

profile used in a Virtual Machine Manager environment. After you create the

virtual SCSI adapter, you can use the Set-VirtualSCSIAdapter cmdlet to mod

ify its settings.

Note: Using the Shared parameter to share a virtual SCSI adapter on a virtu

al machine in order to enable guest clustering is supported only if the vir

tual machine is deployed on a Virtual Server host or ESX host. The Shared p

arameter is not used for a virtual machine on a Hyper-V host because a virt

ual machine on a Hyper-V host uses iSCSI for shared storage.

PARAMETERS

-AdapterID <Int32>

Specifies the logical unit number, or LUN ID, for a virtual SCSI adapte

r on a virtual machine on a Virtual Server host or on a template. (For

a VMware ESX host, the default is 7 and cannot be changed. Hyper-V does

not expose this value, and it cannot be changed.)

Valid values for AdapterID for a Virtual Server host: 6 or 7. Default v

alue: 7.

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-Shared <Boolean>

Specifies that a virtual SCSI adapter will be shared so that it can be

used in guest clustering.

TYPE OF HOST USES SHARED PARAMETER

------------ ---------------------

Virtual Server host Yes

VMware ESX host Yes

Hyper-V host No (for guest clustering, use iSCSI storage)

Note: When sharing a SCSI controller on a virtual machine on an ESX hos

t, VMM defaults the SCSI sharing policy on VMware to "physical."

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a virtual SCSI adapter on a VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> New-VirtualSCSIAdapter -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VM01 from the VMM databa

se and stores the virtual machine object in variable $VM.

The last command creates a virtual SCSI adapter on VM01.

2: Create a virtual SCSI adapter on a template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -eq "Template1" }

PS C:\> New-VirtualSCSIAdapter -Template $Template

The first command connects to VMMServer1.

The second command gets the object that represents Template1 from the VMM l

ibrary and stores the template object in $Template.

The last command creates a virtual SCSI adapter on Template1.

3: Create a virtual SCSI adapter on a hardware profile.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "HardwarePro

file1" }

PS C:\> New-VirtualSCSIAdapter -HardwareProfile $HWProfile

The first command connects to VMMServer1.

The second command gets the object that represents HardwareProfile1 from th

e VMM library and stores the hardware profile object in $HWProfile.

The last command creates a virtual SCSI adapter on HardwareProfile1.

4: Create a virtual SCSI adapter used for guest clustering on a VM on a Vir

tual Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM02" }

PS C:\> New-VirtualSCSIAdapter -VM $VM -Shared $TRUE -AdapterID 6

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 from the VMM database and stores the virtual machine object in $VM.

This example assumes that VM02 is deployed on a Virtual Server host.

The last command creates a virtual SCSI adapter on VM02, sets the adapter I

D to 6, and specifies that the adapter is shared so that it can be used in

guest clustering.

NOTES:

Using the Shared parameter to share a virtual SCSI adapter on a VM and spec

ifying the adapter ID depends on the virtualization platform of the host:

\* If the VM is on a Virtual Server host, you can set the adapter ID

to 6 or 7.

\* If the VM is on a VMware ESX host, VMM supports the use of a shared

virtual SCSI adapter for guest clustering but sets the default value

to 7. You cannot change this value.

However, if the VM is deployed on a Hyper-V host, you must use iSCSI storag

e for guest clustering.

This type of clustering enables failover of applications or services betwee

n two virtual machines. Both virtual machines must be on the same physical

host and the shared disk must be a fixed (not dynamic) virtual hard disk.

For more information:

\* At the VMM command prompt, type:

Get-Help about\_VMM\_2008\_Failover\_Clusters

Scroll down to the sub-section "VMM 2008 AND GUEST CLUSTERS"

\* See the Windows Server 2008 help in the Failover Cluster Management

console for information about guest clusters for virtual machines

deployed on Hyper-V hosts.

\* See the Virtual Server help for information about guest clusters

for virtual machines deployed on Virtual Server hosts, and see

"Using Microsoft Virtual Server 2005 to Create and Configure a

Two-Node Microsoft Windows Server 2003 Cluster" at:

http://go.microsoft.com/fwlink/?LinkId=30134

REMARKS

For more information, type: "get-help New-VirtualSCSIAdapter -detailed".

For technical information, type: "get-help New-VirtualSCSIAdapter -full".

### Remove-VirtualSCSIAdapter

SYNOPSIS

Removes a virtual SCSI adapter object from Virtual Machine Manager.

SYNTAX

Remove-VirtualSCSIAdapter [-VirtualSCSIAdapter] <VirtualSCSIAdapter> [-JobG

roup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously

] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent virtual SCSI adapters from a vir

tual machine, template, or hardware profile used in a Virtual Machine Manag

er environment.

The Remove-VirtualSCSIAdapter cmdlet removes a virtual SCSI adapter success

fully only if the adapter does not have any devices attached to it.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualSCSIAdapter <VirtualSCSIAdapter>

Specifies a virtual SCSI adapter object for a virtual machine.

TYPE OF HOST NUMBER OF VIRTUAL SCSI ADAPTERS

------------ -------------------------------

Virtual Server Up to 4 virtual SCSI adapters per VM,

and up to 7 devices per adapter.

Supports a virtual disk drive size up to 2040 GB.

Hyper-V Up to 4 synthetic virtual SCSI adapters per VM,

and up to 64 devices per adapter

Supports a virtual disk drive size up to 2040 GB.

Does not support emulated virtual SCSI adapters.

VMware ESX Up to 4 virtual SCSI adapters per VM,

and up to 15 devices per adapter.

Supports a virtual disk drive size up to 2048 GB.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the third virtual SCSI adapter from a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> $Adapter = Get-VirtualSCSIAdapter -VM $VM

PS C:\> $Adapter[2] | Remove-VirtualSCSIAdapter

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command gets all objects that represent virtual SCSI adapters on

VM01 and stores these virtual SCSI adapter objects in $Adapter (an object a

rray). A virtual machine can have up to four virtual SCSI adapters attached

. This example assumes that VM01 has four virtual SCSI adapters and therefo

re the array contains four elements (counting 0 through 3).

The last command passes the third virtual SCSI adapter (object [2]) stored

in $Adapter) to Remove-VirtualSCSIAdapter, which removes this virtual SCSI

adapter from VM01.

REMARKS

For more information, type: "get-help Remove-VirtualSCSIAdapter -detailed".

For technical information, type: "get-help Remove-VirtualSCSIAdapter -full"

.

### Set-VirtualSCSIAdapter

SYNOPSIS

Changes properties of a virtual SCSI adapter used in Virtual Machine Manage

r.

SYNTAX

Set-VirtualSCSIAdapter [-VirtualSCSIAdapter] <VirtualSCSIAdapter> [-Adapter

ID <Int32>] [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [

-RunAsynchronously] [-Shared <Boolean>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual SCSI adapter used in a Virtual

Machine Manager environment. Modifiable settings include specifying whether

or not a virtual SCSI adapter is shared and setting the adapter ID.

Note: Using the Shared parameter to share a virtual SCSI adapter on a virtu

al machine in order to enable guest clustering is supported only if the vir

tual machine is deployed on a Virtual Server host or ESX host. The Shared p

arameter is not used for a virtual machine on a Hyper-V host because a virt

ual machine on a Hyper-V host uses iSCSI for shared storage.

PARAMETERS

-VirtualSCSIAdapter <VirtualSCSIAdapter>

Specifies a virtual SCSI adapter object for a virtual machine.

TYPE OF HOST NUMBER OF VIRTUAL SCSI ADAPTERS

------------ -------------------------------

Virtual Server Up to 4 virtual SCSI adapters per VM,

and up to 7 devices per adapter.

Supports a virtual disk drive size up to 2040 GB.

Hyper-V Up to 4 synthetic virtual SCSI adapters per VM,

and up to 64 devices per adapter

Supports a virtual disk drive size up to 2040 GB.

Does not support emulated virtual SCSI adapters.

VMware ESX Up to 4 virtual SCSI adapters per VM,

and up to 15 devices per adapter.

Supports a virtual disk drive size up to 2048 GB.

-AdapterID <Int32>

Specifies the logical unit number, or LUN ID, for a virtual SCSI adapte

r on a virtual machine on a Virtual Server host or on a template. (For

a VMware ESX host, the default is 7 and cannot be changed. Hyper-V does

not expose this value, and it cannot be changed.)

Valid values for AdapterID for a Virtual Server host: 6 or 7. Default v

alue: 7.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-Shared <Boolean>

Specifies that a virtual SCSI adapter will be shared so that it can be

used in guest clustering.

TYPE OF HOST USES SHARED PARAMETER

------------ ---------------------

Virtual Server host Yes

VMware ESX host Yes

Hyper-V host No (for guest clustering, use iSCSI storage)

Note: When sharing a SCSI controller on a virtual machine on an ESX hos

t, VMM defaults the SCSI sharing policy on VMware to "physical."

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Share a specific virtual SCSI adapter on a virtual machine to enable it

for guest clustering.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> $Adapter = Get-VirtualSCSIAdapter -VM $VM

PS C:\> Set-VirtualSCSIAdapter -VirtualSCSIAdapter $Adapter[3] -Shared $TRU

E

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 deployed on a Virtual Server host from the VMM database and stores t

he virtual machine object in variable $VM.

The third command gets all objects that represent virtual SCSI adapters on

VM01 and stores these virtual SCSI adapter objects in $Adapter (an object a

rray). A virtual machine can have up to four virtual SCSI adapters attached

. This example assumes that VM01 has four virtual SCSI adapters and that, t

herefore, the array contains four elements (counting 0 through 3).

The last command enables the fourth virtual SCSI adapter (object [3] in the

variable array) and specifies that is it shared so that it can be used in

guest clustering.

NOTE: Using the Shared parameter to share a virtual SCSI adapter on a VM is

supported only if the VM is deployed on a Virtual Server or ESX host. The

Shared parameter is not used for a VM a Hyper-V host because a VM on a Hype

r-V host uses iSCSI for shared storage.

REMARKS

For more information, type: "get-help Set-VirtualSCSIAdapter -detailed".

For technical information, type: "get-help Set-VirtualSCSIAdapter -full".

## VM

### DisableUndoDisk-VM

SYNOPSIS

Merges or discards undo disks associated with a virtual machine on a Virtua

l Server host managed by Virtual Machine Manager.

SYNTAX

DisableUndoDisk-VM [-VM] [<String VM>] [-Discard <Boolean>] [-JobVariable <

String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DisableUndoDisk-VM [-VM] [<String VM>] [-JobVariable <String>] [-Merge <Boo

lean>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Merges or discards undo disks associated with a virtual machine on a Virtua

l Server host managed by Virtual Machine Manager.

Virtual Machine Manager provides limited support for the Virtual Server fea

ture called Undo Disks. If the Undo Disks feature is enabled, any changes t

o a virtual machine's configuration or data are stored temporarily in a sep

arate virtual undo disk (.vud) file instead of on the virtual machine's ori

ginal (parent) virtual hard disk file.

When an undo disk exists, you can use the DisableUndoDisk-VM cmdlet to perf

orm the following tasks:

\* Merge undo disks, which updates the virtual machine's original virtual

hard disk file with all of the changes that are stored in the undo disk

file. (In Virtual Server, the "merge" action is known as "commit.")

\* Discard undo disks, which deletes the undo disk without committing any

changes to the virtual machine's original virtual hard disk file, thus

leaving the virtual machine in the same state that it was in before the

undo disk was created.

If the Undo Disks feature is enabled and you take no action, the undo disk

file continues to exist, storing the changes that it contains separately fr

om the virtual machine's original virtual hard disk until you merge or disc

ard those changes.

For information about undo disks, see the Virtual Server 2005 Administrator

's Guide.

For information about the Virtual Machine Manager feature called checkpoint

s, which provides functionality similar to Undo Disks but which is designed

specifically for virtual machines on a host managed by Virtual Machine Man

ager, type:

Get-Help New-VMCheckpoint -detailed

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-Discard <Boolean>

Discards an undo disk associated with a virtual machine.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Merge <Boolean>

Updates the virtual machine's original virtual hard disk file by mergin

g into it all of the changes that are stored in the undo disk file.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Discard undo disks associated with a specific virtual machine on a Virtu

al Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" -and $\_.VirtualizationPla

tform -eq "VirtualServer" }

PS C:\> DisableUndoDisk-VM -VM $VM -Discard

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets all virtual machine objects from the VMM database,

selects the object that represents the virtual machine named VM01 and whose

virtualization platform is Virtual Server, and stores this virtual machine

object in variable $VM. This example assumes that only one virtual machin

e named VM01 exists.

The last command discards the undo disks on VM01.

REMARKS

For more information, type: "get-help DisableUndoDisk-VM -detailed".

For technical information, type: "get-help DisableUndoDisk-VM -full".

### DiscardSavedState-VM

SYNOPSIS

Discards the saved state of virtual machines managed by Virtual Machine Man

ager.

SYNTAX

DiscardSavedState-VM -VM [<String VM>] [-JobVariable <String>] [-PROTipID <

Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Discards the saved state of one or more virtual machines managed by Virtual

Machine Manager. Discarding the saved state of a virtual machine returns i

ts object in a stopped state.

After the saved state of a virtual machine is discarded, the virtual machin

e cannot be resumed in a condition identical to the discarded saved state.

To save the state of a virtual machine, use SaveState-VM.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Discard the saved state of the specified virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> DiscardSavedState-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets all virtual machine objects from the VMM database,

selects the object that represents the virtual machine named VM01, and stor

es this virtual machine object in variable $VM. This example assumes that o

nly one virtual machine named VM01 exists.

The third command discards the saved state of VM01, puts it into a stopped

state, and returns its stopped object to the user.

REMARKS

For more information, type: "get-help DiscardSavedState-VM -detailed".

For technical information, type: "get-help DiscardSavedState-VM -full".

### Get-VM

SYNOPSIS

Gets virtual machine objects from the Virtual Machine Manager database.

SYNTAX

Get-VM [[-Name] <String>] -VMHost [<String Host>] [-VMMServer [<String Serv

erConnection>]] [<CommonParameters>]

Get-VM [[-Name] <String>] [-VMMServer [<String ServerConnection>]] [<Common

Parameters>]

Get-VM [[-Name] <String>] [-ID <Guid>] [-VMMServer [<String ServerConnectio

n>]] [<CommonParameters>]

Get-VM [[-Name] <String>] [-All] [-VMMServer [<String ServerConnection>]] [

<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual machines from the Virtual M

achine Manager database. A virtual machine can be deployed on a virtual mac

hine host or can be stored in the Virtual Machine Manager library.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-All

Retrieves a full list of all subordinate objects independent of the par

ent object. For example, the command Get-VirtualDiskDrive -All retrieve

s all virtual disk drive objects regardless of the virtual machine obje

ct or template object that each virtual disk drive object is associated

with.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all virtual machines and display complete information about each one

.

PS C:\> Get-VM -VMMServer VMMServer1.Contoso.com

Gets from the VMM database on VMMServer1 (in the Contoso.com domain) all ob

jects that represent virtual machines and displays information about these

virtual machines to the user.

2: Get all virtual machines and display information about specific properti

es.

PS C:\> Get-VM -VMMServer VMMServer1.Contoso.com | Format-List -property Na

me, Owner, Description, HostName, OperatingSystem, CPUCount, Memory

Gets from VMMServer1 all objects that represent virtual machines and displa

ys information about the specified properties of these virtual machines to

the user.

3: Get a specific virtual machine by name that is stored on the specified l

ibrary server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VM | where { $\_.Name -eq "VM03" -and $\_.LibraryServer

-eq "FileServer01" } | select Name,LibraryServer,Status

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM03 that is stored on the library server named FileServer 01 and display

s the virtual machine name, the name of the library server, and the status

of the virtual machine.

4: Get all virtual machines on the specified host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VM -VMHost "VMHost04.Contoso.com"

The first command connects to VMMServer1.

The last command gets all objects that represent virtual machines currently

deployed on VMHost04 and displays information about these virtual machines

to the user.

REMARKS

For more information, type: "get-help Get-VM -detailed".

For technical information, type: "get-help Get-VM -full".

### Move-VM

SYNOPSIS

Moves a virtual machine currently stored in the Virtual Machine Manager lib

rary or deployed on a host server to a new location on a host server.

SYNTAX

Move-VM [-VM] [<String VM>] -VMHost [<String Host>] [-JobGroup <Guid>] [-Jo

bVariable <String>] [-Path <String>] [-PROTipID <Guid>] [-RunAsynchronously

] [-StartVMOnTarget] [-UseLAN <Boolean>] [<CommonParameters>]

DETAILED DESCRIPTION

Moves a virtual machine currently stored in the Virtual Machine Manager lib

rary or deployed on a host server to a new location on a host server. Alter

natively, if you want to move a virtual machine from a host and store it in

the library, you must use the Store-VM cmdlet.

You can run the Move-VM cmdlet on a running virtual machine and VMM will at

tempt to shut down the virtual machine before moving the VM. If VMM cannot

use either VMware Live Migration or Windows 2008 Cluster Migration to move

the VM, you must put the virtual machine in a stopped state or saved state

before using the Move-VM cmdlet.

VMM 2008 can use any of the following transfer methods (listed in the order

in which VMM tries to use them):

\* VMWARE LIVE MIGRATION – If a virtual machine deployed on a VMware ESX

host uses shared storage, VMM can use the VMware live migration feature

(also called VMotion) to move the virtual machine to a new host. You

do not need to specify a path for this type of move.

The Move-VM cmdlet can use the VMware VMotion feature to move a virtual

machine from one ESX host to another only if both ESX servers are in

the same Datacenter container on the VMware VirtualCenter Server.

\* WINDOWS 2008 CLUSTER MIGRATION – If a virtual machine is deployed on

a Hyper-V host that is one node on a host cluster, VMM can use

Windows Server 2008 Cluster Migration (sometimes called Quick Migration)

to move the virtual machine to another node in the cluster. You do not

need to specify a path for this type of move.

\* SAN MIGRATION (Fibre Channel, iSCSI, or NPIV) – If the virtual machine

is on a host that is connected to SAN storage, VMM can move that

virtual machine to another host connected to the same SAN. In a SAN

transfer, the target LUNs are remapped from the source host to the

destination host (no files are moved), which is why a SAN transfer is

much faster than moving virtual machine files from one host to another

over a local area network (LAN). VMM can use an NPIV SAN transfer if a

host bus adapter (HBA) with NPIV support is available. You must specify

a path for this type of move.

\* NETWORK MIGRATION – If no faster method is available, VMM uses a network

transfer to move the virtual machine files from one host to another

over the LAN that connects the two hosts. You must specify a path for

this type of move.

The Move-VM cmdlet automatically uses the fastest available transfer type t

o move a VM. If the first method is not appropriate or available for the vi

rtual machine you want to migrate, VMM tries to use the next method, and so

on. If you want to force the use of a network transfer, specify the -UseLA

N parameter.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-StartVMOnTarget

Specifies that a virtual machine starts as soon as it reaches its desti

nation host.

-UseLAN <Boolean>

Forces a transfer over the local area network (LAN) even if a faster tr

ansfer mechanism, such as a storage area network (SAN) transfer, is ava

ilable.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Move a virtual machine from the library to a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" -and $\_.LibraryServer -eq

"FileServer01.Contoso.com" }

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> Move-VM -VMHost $VMHost -VM $VM -Path "D:\VMs"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets from the VMM database the object that represents th

e virtual machine named VM01 (which is currently stored in the VMM library

on the library server named FileServer01) and stores the virtual machine ob

ject in variable $VM. This example assumes that only one virtual machine na

med VM01 is currently stored on FileServer01, and that, therefore, Get-VM r

etrieves only one object.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in variable $VMHost.

The last command moves the virtual machine from its current location in the

library to the location D:\VMs on the host. The command automatically uses

the fastest available transfer type. When the command completes, it return

s information about the moved virtual machine to the screen, including the

MostRecentTask, which contains "Move virtual machine from <YourLibraryServe

rName> to <YourVMHostName>."

NOTE: If you want to move a virtual machine that is currently deployed on a

host to another location and neither VMotion nor Quick Migration is availa

ble, the virtual machine must be turned off or put in a saved state before

it can be moved. If you want to move a virtual machine that is currently de

ployed on a host into the VMM library, you must use the Store-VM cmdlet.

2: Move a virtual machine from the library to a host asynchronously.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM02" -and $\_.LibraryServer -eq

"FileServer02.Contoso.com" }

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> Move-VM -VMHost $VMHost -VM $VM -Path "D:\VMs" -RunAsynchronously -

JobVariable "MyJob"

PS C:\> $MyJob

The first four commands in this example are identical to the commands in ex

ample 1, except that when the fourth command moves the virtual machine from

its current location to D:\VMs on VMHost02, it also uses the RunAsynchrono

usly parameter to return control to the command shell immediately, and it u

ses the JobVariable parameter to track job progress and store a record of i

ts progress in variable MyJob. For the JobVariable parameter, you do not us

e the dollar sign ($) when the variable is created.

The last command displays the contents of $MyJob, which includes a descript

ion of the move job, its status ("Running"), and other information.

3: Move a virtual machine from the library to a host by forcing a LAN trans

fer.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03.Contoso.com"

PS C:\> Move-VM -VMHost $VMHost -VM "VM03" -Path "D:\VMs" -UseLAN

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost03

and stores the host object in $VMHost.

The last command moves the virtual machine VM03 (by specifying its name) fr

om its current location in the library to D:\VMs on VMHost03, it uses the U

seLAN parameter to specify that the transfer uses a network transfer (even

if faster transfer mechanisms are available). This example assumes that VM0

3 was stored in the VMM library before the move operation started.

4: Move a virtual machine between hosts by using VMware VMotion.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM04" | where {$\_.VMHost.Name -eq "10.199.53.11

"}

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "10.199.53.12"}

PS C:\> Move-VM -VM $VM -VMHost $VMHost

PS C:\> Move-VM -VM $VM -VMHost $VMHost -Path "[Storage2]\VM04\VM04.vmx"

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM04 (which is currently running on the VMWare ESX host identified by its

IP address, 10.199.53.11) and stores the virtual machine object in $VM.

The third command gets the object that represents the ESX host identified

by IP address10.100.53.12 and stores the host object in $VMHost.

In the last command, the Move-VM cmdlet uses VMware VMotion to move the vir

tual machine from its current ESX host to the other ESX host.

IMPORTANT: The Move-VM cmdlet can use the VMware VMotion feature to move a

virtual machine from one ESX host to another only if both ESX servers are i

n the same Datacenter container on the VMware VirtualCenter Server.

5: Move a highly available VM between nodes in a host cluster by using Hype

r-V quick migration.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "HAVM05" | where {$\_.VMHost.Name -eq "VMHVHostNo

de05A.Contoso.com"}

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "VMHVHostNode05B.Contoso.

com"}

PS C:\> Move-VM -VM $VM -VMHost $VMHost

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d HAVM05 (which is currently running on the Hyper-V host named VMHVHostNode

05A) and stores the virtual machine object in $VM. This example assumes tha

t HAVM05 is a highly available virtual machine and that VMHVHostNode05A and

VMHVHostNode05B are nodes in a Hyper-V host cluster.

The third command gets the object that represents the host named VMHVHostNo

de05B and stores the host object in $VMHost.

In the last command, the Move-VM cmdlet uses Windows Server 2008 quick migr

ation to move the virtual machine from VMHVHostNode05A to VMHVHostNode05B.

REMARKS

For more information, type: "get-help Move-VM -detailed".

For technical information, type: "get-help Move-VM -full".

### New-VM

SYNOPSIS

Creates a virtual machine to be managed by Virtual Machine Manager.

SYNTAX

New-VM [-Name] <String> -Path <String> -VM [<String VM>] -VMHost [<String H

ost>] [-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-DelayStart

<Int32>] [-Description <String>] [-HardwareProfile <HardwareProfile>] [-Job

Group <Guid>] [-JobVariable <String>] [-LimitCPUFunctionality <Boolean>] [-

MemoryMB <Int32>] [-OperatingSystem <OperatingSystem>] [-Owner <String>] [-

PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSystem] [-RunAsUserCreden

tial <PSCredential>] [-RunAsynchronously] [-StartAction <String>] [-StartVM

] [-StopAction <String>] [-UseHardwareAssistedVirtualization] [<CommonParam

eters>]

New-VM [-Name] <String> -Path <String> -VMHost [<String Host>] [-CPUCount <

Int32>] [-CPUType [<ProcessorType String>]] [-DelayStart <Int32>] [-Descrip

tion <String>] [-HardwareProfile <HardwareProfile>] [-JobGroup <Guid>] [-Jo

bVariable <String>] [-LimitCPUFunctionality <Boolean>] [-MemoryMB <Int32>]

[-OperatingSystem <OperatingSystem>] [-Owner <String>] [-PROTipID <Guid>] [

-RelativeWeight <Int32>] [-RunAsSystem] [-RunAsUserCredential <PSCredential

>] [-RunAsynchronously] [-StartAction <String>] [-StartVM] [-StopAction <St

ring>] [-UseHardwareAssistedVirtualization] [-VMMServer [<String ServerConn

ection>]] [<CommonParameters>]

New-VM [-Name] <String> -LibraryServer [<String LibraryServer>] -SharePath

<String> -VirtualHardDisk <VirtualHardDisk> [-CPUCount <Int32>] [-CPUType [

<ProcessorType String>]] [-DelayStart <Int32>] [-Description <String>] [-Ha

rdwareProfile <HardwareProfile>] [-JobGroup <Guid>] [-JobVariable <String>]

[-LimitCPUFunctionality <Boolean>] [-MemoryMB <Int32>] [-OperatingSystem <

OperatingSystem>] [-Owner <String>] [-PROTipID <Guid>] [-RelativeWeight <In

t32>] [-RunAsSystem] [-RunAsUserCredential <PSCredential>] [-RunAsynchronou

sly] [-StartAction <String>] [-StopAction <String>] [-UseHardwareAssistedVi

rtualization] [<CommonParameters>]

New-VM [-Name] <String> -LibraryServer [<String LibraryServer>] -SharePath

<String> -VM [<String VM>] [-CPUCount <Int32>] [-CPUType [<ProcessorType St

ring>]] [-DelayStart <Int32>] [-Description <String>] [-HardwareProfile <Ha

rdwareProfile>] [-JobGroup <Guid>] [-JobVariable <String>] [-LimitCPUFuncti

onality <Boolean>] [-MemoryMB <Int32>] [-OperatingSystem <OperatingSystem>]

[-Owner <String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSyst

em] [-RunAsUserCredential <PSCredential>] [-RunAsynchronously] [-StartActio

n <String>] [-StopAction <String>] [-UseHardwareAssistedVirtualization] [<C

ommonParameters>]

New-VM [-Name] <String> -LibraryServer [<String LibraryServer>] -SharePath

<String> [-CPUCount <Int32>] [-CPUType [<ProcessorType String>]] [-DelaySta

rt <Int32>] [-Description <String>] [-HardwareProfile <HardwareProfile>] [-

JobGroup <Guid>] [-JobVariable <String>] [-LimitCPUFunctionality <Boolean>]

[-MemoryMB <Int32>] [-OperatingSystem <OperatingSystem>] [-Owner <String>]

[-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSystem] [-RunAsUserCre

dential <PSCredential>] [-RunAsynchronously] [-StartAction <String>] [-Stop

Action <String>] [-UseHardwareAssistedVirtualization] [-VMMServer [<String

ServerConnection>]] [<CommonParameters>]

New-VM [-Name] <String> -Path <String> -Template [<Template String>] -VMHos

t [<String Host>] [-AdminPasswordCredential <PSCredential>] [-AnswerFile <S

cript>] [-ComputerName <String>] [-CPUCount <Int32>] [-CPUType [<ProcessorT

ype String>]] [-DelayStart <Int32>] [-Description <String>] [-FullName <Str

ing>] [-GuestOSProfile [<GuestOSProfile String>]] [-GuiRunOnceCommands <Str

ing[]>] [-HardwareProfile <HardwareProfile>] [-JobGroup <Guid>] [-JobVariab

le <String>] [-JoinDomain <String>] [-JoinDomainCredential <PSCredential>]

[-JoinWorkgroup <String>] [-LimitCPUFunctionality <Boolean>] [-MemoryMB <In

t32>] [-OperatingSystem <OperatingSystem>] [-OrgName <String>] [-Owner <Str

ing>] [-ProductKey <String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [

-RunAsSystem] [-RunAsUserCredential <PSCredential>] [-RunAsynchronously] [-

SelfServiceRole <UserRole>] [-StartAction <String>] [-StartVM] [-StopAction

<String>] [-TimeZone <Int32>] [-UseHardwareAssistedVirtualization] [<Commo

nParameters>]

New-VM [-Name] <String> -Path <String> -VirtualHardDisk <VirtualHardDisk> -

VMHost [<String Host>] [-CPUCount <Int32>] [-CPUType [<ProcessorType String

>]] [-DelayStart <Int32>] [-Description <String>] [-HardwareProfile <Hardwa

reProfile>] [-JobGroup <Guid>] [-JobVariable <String>] [-LimitCPUFunctional

ity <Boolean>] [-MemoryMB <Int32>] [-OperatingSystem <OperatingSystem>] [-O

wner <String>] [-PROTipID <Guid>] [-RelativeWeight <Int32>] [-RunAsSystem]

[-RunAsUserCredential <PSCredential>] [-RunAsynchronously] [-StartAction <S

tring>] [-StartVM] [-StopAction <String>] [-UseHardwareAssistedVirtualizati

on] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual machine to be managed by Virtual Machine Manager. You can

create a virtual machine from an existing stopped virtual machine deployed

on a host, from an existing virtual machine stored in the Virtual Machine

Manager library, from a virtual machine template, from an existing virtual

hard disk (that already contains an operating system), or from a blank virt

ual hard disk.

You can, for example, create a new virtual machine from an existing hard di

sk that contains a third-party operating system, such as Linux.

For a new (or moved) virtual machine deployed on a host, the default locati

on for the virtual machine files in the file system is:

\* Virtual machine deployed on a Hyper-V host:

<C>:\ProgramData\Microsoft\Windows\Hyper-V

\* Virtual machine deployed on a Virtual Server host:

<C>:\Documents and Settings\All Users\Documents\Shared Virtual Machines\

\* Virtual machine deployed on a VMware ESX host:

No default path exists, so you must specify the path when you create

the new virtual machine.

As an alternative to using New-VM, you can also use the following cmdlets t

o create a new virtual machine:

\* New-P2V - creates a new virtual machine from an existing physical

machine (a P2V conversion).

\* New-V2V - creates a new virtual machine from an existing virtual

machine; for example, from a virtual machine created in VMWare

(a V2V conversion).

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VirtualHardDisk <VirtualHardDisk>

Specifies a virtual hard disk object.

-VM [<String VM>]

Specifies a virtual machine object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-AdminPasswordCredential <PSCredential>

Specifies the password for the local Administrator account. Specifying

a password (on a new or existing template, on a new or existing guest

operating system profile, or on a new virtual machine) overrides any ex

isting Administrator password.

-AnswerFile <Script>

Specifies a script object stored in the VMM library to use as an answer

file. The name of the answer file script depends on the operating syst

em that you want to install on a virtual machine:

ANSWER FILE GUEST OS TO INSTALL ON VM

----------- -------------------------

Sysprep.inf Windows XP, Windows Server 2000, or Windows Server 2003

Unattend.xml Windows Vista or Windows Server 2008

Note: In VMM 2007, this parameter was named SysPrepFile.

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-DelayStart <Int32>

Specifies the number of seconds to wait after the virtualization servic

e starts before automatically starting a virtual machine. Used to stagg

er the startup time of multiple virtual machines to help reduce the dem

and on the physical computer’s resources. A typical setting might be 30

to 60 seconds.

TYPE OF HOST MAXIMUM CONFIGURABLE DELAY

------------ --------------------------------

Hyper-V 1000000000 seconds (277777 hours)

Virtual Server 86400 seconds (24 hours)

VMware ESX 65535 seconds (18 hours)

-Description <String>

Specifies a description for the specified object.

-FullName <String>

Specifies the name of the person in whose name a virtual machine is reg

istered.

-GuestOSProfile [<GuestOSProfile String>]

Specifies a guest operating system profile object.

-GuiRunOnceCommands <String[]>

Specifies one or more commands to add to the [GuiRunOnce] section of an

unattended answer file (such as SysPrep.inf or Unattend.xml). Use sing

le quotes around each string enclosed in double quotes.

Example:

-GuiRunOnceCommands '"C:\APF\APFPostSysPrepCopy.cmd PARAMS1"', '"C:\APF

\APFPostSysPrepCopy.cmd PARAMS1"'

For information about how Windows PowerShell uses quotes, type: Get-Hel

p about\_Quoting\_Rules

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-JoinDomain <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the do

main to which you want to join a virtual machine. You can use this para

meter to override the existing value on a template or on a guest operat

ing system profile. You can join a VM to a domain only if a virtual net

work adapter is configured for the VM.

-JoinDomainCredential <PSCredential>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the user name and

password of an account with permission to join a virtual machine to the

domain. A limited rights account should be used for joining machines t

o the domain.

-JoinWorkgroup <String>

Specifies (on a new or existing template, on a new or existing guest op

erating system profile, or on a new virtual machine) the name of the wo

rkgroup to which you want to join a virtual machine. You can use this p

arameter to override the existing value on a template or on a guest ope

rating system profile.

-LimitCPUFunctionality <Boolean>

Enables running an older operating system (such as Windows NT 4.0) on a

virtual machine deployed on a Hyper-V host or on a VMware ESX host by

providing only limited CPU functionality for the virtual machine.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-OrgName <String>

Specifies the name of the organization of the person in whose name a vi

rtual machine is registered.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-ProductKey <String>

Specifies the product key to use for the operating system to be install

ed on a virtual machine. The product key is a 25-digit number that iden

tifies the product license.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RunAsSystem

Specifies that a virtual machine on a Virtual Server host will run unde

r the local system account. If specified, Virtual Server will not autom

atically start the virtual machine when the Virtual Server service star

ts. (This parameter does not apply to virtual machines on Hyper-V or VM

ware ESX hosts because Hyper-V and VMware run a virtual machine under t

he local system account by default; you cannot change this setting on t

hose virtualization platforms.)

-RunAsUserCredential <PSCredential>

Specifies the guest account (domain\account) that a virtual machine on

a Virtual Server host runs under. If specified, Virtual Server will aut

omatically start a virtual machine when the Virtual Server service star

ts. For enhanced security, create a special account with limited permis

sions:

FILE TYPE MINIMUM REQUIRED PERMISSIONS FOR GUEST ACCOUNT

----------- ----------------------------------------------

.vmc file Read Data, Write Data, Execute File

.vmc folder List Folder, Write/Create File (required to save VM state)

.vhd file Read Data, Read Attributes, Read Extended Attributes,

Write Data

.vnc file Execute File, Read Data, Read Attributes, Read

(required if VM connects to a virtual network)

Note: This parameter does not apply to virtual machines on Hyper-V or V

Mware ESX hosts.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SelfServiceRole <UserRole>

Specifies the self-service role with permission to access the virtual m

achine.

-StartAction <String>

Specifies the behavior of a virtual machine when the virtualization ser

vice (Hyper-V, Virtual Server, or VMware) starts. To specify that a vir

tual machine deployed on a Virtual Server host starts automatically, us

e the -RunAsUserCredential parameter to specify an account with appropr

iate permissions (otherwise, the StartAction reverts to NeverAutoTurnOn

VM).

Valid values: AlwaysAutoTurnOnVM, NeverAutoTurnOnVM, TurnOnVMIfRunningW

henVSStopped

-StartVM

Specifies that the virtual machine starts when it arrives at the destin

ation host.

-StopAction <String>

Specifies the behavior of the virtual machine when the virtualization s

ervice (Hyper-V, Virtual Server, or VMware) stops.

Valid values: SaveVM, TurnOffVM, ShutdownGuestOS

-TimeZone <Int32>

Specifies a number (an "index") that identifies a geographical region t

hat shares the same standard time. For a list of time zone indexes, see

"Microsoft Time Zone Index Values" at: http://go.microsoft.com/fwlink/

?LinkId=120935. If no time zone is specified, the default time zone use

d for a virtual machine is the same time zone setting that is on the vi

rtual machine host.

Example: To specify the GMT Standard Time zone, type: -TimeZone 085

-UseHardwareAssistedVirtualization

Specifies that, for a virtual machine deployed on a Virtual Server host

, hardware-assisted virtualization is used if it is available (when set

to TRUE). The Virtual Server host must support AMD Virtualization (AMD

-V) or Intel Virtualization Technology (Intel-VT) hardware virtualizati

on. This parameter does not apply to virtual machines on Hyper-V hosts

or VMware ESX hosts.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a virtual machine from a virtual hard disk and deploy it on a hos

t.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VHD = Get-VirtualHardDisk | where {$\_.Name -eq "Blank Disk - Large

"}

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "VMHost01.Contoso.com"}

PS C:\> New-VM -Name "VM01" -VirtualHardDisk $VHD -VMHost $VMHost -Path "C:

\MyVMs" –RunAsynchronously

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the virtual hard disk named "Blank Disk - Large" fr

om the VMM library and stores the virtual hard disk object in variable $VHD

.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in variable $VMHost.

The last command creates a virtual machine named VM01 from the virtual hard

disk called "Blank Disk - Large" and deploys the new virtual machine on th

e C: drive of VMHost01 in the MyVMs folder. The -RunAsynchronously paramete

r returns control to the shell immediately (before the command completes).

2: Create a virtual machine from a template and deploy it on a host.

PS C:\> $Template = Get-Template -VMMServer "VMMServer1.Contoso.com" | wher

e {$\_.Name -eq "Gold Windows Server 2003 R2"}

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "VMHost02.Contoso.com"}

PS C:\> New-VM -Template $Template -Name "VM02" -VMHost $VMHost -Path "C:\D

ocuments and Settings\All Users\Documents\Shared Virtual Machines\" -RunAsy

nchronously -ComputerName "Server-01" -FullName "Rachel Valdez" -OrgName "C

ontoso Corp" -ProductKey "XXXXX-XXXXX-XXXXX-XXXXX-XXXXX"

The first command gets the object that represents the template named "Gold

Windows Server 2003 R2" and stores the template object in $Template.

The second command gets the object that represents the host named VMHost02

and stores the host object in $VMHost.

The last command uses the New-VM cmdlet to perform the following tasks:

\* Create a virtual machine from the "Gold Windows Server 2003 R2"

template.

\* Name the new virtual machine VM02.

\* Deploy the new virtual machine on VMHost02 and specify the path

where the virtual machine files will be stored on VMHost02:

C:\Documents and Settings\All Users\Documents\Shared Virtual Machines\

\* Specify that the command run asynchronously to return control

to the shell immediately (before the command completes).

\* Sets the following information for VM02:

- Server-01.Contoso.com (the computer name and domain for

VM02 that appears on the Computer Name tab of System Properties)

- Rachel Valdez (the name of the person that appears on the General

tab of System Properties)

- ContosoCorp (the organization name that appears on the General tab

of System Properties)

- XXXXX-XXXXX-XXXXX-XXXXX-XXXXX (the product key - substitute

your 25-digit product key number for the X's)

3: Clone a virtual machine from an existing virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where {$\_.Name -eq "VM03"}

PS C:\> $VMHost = Get-VMHost | where { $\_.Name -eq "VMHost03.Contoso.com" }

PS C:\> if($VM.Status -eq "PowerOff"){New-VM -Name "VM04" -VM $VM -VMHost $

VMHost -Path "C:\MyVMs" –RunAsynchronously}

The first command connects to VMMServer1.

The second command gets the object that represents the VM named VM03 and st

ores the virtual machine object in $VM.

The third command gets the object that represents the host named VMHost03 a

nd stores the host object in $VMHost.

The last command first checks if the VM is in a powered off state. If the V

M is powered off, the command creates a virtual machine named VM04 from VM0

3 and deploys the new virtual machine on the C:\ drive of VMHost03 in the M

yVMs folder. The -RunAsynchronously parameter returns control to the shell

immediately (before the command completes).

4. Create a virtual machine from a virtual machine stored in the library.

PS C:\> Get-VMMServer "VMMServer1.contoso.com"

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "VMHost04"}

PS C:\> $VM = Get-VM -Name "StoredVM" | where {$\_.LibraryServer.Name -eq "F

ileServer01"} | where {$\_.Location -eq "\\FileServer01\MSSCVMMLibrary\Store

dVM\"}

PS C:\> New-VM -VM $VM -Name "VM04" -Description "New VM from VM stored in

Library" -Owner "Contoso\VMMAdmin" -VMHost $VMHost -Path "C:\MyVMs" -RunAsy

nchronously -RunAsSystem -StartAction NeverAutoTurnOnVM -UseHardwareAssiste

dVirtualization $FALSE -StopAction SaveVM -MemoryMB 1024

The first command connects to VMMServer1.

The second command gets the object for the host named VMHost04 and stores t

he host object in $VMHost.

The third command gets the object for the VM named StoredVM, which is locat

ed at the specified path on library server FileServer01, and stores the vir

tual machine object in $VM.

The last command uses the New-VM cmdlet to perform the following operations

:

\* Creates a new VM named VM04 from StoredVM and provides

VM04 with 1024 MB of memory.

\* Deploys VM04 on host VMHost04 at the specified path.

\* Provides a description and specifies VMMAdmin as the owner.

\* Uses the -RunAsynchronously parameter to return control to the shell

immediately (before the command completes).

\* Uses the -RunAsSystem parameter to specify that the VM runs under the

local system account (it is only necessary to specify this for a VM on

a Virtual Server host because Hyper-V and VMware run a VM under the

local system account by default).

\* Specifies that hardware assisted virtualization is not used.

\* Specifies that the start action for the new VM is set to never turn on

automatically and that the stop action is set to save the VM.

5. Create a highly available virtual machine.

PS C:\> Get-VMMServer "VMMServer1.Contoso.com"

PS C:\> $VMGuid = [System.Guid]::NewGuid().ToString()

PS C:\> $VMName = "HAVM01"

PS C:\> New-VirtualNetworkAdapter -JobGroup $VMGuid -PhysicalAddressType Dy

namic -VLANEnabled $FALSE

PS C:\> New-VirtualDVDDrive -JobGroup $VMGuid -Bus 1 -LUN 0

PS C:\> New-HardwareProfile -Owner "Contoso\VMMAdmin" -Name "HWProfile" -CP

UCount 1 -MemoryMB 512 -HighlyAvailable $TRUE -NumLock $FALSE -BootOrder "C

D", "IdeHardDrive", "PxeBoot", "Floppy" -LimitCPUFunctionality $FALSE -JobG

roup $VMGuid

PS C:\> New-VirtualDiskDrive -IDE -Bus 0 -LUN 0 -JobGroup $VMGuid -Size 409

60 -Dynamic -Filename "VMM-HAVM\_disk\_1.vhd"

PS C:\> $VMHost = Get-VMHost | where {$\_.Name -eq "VMM-HVNode2.SCVMM.Contos

o.com"}

PS C:\> $HardwareProfile = Get-HardwareProfile | where {$\_.Name -eq "HWProf

ile"}

PS C:\> $OperatingSystem = Get-OperatingSystem | where {$\_.Name -eq "64-bit

edition of Windows Server 2008 Datacenter"}

PS C:\> New-VM -Name $VMName -Description "" -Owner "Contoso\VMMAdmin" -VMH

ost $VMHost -Path "P:\" -HardwareProfile $HardwareProfile -JobGroup $VMGuid

-OperatingSystem $OperatingSystem -RunAsynchronously -RunAsSystem -StartAc

tion NeverAutoTurnOnVM -StopAction SaveVM -UseHardwareAssistedVirtualizatio

n $FALSE

The first command connects to VMMServer1.

The second command creates a new GUID string and stores it in $VMGuid. This

GUID is a job group ID that functions as an identifier that groups subsequ

ent commands that include this identifier into a single job group.

The third command stores a string ("HAVM01") used to name the new VM in $VM

Name.

The fourth command will create a new virtual network adapter with a dynamic

MAC address and with VLAN disabled, but uses the job group ID to specify t

hat the network adapter is not created until just before the New-VM cmdlet

(in the last command) runs.

The fifth command will create a new IDE virtual DVD drive connected to the

second channel and the first slot, but uses the job group ID to specify tha

t the virtual DVD drive is not created until just before the New-VM cmdlet

(in the last command) runs.

The sixth command will create a new hardware profile and specifies values f

or the profile name, owner, CPU count, memory, and bootorder; disables NumL

ock as well as limited CPU functionality (which is not needed because this

is VM will not have an older operating system); and designates that the VM

created by using this hardware profile will be a highly available VM. The c

ommand uses the job group ID to specify that the hardware profile is not cr

eated until just before the New-VM cmdlet (in the last command) runs.

The seventh command will create a new IDE virtual disk drive with a storage

capacity of 40 GB on the first channel and first slot, but uses the job gr

oup ID to specify that the new virtual disk drive is not created until just

before the New-VM cmdlet (in the last command) runs.

The eighth command gets a VM host object by name (VMM-HVNode2) and stores t

he host object in $VMHost. This host is one node of a host cluster that is

managed by VMM.

The ninth command gets the object that represents the hardware profile spec

ified in the fifth command by name, and stores the return value in $Hardwar

eProfile.

The tenth command gets an operating system object by name and stores the re

turn value in $OperatingSystem.

The last command uses the New-VM cmdlet and the job group ID to create a ne

w VM named HAVM01 by using the objects obtained or created in the preceding

commands. The command also performs the following operations:

\* Uses the Path parameter to specify the location where the VM is stored;

this location must be a cluster-migratable LUN.

\* Uses the RunAsynchronously parameter to return control to the shell

immediately (before the command completes)

\* Uses the -RunAsSystem parameter to specify that the VM runs under

the local system account (it is only necessary to specify this for a

VM on a Virtual Server host because Hyper-V and VMware run a VM under

the local system account by default).

\* Specifies that the VM is not started automatically when the host

starts and that the VM is put into a saved state when the

virtualization service stops.

\* Specifies that hardware-assisted virtualization is not used.

REMARKS

For more information, type: "get-help New-VM -detailed".

For technical information, type: "get-help New-VM -full".

### Refresh-VM

SYNOPSIS

Refreshes the properties of a virtual machine so that the Virtual Machine M

anager Administrator Console displays updated information about the virtual

machine.

SYNTAX

Refresh-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>] [

-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Refreshes the properties of a virtual machine so that the Virtual Machine M

anager Administrator Console displays updated information about the virtual

machine. The updated properties include Name, Location, Status, OperatingS

ystem, and other properties.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Refresh information about a specific virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> Refresh-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The last command refreshes the properties Name, Location, Status, Operating

System, and VMAdditions for VM01. After this command completes successfully

, current information about this virtual machine will appear in the Adminis

trator Console.

2: Refresh all virtual machines on each host whose name matches the specifi

ed string.

PS C:\> $VMs = Get-VM -VMMServer VMMServer1.Contoso.com | where { $\_.VMHost

.Name -match "VMM" }

PS C:\> $VMs | Refresh-VM

The first command gets all objects from VMMServer1 that represent virtual m

achines deployed on hosts whose name contains the string "VMM" and stores t

he virtual machine objects in $VMs.

The second command refreshes the properties of each virtual machine object

in $VMs.

REMARKS

For more information, type: "get-help Refresh-VM -detailed".

For technical information, type: "get-help Refresh-VM -full".

### Register-VM

SYNOPSIS

Registers an existing virtual machine with Virtual Machine Manager that, cu

rrently, is not registered with the virtualization platform (Virtual Server

, Hyper-V, or VMware) of any host managed by Virtual Machine Manager and is

not stored in the Virtual Machine Manager library.

SYNTAX

Register-VM [-VMHost] [<String Host>] [-Path] <String> [-JobGroup <Guid>] [

-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonPara

meters>]

DETAILED DESCRIPTION

Registers an existing virtual machine with Virtual Machine Manager that, cu

rrently, is not registered with the virtualization platform (Virtual Server

, Hyper-V, or VMware) of any host managed by Virtual Machine Manager, and i

s not stored in the Virtual Machine Manager library. If virtual machine fil

es are stored in the Virtual Machine Manager library, you do not need to re

gister the virtual machine before you deploy it on a host.

The configuration files for the virtual machine that you want to register m

ust be stored either in the file system on the host on which you want to de

ploy the virtual machine or stored on shared storage (such as SAN) availabl

e to this host:

\* HYPER-V HOST – You can register a virtual machine for a Hyper-V host

if the configuration files for that virtual machine are stored in a

folder on the host's file system or on shared storage, and an export

of the virtual machine was created using the "Export" function in the

Hyper-V Manager console.

The path must specify a folder. Example:

-Path "D:\MyHyperVFolderForVMs"

\* VIRTUAL SERVER HOST – You can register a virtual machine for a Virtual

Server host if the configuration file for that virtual machine is

stored on the host's file system or on shared storage. No separate

"export" step is required.

The path must specify the folder and the configuration file. Example:

-Path "D:\MyVirtualServerFolderForVMs\MyVM.vmc"

\* VMWARE ESX HOST - You can register a virtual machine for a VMware ESX

host if the VMware virtual machine configuration file (a .vmx file)

is stored on the host's file system or on shared storage. No separate

"export" step is required.

The path must specify the folder and the configuration file. Example:

-Path [MyDatastore]\MyVMwareFolderForVMs\MyVM.vmx

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Register an existing virtual machine on a Virtual Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VSHost01"

PS C:\> Register-VM -VMHost $VMHost -Path "D:\MyVirtualServerFolderForVMs\V

M01.vmc"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents a Virtual Server host ca

lled VSHost01 and stores the host object in variable $VMHost.

The last command adds the existing virtual machine on VSHost01 to the VMM d

atabase by specifying the path to the virtual machine's virtual machine con

figuration file (VM01.vmc).

2: Register an existing virtual machine on a Hyper-V host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "HVHost02"

PS C:\> Register-VM –VMHost $VMHost –Path “D:\MyHyperVFolderForVMs”

The first command connects to VMMServer1.

The second command gets the object that represents a Hyper-V host called HV

Host02 and stores the host object in $VMHost.

The last command adds the existing virtual machine on HVHost02 to VMM by sp

ecifying the path to the folder that contains the virtual machine's virtual

machine configuration file.

IMPORTANT: If the virtual machine is on a Hyper-V host, you specify the pat

h to the folder (as shown here). However, if the virtual machine is on a Vi

rtual Server host or on a VMware ESX host, you specify the path to the fold

er and file (as shown in examples 1 and 3).

3: Register an existing virtual machine on a VMware ESX host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "ESXHost03"

PS C:\> Register-VM –VMHost $VMHost –Path "[storage1]\MyVMwareFolderForVMs\

MyVM.vmx"

The first command connects to VMMServer1.

The second command gets the object that represents a VMware ESX host called

ESXHost03 and stores the host object in $VMHost.

The last command adds an existing virtual machine on ESXHost03 to VMM by sp

ecifying the path to the virtual machine's virtual machine configuration fi

le (MyVM.vmx).

REMARKS

For more information, type: "get-help Register-VM -detailed".

For technical information, type: "get-help Register-VM -full".

### Remove-VM

SYNOPSIS

Removes a virtual machine object from Virtual Machine Manager.

SYNTAX

Remove-VM [-VM] [<String VM>] [-Confirm] [-Force] [-JobVariable <String>] [

-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes an object that represents a virtual machine deployed on a host or s

tored on a library server from Virtual Machine Manager.

The Remove-VM cmdlet deletes the virtual machine record from the Virtual Ma

chine Manager database, deletes all files associated with the virtual machi

ne, and removes the virtual machine from the host on which it is deployed o

r from the library server on which it is stored.

If a folder on a host was created for this virtual machine by Virtual Machi

ne Manager (rather than by Hyper-V, Virtual Server, or VMware) and if that

folder contains no other virtual machines or other data, you can use the fi

le system to delete the folder after you have removed the virtual machine.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-Confirm

Prompts for confirmation before running the command.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific virtual machine deployed on a host and delete the asso

ciated files from the host.

PS C:\> $VM = Get-VM -VMMServer VMMServer1.Contoso.com | where { $\_.VMHost.

Name -eq "VMHost01.Contoso.com" -and $\_.Name -eq "VM01" }

PS C:\> Remove-VM -VM $VM

The first command gets all objects that represent virtual machines from the

VMM database provided by VMMServer1, selects from the results the virtual

machine deployed on VMHost01 named VM01, and stores the virtual machine ob

ject in variable $VM.

The second command removes the object that represents VM01 from the VMM dat

abase and deletes the corresponding virtual machine files from the file sys

tem on its host.

2: Remove all virtual machines with names that include a specific string fr

om the database and from their hosts.

PS C:\> $MyVMs = @(Get-VM -VMMServer VMMServer1.Contoso.com | where { $\_.Na

me -Match "VM0" } )

PS C:\> $MyVMs | Remove-VM

The first command gets all objects that represent any virtual machines depl

oyed on any host whose name includes the string "VM0" (as in VM01, VM02, an

d so on) and stores these virtual machine objects in $MyVMs. Using the "@"

symbol and parentheses ensures that the command stores the results in an ar

ray (in case the command returns a single object or a null value).

The second command removes each virtual machine object in $MyVMs from the V

MM database and deletes the corresponding virtual machine files from the fi

le system on each host.

3: Remove from the library the object that represents a specific virtual ma

chine stored on a VMM library server.

PS C:\> $VM = Get-VM -VMMServer VMMServer1.Contoso.com | where { $\_.Library

Server.Name -eq "FileServer01.Contoso.com" -and $\_.Name -eq "VM03" }

PS C:\> Remove-VM -VM $VM

The first command gets the object that represents the virtual machine named

VM03 (which is stored on the library server named FileServer01) and stores

the virtual machine object in $VM. This example assumes that only one virt

ual machine named VM03 exists.

The second command removes the object that represents VM03 from the library

and deletes the corresponding virtual machine files from the file system o

n the library server.

4: Remove multiple stored virtual machines from the VMM library and from th

e library server.

PS C:\> $MyVMs = Get-VM -VMMServer VMMServer1.Contoso.com | where { $\_.Libr

aryServer.Name -eq "FileServer01.Contoso.com" -and $\_.Name -match “VM0” }

PS C:\> $MyVMs | Remove-VM -Confirm

The first command gets all objects that represent virtual machines whose na

mes include the string "VM0" (as in VM01, VM02, and so on) and that are sto

red stored on the library server named FileServer01. The command stores the

matching virtual machine objects in $MyVMs (an object array).

The second command passes each virtual machine object in $MyVMs to the Remo

ve-VM cmdlet, which removes each object from the library and deletes the co

rresponding virtual machine files from the file system on the library serve

r. The Confirm parameter prompts you to confirm whether you want to delete

these virtual machines.

REMARKS

For more information, type: "get-help Remove-VM -detailed".

For technical information, type: "get-help Remove-VM -full".

### Repair-VM

SYNOPSIS

Repairs a virtual machine on a host managed by Virtual Machine Manager if t

he virtual machine is in a failed state.

SYNTAX

Repair-VM [-VM] [<String VM>] -Undo <Boolean> [-JobVariable <String>] [-PRO

TipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Repair-VM [-VM] [<String VM>] -Dismiss <Boolean> [-JobVariable <String>] [-

PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

Repair-VM [-VM] [<String VM>] -Retry <Boolean> [-JobVariable <String>] [-PR

OTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Repairs a virtual machine on a host managed by Virtual Machine Manager if t

he virtual machine is in a failed state. A virtual machine can be in one of

four types of failed state:

- Creation Failed

- Migration Failed

- Update Failed

- Deletion Failed

You can use this command to repair a failure as follows:

\* RETRY. You can use the Retry option, which will attempt to perform

the failed job again.

\* UNDO. You can use the Undo option, which will attempt to undo any

changes made to the virtual machine object and restore it to a healthy

state. For example, if a Move-VM job fails, using the Undo option

attempts to move the virtual machine back to its previous host.

\* DISMISS. You can use the Dismiss option, which dismisses the

failed job and refreshes the virtual machine object based on its

current state. If you manually fix a failure (for example, by

manually moving the .vhd and .vmc files to a new host after a

Move-VM failure), you can use the Dismiss option to refresh the

data for the virtual machine in the Virtual Machine Manager

database. However, using the Dismiss option might return the

object to the failed state.

When you run Repair-VM, you can specify only one type of action (Retry, Und

o, or Dismiss) at a time.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-Dismiss <Boolean>

Dismisses the error on an object and refreshes that object. If the erro

r reappears, refreshing does not solve the problem and you must fix the

error.

-Retry <Boolean>

Retries the last task that failed on a VMM object in an attempt to comp

lete the task successfully.

-Undo <Boolean>

Cancels the last job run on a VMM object and reverses any changes that

were made. This parameter is available only if the most recent job fail

ed.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Repair a failed migration task by retrying the migration task.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Repair-VM -VM $VM -Retry

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 and stores the virtual machine object in variable $VM. This example

assumes that the task that you want to repair by using the Retry parameter

is an attempt to move (migrate) the virtual machine from one host to anothe

r.

The third command repairs VM01 by restarting the previous failed migration

task, and returns the virtual machine object in the transitional state to t

he user.

REMARKS

For more information, type: "get-help Repair-VM -detailed".

For technical information, type: "get-help Repair-VM -full".

### Resume-VM

SYNOPSIS

Resumes paused virtual machines managed by Virtual Machine Manager.

SYNTAX

Resume-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>] [-

RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Resumes one or more paused virtual machines managed by Virtual Machine Mana

ger. A paused virtual machine is one that has been suspended by using the S

uspend-VM cmdlet. Using the Resume-VM cmdlet to resume a virtual machine re

turns its object in a Running state. When the virtual machine is running ag

ain, the user can resume activity on that virtual machine.

If you run Resume-VM on a virtual machine that is already running, the cmdl

et returns a message indicating success but does not change the state of th

at virtual machine.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Resume a paused virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM01" }

PS C:\> Resume-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 (which this example assumes to be in a paused state) from the VMM da

tabase and stores the virtual machine object in variable $VM.

The last command puts VM01 into a running state and returns its object in a

running state to the user.

2: Resume all paused virtual machines.

PS C:\> $VMs = Get-VM -VMMServer VMMServer1.Contoso.com | where {$\_.Status

-eq "Paused"}

PS C:\> $VMs | Resume-VM

The first command gets from VMMServer1 all objects that represent virtual m

achines that are currently paused and stores the virtual machine objects in

$VMs (an object array).

The second command passes each object in $VMs to Resume-VM, which resumes e

ach virtual machine.

REMARKS

For more information, type: "get-help Resume-VM -detailed".

For technical information, type: "get-help Resume-VM -full".

### SaveState-VM

SYNOPSIS

Saves the state of virtual machines managed by Virtual Machine Manager.

SYNTAX

SaveState-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>]

[-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Saves the state (status and configuration changes) of one or more virtual m

achines managed by Virtual Machine Manager. Putting a virtual machine into

a saved state also returns its object in a saved state.

The SaveState-VM cmdlet stores a virtual machine so that it can be quickly

resumed, similar to a hibernated laptop. When you put a running virtual mac

hine into a saved state, the cmdlet stops the virtual machine, writes the d

ata that exists in memory to a temporary file with a .vsv extension (a save

d-state file), and stops the consumption of system resources by the virtual

machine.

Restoring a virtual machine from a saved state returns it to the same condi

tion that it was in when its state was saved. When you want to restore a vi

rtual machine from a saved state, you can use either DiscardSavedState-VM (

which returns the virtual machine object in a stopped state) or Start-VM (w

hich returns the virtual machine object in a running state).

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Save the state of a specific virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> SaveState-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM. This example assumes that only one virtual machine named VM01 exis

ts and that it is in a running state.

The last command saves the state of VM01 and returns its object in a saved

state to the user.

2: Save the state of the specified virtual machines.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMs = Get-VM | where { $\_.Name -match "VM0" -and $\_.VMHost.Name -e

q "VMHost02.Contoso.com" -and $\_.Status -eq “Running” }

PS C:\> $VMs | select Name,VMHost,Status

PS C:\> $VMs | SaveState-VM | Out-Null

PS C:\> $VMs | select Name,VMHost,Status

The first command connects to VMMServer1.

The second command gets the object for all virtual machines that match the

following characteristics: the virtual machine name matches the string "VM0

"; the host of the virtual machine is named VMHost02; and the virtual machi

ne is in a running state. The command stores the selected virtual machine o

bjects in $VMs.

The third command displays the name, host, and current status of each virtu

al machine object in $VMs to verify that these virtual machines match the s

pecified characteristics.

The fourth command passes each virtual machine object in $VMs to the SaveSt

ate-VM cmdlet, which puts each virtual machine into a saved state. The comm

and also uses the Out-Null cmdlet to redirect the output to $Null instead o

f sending the output to the console.

The last command displays the name, host, and current status of each virtua

l machine object in $VMs to verify that the status of these virtual machine

s is Saved.

REMARKS

For more information, type: "get-help SaveState-VM -detailed".

For technical information, type: "get-help SaveState-VM -full".

### Set-VM

SYNOPSIS

Changes properties of a virtual machine managed by Virtual Machine Manager.

SYNTAX

Set-VM [-VM] [<String VM>] -JobGroup <Guid> [-BootOrder <BootDevice[]>] [-C

ostCenter <String>] [-CPUCount <Int32>] [-CPUMax <Int32>] [-CPUReserve <Int

32>] [-CPUType [<ProcessorType String>]] [-Custom1 <String>] [-Custom10 <St

ring>] [-Custom2 <String>] [-Custom3 <String>] [-Custom4 <String>] [-Custom

5 <String>] [-Custom6 <String>] [-Custom7 <String>] [-Custom8 <String>] [-C

ustom9 <String>] [-DelayStart <Int32>] [-Description <String>] [-DiskIO <In

t32>] [-EnableBackup <Boolean>] [-Enabled <Boolean>] [-EnableDataExchange <

Boolean>] [-EnableHeartbeat <Boolean>] [-EnableOperatingSystemShutdown <Boo

lean>] [-EnableTimeSynchronization <Boolean>] [-ExcludeFromPRO <Boolean>] [

-ExpectedCPUUtilization <Int32>] [-HighlyAvailable <Boolean>] [-InstallVirt

ualizationGuestServices] [-JobVariable <String>] [-LimitCPUFunctionality <B

oolean>] [-MemoryMB <Int32>] [-Name <String>] [-NetworkUtilization <Int32>]

[-NumLock] [-OperatingSystem <OperatingSystem>] [-Owner <String>] [-PROTip

ID <Guid>] [-QuotaPoint <Int32>] [-RelativeWeight <Int32>] [-RemoveSelfServ

iceUserRole <UserRole>] [-RunAsSystem] [-RunAsUserCredential <PSCredential>

] [-RunAsynchronously] [-StartAction <String>] [-StopAction <String>] [-Tag

<String>] [-UseHardwareAssistedVirtualization] [-UserRole <UserRole>] [-VM

wareResourcePool <VMwareResourcePool>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual machine managed by Virtual Mach

ine Manager. Properties that you can change include:

\* Name, owner, and description of a VM.

\* BIOS boot order (for VMs on a Hyper-V host)

\* Amount of resources on the host used by a VM. These include:

- Maximum amount of host CPU resources that a VM can use.

- Minimum amount of host CPU resources that a VM can use.

- Expected use of host CPU by a VM.

- Amount of host CPU resources used by one VM relative to

other VMs on the same host.

- Amount of host memory that a VM can use.

- Amount of bandwidth on the host's network that a VM can use.

\* Hardware settings for a VM unrelated to host resources. These include:

- Number of CPUs

- Type of CPU

- Number of disk input/output operations per second (IOPS)

- Limiting CPU functionality (for an older operating system,

such as Windows NT 4.0)

\* Cost center, tag, and custom settings used to filter VMs by criteria

that you set.

\* Settings that enable various optional capabilities, including:

- Enabling or disabling a library object to make it available,

or temporarily unavailable, to users

- Enabling backing up a VM on a Hyper-V host with Volume Shadow Copy

- Enabling a key/value pair for data exchange between a VM and its

Hyper-V host

- Enabling a signal to monitor a VM on a Hyper-V host.

- Enabling shutdown of a VM from teh Hyper-V console.

- Enabling time synchronization between a VM and its Hyper-V host.

- Enabling the BIOS value for NumLock for a VM on a Windows host.

\* Setting that identifies whether a VM is highly available, that is,

a VM to be deployed on a node of a Hyper-V host cluster.

\* Setting that determines whether virtualization guest services are

installed on a Windows-based VM.

\* Number of seconds to delay before starting a VM.

\* Setting that identifies the operating system used for a VM.

\* Settings that specify whether to run a VM on a Virtual Server host under

the local system account or under a guest account (domain\account).

\* Start and stop actions for a VM.

\* Setting that determines whether a VM on a Virtual Server host uses

hardware-assisted virtualization.

\* Setting that limits the number of VMs self-service users can create.

\* Setting used to switch the role that a self-service user who belongs

to multiple roles uses to manage a VM.

\* Setting that assigns a VM on an ESX host to a VMware resource pool.

If you want to change the properties of a virtual floppy drive, virtual DVD

drive, virtual network adapter, or virtual SCSI adapter associated with a

specific virtual machine, you can use Set-VirtualFloppyDrive, Set-VirtualDV

DDrive, Set-VirtualNetworkAdapter, or Set-VirtualSCSIAdapter, respectively.

For more information about virtual machines, type:

Get-Help New-VM -detailed

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-BootOrder <BootDevice[]>

Specifies the order of devices that a virtual machine on a Hyper-V host

uses to start up.

Valid values: CD, IDEHardDrive, PXEBoot, or Floppy.

Example: -BootOrder PXEBoot,IDEHardDrive,CD,Floppy

-CostCenter <String>

Specifies the cost center for a virtual machine so that you can collect

data about the allocation of virtual machines (or resources allocated

to virtual machines) to make use of in your billing system.

-CPUCount <Int32>

Specifies the number of CPUs on a virtual machine, on a hardware profil

e, or on a template. See the examples for a specific cmdlet to determin

e how that cmdlet uses this parameter.

TYPE OF HOST NUMBER OF PROCESSORS

------------ --------------------

Virtual Server 1 CPU per VM

Hyper-V Up to 4 CPUs per VM; varies by guest OS

VMware ESX Up to 4 CPUs per VM for any supported guest OS

Exception: 1 CPU on a VM running Windows NT 4.0

Note: In VMM 2007, this parameter was named ProcessorCount.

-CPUMax <Int32>

Specifies the highest percentage of the total resources of a single CPU

on the host that can be used by a specific virtual machine at any give

n time.

Example: -CPUMax 80 (to specify 80 per cent)

-CPUReserve <Int32>

Specifies the minimum percentage of the resources of a single CPU on th

e host to allocate to a virtual machine. The percentage of CPU capacity

that is available to the virtual machine is never less than this perce

ntage.

-CPUType [<ProcessorType String>]

Specifies the type of CPU for a virtual machine. To retrieve a list of

all CPU types that are available for use in virtual machines in a VMM e

nvironment, type: "Get-CPUType"

-Custom1 <String>

Specifies a custom property on a VMM object.

-Custom10 <String>

Specifies a custom property on a VMM object.

-Custom2 <String>

Specifies a custom property on a VMM object.

-Custom3 <String>

Specifies a custom property on a VMM object.

-Custom4 <String>

Specifies a custom property on a VMM object.

-Custom5 <String>

Specifies a custom property on a VMM object.

-Custom6 <String>

Specifies a custom property on a VMM object.

-Custom7 <String>

Specifies a custom property on a VMM object.

-Custom8 <String>

Specifies a custom property on a VMM object.

-Custom9 <String>

Specifies a custom property on a VMM object.

-DelayStart <Int32>

Specifies the number of seconds to wait after the virtualization servic

e starts before automatically starting a virtual machine. Used to stagg

er the startup time of multiple virtual machines to help reduce the dem

and on the physical computer’s resources. A typical setting might be 30

to 60 seconds.

TYPE OF HOST MAXIMUM CONFIGURABLE DELAY

------------ --------------------------------

Hyper-V 1000000000 seconds (277777 hours)

Virtual Server 86400 seconds (24 hours)

VMware ESX 65535 seconds (18 hours)

-Description <String>

Specifies a description for the specified object.

-DiskIO <Int32>

Specifies the number of disk input/output operations per second (IOPS)

on the host that can be used by a specific virtual machine.

Example: -DiskIO 1500 (to specify 1500 IOPS).

-EnableBackup <Boolean>

Enables the use of the Volume Shadow Copy service to back up a virtual

machine if the virtual machine is deployed on a Hyper-V host.

-Enabled <Boolean>

Enables a library object (when set to TRUE) or disables a library objec

t (when set to FALSE). For example, if you want to upgrade software on

a virtual machine template, you can disable the template object in the

VMM library to temporarily prevent users from using that object.

-EnableDataExchange <Boolean>

Enables the use of a key/value pair for the exchange of data between a

virtual machine and the host operating system if the virtual machine is

deployed on a Hyper-V host.

-EnableHeartbeat <Boolean>

Enables the use of a heartbeat (a signal emitted at regular intervals)

to monitor the health of a virtual machine deployed on a Hyper-V host.

-EnableOperatingSystemShutdown <Boolean>

Enables the shut down of the operating system on a virtual machine mana

ged by VMM from Hyper-V's management interfaces on the host if the virt

ual machine is deployed on a Hyper-V host.

-EnableTimeSynchronization <Boolean>

Enables synchronizing the system time of a virtual machine with the sys

tem time of the operating system running on the host if the virtual mac

hine is deployed on a Hyper-V host.

-ExcludeFromPRO <Boolean>

Excludes (when set to TRUE) this virtual machine from being changed by

implementing host-targeted PRO tips.

-ExpectedCPUUtilization <Int32>

Specifies (as a percentage) the amount of CPU on the host that you expe

ct this virtual machine to use. This value is used only when VMM determ

ines a suitable host for the virtual machine.

-HighlyAvailable <Boolean>

Specifies that a virtual machine will be placed on a Hyper-V host that

is part of a host cluster. Configure this setting on a virtual machine,

or on a template or hardware profile that will be used to create virtu

al machines.

-InstallVirtualizationGuestServices

Installs virtualization guest services on a Windows-based virtual machi

ne. By default, this parameter is set to FALSE and

VMM installs the appropriate virtualization guest service automatically

. For a virtual machine on a Hyper-V host, the virtualization guest ser

vice is called Integration Components (VMGuest.iso). For a virtual mach

ine on a Virtual Server host, the virtualization guest service is calle

d Virtual Machine Additions (VMAdditions.iso). Virtual machines on a VM

ware ESX host do not use a virtualization guest service.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LimitCPUFunctionality <Boolean>

Enables running an older operating system (such as Windows NT 4.0) on a

virtual machine deployed on a Hyper-V host or on a VMware ESX host by

providing only limited CPU functionality for the virtual machine.

-MemoryMB <Int32>

Specifies, in megabytes (MB), the total amount of memory on the host th

at is assigned to a virtual machine.

TYPE OF HOST MAXIMUM HOST MEMORY ASSIGNABLE TO VM

------------ ------------------------------------

Virtual Server Up to 3712 MB ( 3.7 GB) RAM per VM

Hyper-V Up to 65536 MB (64.0 GB) RAM per VM

VMware ESX Server 3.0.x Up to 16384 MB (16.0 GB) RAM per VM

VMware ESX Server 3.5.x Up to 65536 MB (64.0 GB) RAM per VM

-Name <String>

Specifies the name of a VMM object.

-NetworkUtilization <Int32>

Specifies, in megabits per second (Mb/s), the amount of bandwidth on th

e host's network that can be used by a specific virtual machine.

Example: -NetworkUtilization 10 (to specify 10 Mb/s)

-NumLock

Enables the BIOS value for NumLock on a virtual machine (or on a templa

te or hardware profile that is used to create virtual machines) on a Hy

per-V host. This parameter does not apply to virtual machines on Virtua

l Server hosts or on VMware ESX hosts.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-QuotaPoint <Int32>

Specifies a quota that limits the number of virtual machines self-servi

ce users can create.

-RelativeWeight <Int32>

Specifies the amount of CPU resources on a host that this virtual machi

ne can use relative to other virtual machines on the same host. A virtu

al machine with a higher setting is allocated more CPU resources than a

virtual machine with a lower setting.

TYPE OF HOST RANGE OF RELATIVE VALUES

------------ ------------------------

Virtual Server 1 to 10000

Hyper-V 1 to 10000

VMware ESX 2000 = High

1500 = Above Normal

1000 = Normal (default)

750 = Below Normal

500 = Low

1 to 1000000 = Custom

The VMware term for these values is "shares."

Note: See the examples for a specific cmdlet to determine how that cmdl

et uses this parameter.

-RemoveSelfServiceUserRole <UserRole>

Removes the specified self-service user role from the permission list o

f the virtual machine.

-RunAsSystem

Specifies that a virtual machine on a Virtual Server host will run unde

r the local system account. If specified, Virtual Server will not autom

atically start the virtual machine when the Virtual Server service star

ts. (This parameter does not apply to virtual machines on Hyper-V or VM

ware ESX hosts because Hyper-V and VMware run a virtual machine under t

he local system account by default; you cannot change this setting on t

hose virtualization platforms.)

-RunAsUserCredential <PSCredential>

Specifies the guest account (domain\account) that a virtual machine on

a Virtual Server host runs under. If specified, Virtual Server will aut

omatically start a virtual machine when the Virtual Server service star

ts. For enhanced security, create a special account with limited permis

sions:

FILE TYPE MINIMUM REQUIRED PERMISSIONS FOR GUEST ACCOUNT

----------- ----------------------------------------------

.vmc file Read Data, Write Data, Execute File

.vmc folder List Folder, Write/Create File (required to save VM state)

.vhd file Read Data, Read Attributes, Read Extended Attributes,

Write Data

.vnc file Execute File, Read Data, Read Attributes, Read

(required if VM connects to a virtual network)

Note: This parameter does not apply to virtual machines on Hyper-V or V

Mware ESX hosts.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-StartAction <String>

Specifies the behavior of a virtual machine when the virtualization ser

vice (Hyper-V, Virtual Server, or VMware) starts. To specify that a vir

tual machine deployed on a Virtual Server host starts automatically, us

e the -RunAsUserCredential parameter to specify an account with appropr

iate permissions (otherwise, the StartAction reverts to NeverAutoTurnOn

VM).

Valid values: AlwaysAutoTurnOnVM, NeverAutoTurnOnVM, TurnOnVMIfRunningW

henVSStopped

-StopAction <String>

Specifies the behavior of the virtual machine when the virtualization s

ervice (Hyper-V, Virtual Server, or VMware) stops.

Valid values: SaveVM, TurnOffVM, ShutdownGuestOS

-Tag <String>

Associates a word or phrase with a virtual machine (or a template used

to create virtual machines) so that you can search for all virtual mach

ines with that tag as a set.

-UseHardwareAssistedVirtualization

Specifies that, for a virtual machine deployed on a Virtual Server host

, hardware-assisted virtualization is used if it is available (when set

to TRUE). The Virtual Server host must support AMD Virtualization (AMD

-V) or Intel Virtualization Technology (Intel-VT) hardware virtualizati

on. This parameter does not apply to virtual machines on Hyper-V hosts

or VMware ESX hosts.

-UserRole <UserRole>

Specifies a user role object.

-VMwareResourcePool <VMwareResourcePool>

Assigns a virtual machine deployed on a VMware ESX host to a specific V

Mware resource pool.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Specify an amount of memory for an existing virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> if($VM.Status -ne "PowerOff"){Stop-VM -VM $VM}

PS C:\> Set-VM -VM $VM -MemoryMB 1024

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command verifies whether the VM is in a powered off state. If the

VM is not in a powered off state, the command uses the Stop-VM command to

power off the VM. (Using Stop-VM to power off the VM is similar to pulling

the plug to power off a physical computer. Alternatively, you can use the S

hutdown-VM cmdlet to shut down the virtual machine in a way that is compara

ble to pushing the power button on a physical machine so that the operating

system can shut itself down systematically.)

The last command changes the memory allocated to VM01 to 1024 MB. This is t

he amount of memory on the host that VM01 can use.

2: For a user who belongs to multiple self-service user roles, switch the r

ole used to manage a specific VM.

PS C:\> $VM = Get-VM -VMMServer "VMMServer1.Contoso.com" -Name "VM02"

PS C:\> $SSRole = Get-VMMUserRole | where { $\_.Name -eq "SSUserRole3" }

PS C:\> Set-VM -VM $VM -UserRole $SSRole

The first command gets from VMMServer1 the object that represents the VM na

med VM02 and stores the VM object in $VM.

The second command gets the object that represents the self-service user ro

le named SSUserRole3 and stores the user role object in $SSRole.

The last command specifies that members of the self-service user role calle

d SSUserRole3 are now granted the permission to manage the VM called VM02.

WHAT YOU NEED TO KNOW:

VMM uses the UserRole parameter to set which VMs are managed by the members

of a specific self-service user role. Typically, you do not need to use th

e Set-VM cmdlet with the UserRole parameter to configure this setting. Howe

ver, if one or more users are members of multiple self-service user roles a

nd you grant them permission to manage multiple VMs on the same host, you m

ight encounter a case where want to switch which user role is authorized to

manage a particular VM. This example illustrates that scenario.

3. Disable time syncronization on a virtual machine used as a domain contro

ller.

PS C:\> $EAP = $ErrorActionPreference

PS C:\> $ErrorActionPreference = "STOP"

PS C:\> Get-VMMServer "VMMServer1.contoso.com"

PS C:\> $VM = Get-VM | where {$\_.Name -eq "VM03"}

PS C:\> trap{"Fail: Cannot disable Time Synchronization for VM: $VM";contin

ue} Set-VM -VM $VM -EnableTimeSynchronization $TRUE | Out-Null

PS C:\> $ErrorActionPreference = $EAP

The first command stores the current setting for $ErrorActionPreference in

variable $EAP. This variable will be used later to return the setting to it

s original value.

The second command sets the action preference to STOP. This error action pr

eference changes an error from a non-terminating error to a terminating err

or. The error object is thrown as an exception instead of being written to

the output pipe, and execution does not continue.

The third command connects to VMMServer1.

The fourth command gets the object that represents a VM named VM03 and stor

es the virtual machine object in $VM.

The fifth command disables the time syncronization setting. Typically, disa

bling time synchronization is required for VMs that act as domain controlle

rs. The command uses the trap statement to catch terminating exceptions. If

the Set-VM command fails, the string in the trap statement is displayed. C

ontinue is used in the trap statement to continue execution instead of exit

ing. The Out-Null cmdlet redirects the output to $Null instead of sending i

t to the console.

The last command sets the value for $ErrorActionPreference to the value sto

red in $EAP.

4. Set the device boot order for all VMs that support this feature.

PS C:\> $EAP = $ErrorActionPreference

PS C:\> $ErrorActionPreference = "STOP"

PS C:\> Get-VMMServer "VMMServer1.Contoso.com"

PS C:\> $VMs = @(Get-VM)

PS C:\> foreach($VM in $VMs){trap{"Fail: Cannot set BIOS for VM: $VM";conti

nue} Set-VM -VM $VM -BootOrder "PXEBoot","IDEHarddrive","CD","Floppy" | Out

-Null}

PS C:\> $ErrorActionPreference = $EAP

The first command stores the current setting for $ErrorActionPreference in

$EAP. This variable will be used later to return the setting to its origina

l value.

The second command sets the action preference to STOP. This error action pr

eference changes an error from a non-terminating error to a terminating err

or. The error object is thrown as an exception instead of being written to

the output pipe, and execution does not continue.

The third command connects to VMMServer1.

The fourth command gets the object that represents each VM and stores the V

M objects in $VMs. Using the '@' symbol and parentheses ensures that the co

mmand stores the results in an array (in case the command returns a single

object or a null value).

The fifth command sets the BIOS boot order for each VM to PXEBoot,IDEHarddr

ive,CD,Floppy. The command uses trap statement to catch terminating excepti

ons. If the Set-VM command fails, the string in the trap statement is displ

ayed. Continue is used in the trap statement to continue execution instead

of exiting the foreach loop. The Out-Null cmdlet redirects the output to $N

ull instead of sending it to the console.

NOTE: The BootOrder parameter is used only for virtual machines on Hyper-V

hosts; it is not used for virtual machines on Virtual Server or VMware ESX

hosts.

The last command sets the value for $ErrorActionPreference to the value sto

red in EAP.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

5: Specify an owner for all virtual machines with an "Unknown" owner.

PS C:\> Get-VM -VMMServer "VMMServer1.Contoso.com" | where {$\_.Owner -eq "U

nknown"} | Set-VM -Owner "Contoso\KimAkers"

Gets all VM objects from the VMM database, selects only those VM objects wh

ose owner is "Unknown," and specifies an owner for each VM.

REMARKS

For more information, type: "get-help Set-VM -detailed".

For technical information, type: "get-help Set-VM -full".

### Shutdown-VM

SYNOPSIS

Shuts down a running virtual machine managed by Virtual Machine Manager.

SYNTAX

Shutdown-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>]

[-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Shuts down one or more running virtual machines managed by Virtual Machine

Manager and returns the object or objects in an off state.

The Shutdown-VM cmdlet shuts down a running virtual machine in a way that i

s comparable to pushing the power button on a physical machine so that the

operating system can shut itself down systematically. However, you can use

Shutdown-VM on a virtual machine on a Windows-based host (a Hyper-V host or

a Virtual Server host) only if virtualization guest services are installed

on the virtual machine. Virtualization guest services provide software dri

vers that improve performance and make it easier to use a virtual machine.

For a virtual machine deployed on a Hyper-V host, the virtualization guest

service is called Integration Components. For a virtual machine deployed on

a Virtual Server host, the virtualization guest service is called Virtual

Machine Additions.

If virtualization guest services are not installed on a Hyper-V or Virtual

Server host, you must use the Stop-VM cmdlet to stop a running virtual mach

ine. Stop-VM stops a running virtual machine in the same way that a physica

l machine stops when you pull out its power plug.

When you want to start a virtual machine that has been shut down, use the S

tart-VM cmdlet.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Shuts down the specified virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Shutdown-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM. This example assumes that only one virtual machine named VM01 exis

ts and that it is in a running state.

The last command shuts down VM01 and returns its object in an Off state to

the user.

REMARKS

For more information, type: "get-help Shutdown-VM -detailed".

For technical information, type: "get-help Shutdown-VM -full".

### Start-VM

SYNOPSIS

Starts virtual machines managed by Virtual Machine Manager.

SYNTAX

Start-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>] [-R

unAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Starts one or more virtual machines on hosts managed by Virtual Machine Man

ager when the machines are currently turned off. Starting a turned off virt

ual machine restores it to a running state and returns its object in a runn

ing state. When the virtual machine is running again, you can resume activi

ty on that virtual machine.

If you run Start-VM on a virtual machine that is already running, the cmdle

t returns a message indicating success but does not change the state of the

virtual machine.

If you want to stop a running virtual machine, use the Stop-VM cmdlet.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Start a virtual machine that is currently turned off.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Start-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The last command starts VM01, which is currently turned off, and returns it

s object in a running state to the user.

2: Start all virtual machines that are currently turned off.

PS C:\> $VMs = Get-VM -VMMServer VMMServer1.Contoso.com | where { $\_.Status

-eq "PowerOff" }

PS C:\> $VMs | Start-VM

The first command gets all objects that represent virtual machines that are

currently turned off and stores these objects in $VMs (an object array).

The second command passes each virtual machine object in $VMs to the Start-

VM cmdlet, which starts each virtual machine in the array.

REMARKS

For more information, type: "get-help Start-VM -detailed".

For technical information, type: "get-help Start-VM -full".

### Stop-VM

SYNOPSIS

Stops virtual machines on hosts managed by Virtual Machine Manager.

SYNTAX

Stop-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>] [-Ru

nAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Stops one or more running virtual machines on hosts managed by Virtual Mach

ine Manager and returns the virtual machine object or objects in a stopped

state. Stop-VM stops a virtual machine in the same way that a physical mach

ine stops when you pull out its power plug.

When you want to resume running a stopped virtual machine, use the Start-VM

cmdlet.

For information about an alternative way to shut down a virtual machine, wh

ich is comparable to pushing the power button on a physical machine so that

the operating system can shut itself down systematically, type:

Get-Help Shutdown-VM -detailed

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Stop a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Stop-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM. This example assumes that only one virtual machine named VM01 exis

ts and that it is currently in a running state.

The last command stops VM01 and returns its stopped object to the user.

NOTE: This command will fail if the virtual machine is in a state (such as

powered off or stored) for which the "stop" action is invalid.

2: Stop multiple virtual machines.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $SelectVMs = Get-VM | where { $\_.Name -match "PartialName" -and $\_.

Status -eq "Running" }

PS C:\> Get-VM | Stop-VM

The first command connects to VMMServer1.

The second command gets the objects for all virtual machines whose name mat

ches the specified string and whose current status is "Running", and it sto

res those virtual machine objects in $SelectVMs.

The last command passes each virtual machine object in $SelectVMs to Stop-

VM, which stops each virtual machine.

NOTE: This command will fail for any virtual machine that is in a state (su

ch as powered off) for which the "stop" action is invalid.

REMARKS

For more information, type: "get-help Stop-VM -detailed".

For technical information, type: "get-help Stop-VM -full".

### Store-VM

SYNOPSIS

Stores a virtual machine currently deployed on a virtual machine host by mi

grating it from the host to the Virtual Machine Manager library.

SYNTAX

Store-VM [-VM] [<String VM>] -LibraryServer [<String LibraryServer>] -Share

Path <String> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronousl

y] [-UseLAN <Boolean>] [<CommonParameters>]

DETAILED DESCRIPTION

Stores a virtual machine currently deployed on a virtual machine host by mi

grating it from the host to the Virtual Machine Manager library. A virtual

machine must be in a saved state, shut down, or turned off before it can be

moved from the host and stored in the library.

The Store-VM cmdlet lets you store a virtual machine by using one of the fo

llowing transfer methods:

\* SAN TRANSFER (Fibre Channel, iSCSI, or NPIV) - If the host and library

server are both connected to SAN storage, VMM can use a SAN transfer to

store the virtual machine in the library. In a SAN transfer, the target

LUNs are remapped from the source host to the destination library

server. No files are moved, which is why a SAN transfer is much faster

than moving virtual machine files from one host to another over a

local area network (LAN). VMM can use an NPIV SAN transfer if a host

bus adapter (HBA) with NPIV support is available.

\* NETWORK TRANSFER - If no faster method is available, VMM uses a

network transfer to move the virtual machine files from the host server

to the library server over the LAN that connects the two servers. You

must use the -SharePath parameter to specify the path to the share in

the library where you want to store the virtual machine.

Store-VM automatically uses the fastest available transfer type. If you wan

t to force a network transfer, you can use the UseLAN parameter. If the hos

t server and library server are the same server, the command will not fail

if you specify the UseLAN parameter, but the migration to the library will

occur faster if you do not use that parameter.

When a virtual machine is stored in the library, it cannot be started. Befo

re you can start the virtual machine, you must use the Move-VM cmdlet to mo

ve it from the library to a virtual machine host.

NOTE: For information about how to put the VM into a saved state, shut it d

own, or turn it off, type:

Get-Help SaveState-VM -detailed

Get-Help Shutdown-VM -detailed

Get-Help Stop-VM -detailed

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-SharePath <String>

Specifies a path to a valid library share on an existing library server

that uses a Universal Naming Convention (UNC) path.

Example format: –SharePath "\\FileServer01\LibShare"

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-UseLAN <Boolean>

Forces a transfer over the local area network (LAN) even if a faster tr

ansfer mechanism, such as a storage area network (SAN) transfer, is ava

ilable.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Store a virtual machine in the library.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $Library = Get-LibraryServer -ComputerName "FileServer01"

PS C:\> Store-VM -LibraryServer $Library -VM $VM -SharePath "\\FileServer01

.Contoso.com\MyLibrary1\VMs"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The third command gets the object that represents the library server named

FileServer01 and stores the library server object in variable $Library.

The last command removes VM01 from its host and stores it to the location \

\FileServer01.Contoso.com\MyLibrary1\VMs on FileServer01. The command autom

atically uses the fastest available transfer type.

NOTE: A virtual machine must be in a saved state, shut down, or turned off

before it can be moved from the host and stored in the library. For more in

formation, type:

Get-Help SaveState-VM -detailed

Get-Help Shutdown-VM -detailed

Get-Help Stop-VM -detailed

2: Store a virtual machine in the library asynchronously.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM02"

PS C:\> $Library = Get-LibraryServer -ComputerName "FileServer02"

PS C:\> Store-VM -LibraryServer $Library -VM $VM -SharePath "\\FileServer02

.Contoso.com\MyLibrary1\VMs" -RunAsynchronously -JobVariable "MyJob"

PS C:\> $MyJob

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command gets the object that represents the library server named

FileServer02 and stores the library server object in $Library.

The fourth command stores VM02 to the location \\MyLibrary1\VMs on FileServ

er02. The RunAsynchronously parameter returns control to the shell immediat

ely and the JobVariable parameter tracks job progress and stores a record o

f its progress in MyJob. For JobVariable, you do not use the dollar sign ($

) when the variable is created.

The last command displays the contents of $MyJob.

3: Store a virtual machine in the library by forcing a network transfer.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM03"

PS C:\> $Library = Get-LibraryServer -ComputerName "FileServer01"

PS C:\> Store-VM -LibraryServer $Library -VM $VM -SharePath "\\FileServer01

.Contoso.com\MyLibrary1\VMs" -UseLAN

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM03 and stores the virtual machine object in $VM.

The third command gets the object that represents the library server named

FileServer03 and stores the library server object in $Library.

The last command stores VM03 to the location \\MyLibrary1\VMs on FileServer

03. The UseLAN parameter forces a network transfer over the LAN even if a f

aster transfer mechanism is available.

REMARKS

For more information, type: "get-help Store-VM -detailed".

For technical information, type: "get-help Store-VM -full".

### Suspend-VM

SYNOPSIS

Suspends execution on virtual machines managed by Virtual Machine Manager.

SYNTAX

Suspend-VM [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <Guid>] [

-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Suspends execution on one or more virtual machines deployed on hosts manage

d by Virtual Machine Manager. Suspending a virtual machine freezes, or paus

es, activity on that virtual machine and returns its object in a paused sta

te.

You can use the Resume-VM cmdlet to put a suspended virtual machine back in

to a running state.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Suspend the specified virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Suspend-VM -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the virtual machine name

d VM01 from the VMM database and stores the virtual machine object in varia

ble $VM.

The last command suspends VM01 and displays information about its suspended

object to the user.

REMARKS

For more information, type: "get-help Suspend-VM -detailed".

For technical information, type: "get-help Suspend-VM -full".

## VMCheckpoint

### Get-VMCheckpoint

SYNOPSIS

Gets virtual machine checkpoint objects from the Virtual Machine Manager da

tabase.

SYNTAX

Get-VMCheckpoint [-MostRecent] [-VM [<String VM>]] [-VMMServer [<String Ser

verConnection>]] [<CommonParameters>]

Get-VMCheckpoint [-ID <Guid>] [-VMMServer [<String ServerConnection>]] [<Co

mmonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual machine checkpoints from th

e Virtual Machine Manager database. A virtual machine checkpoint is a point

-in-time "snapshot" of a virtual machine.

If the virtual machine is running on a Hyper-V host or on a VMware host, ch

eckpoint properties that you can retrieve with the Get-VMCheckpoint cmdlet

have been expanded to include hardware properties. If the virtual machine i

s running on a Virtual Server host, checkpoint properties do not include ha

rdware properties.

You can use the checkpoint later to revert the virtual machine to its origi

nal state.

For more information about Virtual Machine Manager checkpoints, type:

Get-Help New-VMCheckpoint -detailed

PARAMETERS

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-MostRecent

Specifies the most recent VMM virtual machine checkpoint object.

-VM [<String VM>]

Specifies a virtual machine object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all checkpoints for each virtual machine.

PS C:\> Get-VMCheckpoint -VMMServer VMMServer1.Contoso.com

Gets from the VMM database on VMMServer1 all checkpoint objects for each vi

rtual machine and displays information about these checkpoint objects to th

e user.

This (and subsequent) examples assume that checkpoints were created earlier

on one or more virtual machines in your VMM environment.

2: Get all checkpoints for one or more virtual machines with a specific nam

e.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VMCheckpoint -VM "VM01"

PS C:\> Write-Output $Checkpoints

The first command connects to VMMServer1.

The second command gets all virtual machine checkpoint objects for any virt

ual machine named VM01 that is deployed on any host managed by VMM. The com

mand stores these checkpoint objects in $Checkpoints (an object array).

The last command displays the checkpoint objects in $Checkpoints.

3: Get the hardware profile of the last restored checkpoint on a VM deploye

d on a Hyper-V host.

PS C:\> $VM = Get-VM -VMMServer VMMServer1.Contoso.com -Name "VM04"

PS C:\> $VM.LastRestoredVMCheckpoint.CheckpointHWProfile

The first command gets the object that represents VM04 from VMMServer1 and

stores this object in $VM. This example assumes that multiple checkpoints e

xist and that the virtual machine has been restored to one of the checkpoin

ts.

The second command displays information about the hardware profile of the l

ast restored checkpoint on VM04, including the number of CPUs, the amount o

f memory, and so on.

NOTE: If the virtual machine is deployed on a Hyper-V host or on a VMware h

ost, the checkpoint properties include the hardware profile. However, if th

e virtual machine is deployed on a Virtual Server host, the checkpoint prop

erties do not include the hardware profile.

4: Get the hardware profile of the most recently created checkpoint on a VM

deployed on a Hyper-V host.

PS C:\> $VM = Get-VM -VMMServer VMMServer1.Contoso.com -Name "VM05"

PS C:\> $Checkpoint = $VM | Get-VMCheckpoint -MostRecent

PS C:\> $Checkpoint.CheckpointHWProfile

The first command gets the object that represents VM05 and stores this obje

ct in $VM.

The second command gets the most recent checkpoint object created for VM05

and stores that checkpoint object in $Checkpoint.

The last command displays information about the hardware profile of the mos

t recent checkpoint created for VM05.

NOTE: If the virtual machine is deployed on a Hyper-V host or on a VMware h

ost, the checkpoint properties include the hardware profile. However, if th

e virtual machine is deployed on a Virtual Server host, the checkpoint prop

erties do not include the hardware profile.

5: Display the .NET type, methods, and properties for checkpoint objects on

VMs deployed on Hyper-V or VMware hosts.

PS C:\> $Checkpoints = Get-VMCheckpoint -VMMServer VMMServer1.Contoso.com

PS C:\> $Checkpoints | Get-Member

PS C:\> $Checkpoints | Get-Member | Format-List

The first command gets from VMMServer1 all checkpoint objects for each virt

ual machine and stores the checkpoint objects in $Checkpoints (an object ar

ray).

The second command passes each checkpoint object in $Checkpoints to the Get

-Member cmdlet, which displays the following:

\* TypeName: Microsoft.SystemCenter.VirtualMachineManager.VMSnapshot

\* MemberType: A list of the name and definition for each method and

property associated with this object type.

The last command is the same as the second command except that it pipes the

output to the Format-List cmdlet so that you can see the complete definiti

on for each method and each property for the checkpoint object type.

NOTE: This example assumes that the checkpoints are for a virtual machine d

eployed on either a Hyper-V host or on a VMware ESX host. Checkpoints for a

virtual machine deployed on a Virtual Server host are a different .NET typ

e:

Microsoft.SystemCenter.VirtualMachineManager.VMCheckpoint

REMARKS

For more information, type: "get-help Get-VMCheckpoint -detailed".

For technical information, type: "get-help Get-VMCheckpoint -full".

### Merge-VMCheckpoint

SYNOPSIS

Removes a virtual machine checkpoint object from the Virtual Machine Manage

r database.

NOTE: In VMM 2008, this cmdlet was renamed Remove-VMCheckpoint. This cmdle

t will be deprecated in future releases.

SYNTAX

Merge-VMCheckpoint -VMCheckpoint <VMCheckpoint> [-Confirm] [-JobGroup <Guid

>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<Common

Parameters>]

DETAILED DESCRIPTION

Removes a checkpoint object for a virtual machine from the Virtual Machine

Manager database.

PARAMETERS

-VMCheckpoint <VMCheckpoint>

Specifies a VMM virtual machine checkpoint object.

-Confirm

Prompts for confirmation before running the command.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the most recent checkpoint for a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VMCheckpoint -VM "VM01"

PS C:\> Remove-VMCheckpoint -VMCheckpoint $Checkpoints[0] -Confirm

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets from the VMM database all checkpoint objects for th

e virtual machine called VM01 and stores these objects in variable $Checkpo

ints (an object array). This example assumes that VM01 has several checkpoi

nts.

The third command removes the first checkpoint in the array ($Checkpoints[0

]). The Confirm parameter prompts you to confirm whether you want to remove

the checkpoint.

CAUTION: You cannot remove a checkpoint from a running virtual machine if t

he virtual machine is deployed on a Virtual Server host. Therefore, if the

virtual machine is on a Virtual Server host, the Remove-VMCheckpoint cmdlet

shuts down or stops the virtual machine while the cmdlet processes the ope

ration. Whether the virtual machine is shut down or stopped depends on whet

her or not the virtualization guest service appropriate for a virtual machi

ne on a Virtual Server host (Virtual Machine Additions) is installed on the

virtual machine:

\* If Virtual Machine Additions is installed on the virtual machine,

VMM shuts down the virtual machine (comparable to using Shut Down

from the Start menu) before the checkpoint is removed.

\* If Virtual Machine Additions is not installed on the virtual machine,

the virtual machine is stopped without warning (comparable to

pulling out the power plug on a physical machine) if you perform this

operation from the command line. (If you perform this operation from

the Administrator Console, a warning message appears before the

virtual machine is stopped.)

2: Remove the specified checkpoint for a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VMCheckpoint -VM "VM02"

PS C:\> Remove-VMCheckpoint -VMCheckpoint $Checkpoints[1]

The first command connects to VMMServer1.

The second command gets all virtual machine checkpoint objects for VM02 and

stores these objects in variable $Checkpoints (an object array). This exam

ple assumes that VM02 has two checkpoints, so $Checkpoints contains two che

ckpoint objects.

The third command removes the second checkpoint ($Checkpoints[1])

CAUTION: You cannot remove a checkpoint from a running virtual machine if t

he virtual machine is deployed on a Virtual Server host. For more informati

on, see the CAUTION in Example 1.

REMARKS

For more information, type: "get-help Merge-VMCheckpoint -detailed".

For technical information, type: "get-help Merge-VMCheckpoint -full".

### New-VMCheckpoint

SYNOPSIS

Creates a virtual machine checkpoint object for a virtual machine deployed

on a host managed by Virtual Machine Manager.

SYNTAX

New-VMCheckpoint [-VM] [<String VM>] [-Confirm] [-Description <String>] [-J

obVariable <String>] [-Name <String>] [-PROTipID <Guid>] [-RunAsynchronousl

y] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a virtual machine checkpoint object for a virtual machine deployed

on a host managed by Virtual Machine Manager. You can use a checkpoint to r

estore a virtual machine to a previous state.

A typical use is to create a checkpoint before you install an update to the

operating system or to an application on the virtual machine so that, if t

he update fails or adversely affects the virtual machine, you can use the R

estore-VMCheckpoint cmdlet to revert the virtual machine to its original st

ate.

For virtual machines deployed on a Hyper-V host or on a VMware ESX host, Vi

rtual Machine Manager creates the checkpoint without stopping the virtual m

achine, so no interruption in service occurs.

However, for virtual machines deployed on a Virtual Server host, it is advi

sable to shut down a virtual machine before creating a checkpoint. If the v

irtual machine is not in a Stopped or Turned Off state, Virtual Machine Man

ager must stop the virtual machine momentarily while the checkpoint is crea

ted.

It is important to back up data files on a virtual machine before you resto

re the virtual machine to a checkpoint. When you restore the virtual machin

e, user data files on its virtual hard disks are returned to their previous

state.

Although checkpoints let you restore a virtual machine to a previous state

after a change such as a system or application update, checkpoints do not p

rovide a permanent backup of the operating system, applications, or files.

Checkpoints are stored with the virtual machine on the host. Therefore, if

the host fails, checkpoints for virtual machines deployed on that host are

lost.

To provide data protection for your virtual machines, you can use the Volum

e Shadow Copy Service (VSS) writer for virtual machines on a Virtual Server

host. You can use a backup application such as Microsoft System Center Dat

a Protection Manager (DPM) to back up virtual machines on any type of host

to external storage.

NOTE: You can grant self-service users permission to create and manage chec

kpoints for their virtual machines. For more information, type: Get-Help Se

t-VMMUserRole -detailed

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-Confirm

Prompts for confirmation before running the command.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a virtual machine checkpoint for virtual machines that have the s

ame name but reside on different hosts.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VM -name "VM01" | New-VMCheckpoint

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command performs the following operations:

\* Gets from the VMM atabase all objects that represent virtual machines

deployed on any host managed by VMM.

\* Selects only those virtual machines named VM01. This

example assumes that more than one host contains a virtual

machine named VM01.

\* Creates a checkpoint for each of these virtual machines

and stores these checkpoint objects in $Checkpoints.

CAUTION: You cannot create a new a checkpoint on a running virtual machine

if the virtual machine is deployed on a Virtual Server host. Therefore, if

the virtual machine is on a Virtual Server host and you have not stopped or

shutdown the virtual machine, the New-VMCheckpoint cmdlet shuts down or st

ops the virtual machine while the cmdlet processes the operation. Whether t

he virtual machine is shut down or stopped depends on whether or not the vi

rtualization guest service appropriate for a virtual machine on a Virtual S

erver host (Virtual Machine Additions) is installed on the virtual machine:

\* If Virtual Machine Additions is installed on the virtual machine,

VMM shuts down the virtual machine (comparable to using Shut Down

from the Start menu) before the checkpoint is removed.

\* If Virtual Machine Additions is not installed on the virtual machine,

the virtual machine is stopped without warning (comparable to

pulling out the power plug on a physical machine) if you perform this

operation from the command line. (If you perform this operation from

the Administrator Console, a warning message appears before the

virtual machine is stopped.)

2: Create a virtual machine checkpoint for a specific virtual machine async

hronously.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VM -name "VM02" | New-VMCheckpoint -RunAsynchronously -JobVaria

ble "NewCkPtJob"

PS C:\> $NewCkPtJob

This example is the same as Example 1 except that the second command uses t

he RunAsynchronously parameter to return control to the shell immediately a

nd uses the JobVariable parameter to track job progress and store a record

of this progress in variable NewCkPtJob. For JobVariable, you do not use th

e dollar sign ($) when the variable is created.

The virtual machine checkpoint object that represents VM02 is returned init

ially in a "creating" state. After the checkpoint is created, the state cha

nges to "available."

The last command displays the contents of $NewCkPtJob.

CAUTION: You cannot create a new a checkpoint on a running virtual machine

if the virtual machine is deployed on a Virtual Server host. Therefore, if

the virtual machine is on a Virtual Server host and you have not stopped or

shut down the virtual machine, the New-VMCheckpoint cmdlet shuts down or s

tops the virtual machine while the cmdlet processes the operation. For more

information, see the CAUTION in Example 1.

REMARKS

For more information, type: "get-help New-VMCheckpoint -detailed".

For technical information, type: "get-help New-VMCheckpoint -full".

### Remove-VMCheckpoint

SYNOPSIS

Removes a virtual machine checkpoint object from the Virtual Machine Manage

r database.

NOTE: In VMM 2007, this cmdlet was named Merge-VMCheckpoint.

SYNTAX

Remove-VMCheckpoint -VMCheckpoint <VMCheckpoint> [-Confirm] [-JobGroup <Gui

d>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<Commo

nParameters>]

DETAILED DESCRIPTION

Removes a checkpoint object for a virtual machine from the Virtual Machine

Manager database.

PARAMETERS

-VMCheckpoint <VMCheckpoint>

Specifies a VMM virtual machine checkpoint object.

-Confirm

Prompts for confirmation before running the command.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the most recent checkpoint for a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VMCheckpoint -VM "VM01"

PS C:\> Remove-VMCheckpoint -VMCheckpoint $Checkpoints[0] -Confirm

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets from the VMM database all checkpoint objects for th

e virtual machine called VM01 and stores these objects in variable $Checkpo

ints (an object array). This example assumes that VM01 has several checkpoi

nts.

The third command removes the first checkpoint in the array ($Checkpoints[0

]). The Confirm parameter prompts you to confirm whether you want to remove

the checkpoint.

CAUTION: You cannot remove a checkpoint from a running virtual machine if t

he virtual machine is deployed on a Virtual Server host. Therefore, if the

virtual machine is on a Virtual Server host, the Remove-VMCheckpoint cmdlet

shuts down or stops the virtual machine while the cmdlet processes the ope

ration. Whether the virtual machine is shut down or stopped depends on whet

her or not the virtualization guest service appropriate for a virtual machi

ne on a Virtual Server host (Virtual Machine Additions) is installed on the

virtual machine:

\* If Virtual Machine Additions is installed on the virtual machine,

VMM shuts down the virtual machine (comparable to using Shut Down

from the Start menu) before the checkpoint is removed.

\* If Virtual Machine Additions is not installed on the virtual machine,

the virtual machine is stopped without warning (comparable to

pulling out the power plug on a physical machine) if you perform this

operation from the command line. (If you perform this operation from

the Administrator Console, a warning message appears before the

virtual machine is stopped.)

2: Remove the specified checkpoint for a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VMCheckpoint -VM "VM02"

PS C:\> Remove-VMCheckpoint -VMCheckpoint $Checkpoints[1]

The first command connects to VMMServer1.

The second command gets all virtual machine checkpoint objects for VM02 and

stores these objects in variable $Checkpoints (an object array). This exam

ple assumes that VM02 has two checkpoints, so $Checkpoints contains two che

ckpoint objects.

The third command removes the second checkpoint ($Checkpoints[1])

CAUTION: You cannot remove a checkpoint from a running virtual machine if t

he virtual machine is deployed on a Virtual Server host. For more informati

on, see the CAUTION in Example 1.

REMARKS

For more information, type: "get-help Remove-VMCheckpoint -detailed".

For technical information, type: "get-help Remove-VMCheckpoint -full".

### Restore-VMCheckpoint

SYNOPSIS

Restores a virtual machine on a host managed by Virtual Machine Manager to

the specified checkpoint.

SYNTAX

Restore-VMCheckpoint -VMCheckpoint <VMCheckpoint> [-Confirm] [-JobVariable

<String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Restores a virtual machine on a host managed by Virtual Machine Manager to

the state of the virtual machine at the time that the specified checkpoint

was created.

If the restore operation is successful, the Restore-VMCheckpoint cmdlet ret

urns (displays) the checkpoint object. If the operation fails, the cmdlet r

eturns an error message.

If you use the optional RunAsynchronously parameter, the cmdlet returns a v

irtual machine checkpoint object in the "Restoring" state. If the operation

then completes successfully, the cmdlet returns the virtual machine checkp

oint in the "Available" state.

PARAMETERS

-VMCheckpoint <VMCheckpoint>

Specifies a VMM virtual machine checkpoint object.

-Confirm

Prompts for confirmation before running the command.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Restore one or more virtual machines named VM01 to the most recent check

point.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMCheckpoint -VM "VM01" -MostRecent | Restore-VMCheckpoint

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets the most recent checkpoint object for all virtual m

achines named VM01 in the VMM database and restores each of these virtual m

achines to the state that it was in at the time its most recent checkpoint

was created.

CAUTION: Restoring a virtual machine to a checkpoint operates differently d

epending on the virtualization platform of the host:

\* VMs ON A HYPER-V OR VMWARE HOST. Restoring a virtual machine to the

specified checkpoint discards all changes made to the virtual

machine since the most recent checkpoint was created. Therefore,

a good practice is to first create a new checkpoint before you

restore the virtual machine to the earlier checkpoint to ensure

that the current state of the virtual machine is available (if

needed) after the restore operation.

After the virtual machine is restored to the specified checkpoint,

any checkpoints that were created earlier than that checkpoint are

still available -- including (if you create it) the checkpoint made

immediately before you perform the restore operation.

\* VMs ON A VIRTUAL SERVER HOST. Restoring a virtual machine to the

specified checkpoint discards all changes made to the virtual machine

after the specified checkpoint was created, including any checkpoints

that were created after the checkpoint to which you are restoring the

virtual machine.

After the restoration operation completes:

\* If the virtual machine is deployed on a Hyper-V or VMware host and

the checkpoint to which it is restored was taken on the virtual

machine while it was in a running state or saved state, the virtual

machine will be restored in a running state.

\* If the virtual machine is deployed on a Virtual Server host and was in

a running or paused state when the restore operation began, the virtual

machine will be restored in a running state.

2: Restore a virtual machine to the specified checkpoint.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Checkpoints = Get-VMCheckpoint -VM "VM02"

PS C:\> Restore-VMCheckpoint –VMCheckpoint $Checkpoints[$Checkpoints.count

- 2]

The first command connects to VMMServer1.

The second command gets all checkpoint objects for the virtual machine call

ed VM02 and stores these checkpoint objects in $Checkpoints (an object arra

y). This example assumes that only one host in the VMM environment contains

a virtual machine named VM02.

The third command restores VM02 to the second-from-last checkpoint. (This

example assumes you have at least 2 checkpoints.) The command deletes any e

xisting checkpoints that were created later than this checkpoint.

CAUTION: Restoring a virtual machine to a checkpoint operates differently d

epending on the virtualization platform of the host. For more information,

see the CAUTION in Example 1.

REMARKS

For more information, type: "get-help Restore-VMCheckpoint -detailed".

For technical information, type: "get-help Restore-VMCheckpoint -full".

### Set-VMCheckpoint

SYNOPSIS

Changes the Description property of a virtual machine checkpoint object in

Virtual Machine Manager.

SYNTAX

Set-VMCheckpoint -VMCheckpoint <VMCheckpoint> [-Description <String>] [-Job

Variable <String>] [-Name <String>] [-PROTipID <Guid>] [-RunAsynchronously]

[<CommonParameters>]

DETAILED DESCRIPTION

Changes a property of a virtual machine checkpoint object in Virtual Machin

e Manager. The only property that you can change by using the Set-VMCheckpo

int cmdlet is the Description property.

PARAMETERS

-VMCheckpoint <VMCheckpoint>

Specifies a VMM virtual machine checkpoint object.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Name <String>

Specifies the name of a VMM object.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Set the description for all checkpoints to the specified string.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMCheckpoint | Set-VMCheckpoint -Description "All checkpoints c

reated prior to upgrade"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets all objects that represent checkpoints that current

ly exist from the VMM database and specifies a description for these checkp

oints.

2: Set the name and description for all checkpoints to the specified string

.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMCheckpoint | Set-VMCheckpoint –Name “Checkpoint Before Upgrad

e” -Description "All checkpoints created prior to upgrade"

The first command connects to VMMServer1.

The second command gets all objects that represent virtual machine checkpoi

nts and specifies a name and description for each checkpoint.

3: Update one checkpoint in an array of checkpoints.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VMget VM03

PS C:\> Set-VMCheckpoint -VMCheckpoint $VM.VMCheckpoints[0] -Description “C

heckpoint Before Upgrade”

The first command connects to VMMServer1.

The second command gets the object that represents a virtual machine named

VM03 and stores the virtual machine object in $VM (an object array).

The last command specifies the description "Checkpoint for Beta" for the fi

rst checkpoint retrieved from VM01 ($VM.VMCheckpoints[0]).

REMARKS

For more information, type: "get-help Set-VMCheckpoint -detailed".

For technical information, type: "get-help Set-VMCheckpoint -full".

## VMDK

### Copy-VMDK

SYNOPSIS

Copies a VMware virtual hard disk file (a .vmdk file) to a Microsoft-compat

ible virtual hard disk file (a .vhd file) and converts the virtual hard dis

k for use in a Virtual Machine Manager environment.

SYNTAX

Copy-VMDK -Path <String> -VMDKPath <String> -VMHost [<String Host>] [-JobVa

riable <String>] [-LibraryServer [<String LibraryServer>]] [-Owner <String>

] [-PROTipID <Guid>] [-RunAsynchronously] [-SourceVMHost <Host>] [-VMMServe

r [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Copies a VMware virtual hard disk file (a .vmdk file) to a Microsoft-compat

ible virtual hard disk file (a .vhd file) and converts the virtual hard dis

k for use in a Virtual Machine Manager environment. The disk's contents are

preserved by this copy operation.

VMware virtual hard disks, stored in .vmdk files, contain the virtual machi

ne's guest operating system, applications, and data. VMWare virtual hard di

sk formats supported by the Copy-VMDK cmdlet include:

- monolithicSparse

- monolithicFlat

- vmfs

- twoGbMaxExtentSparse

- twoGbMaxExtentFlat

The Copy-VMDK cmdlet takes as its input the .vmdk file that the .vmx file p

oints to:

\* The .vmx file points to a .vmdk file that contains metadata, which in

turn points to the binary .vmdk file.

\* The .vmdk file that you specify with the Copy-VMDK cmdlet is the

.vmdk file that contains the metadata (not the binary .vmdk file).

For more information about VMWare .vmx and .vmdk files, type:

Get-Help New-V2V -detailed

PARAMETERS

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-VMDKPath <String>

Specifies the path to a VMware virtual hard disk file (a .vmdk file) to

be converted to a Windows-based virtual hard disk file (a .vhd file).

Example format: -VMDKPath "\\FileServer01\MSSCVMMLibrary\VMDKS\VM01.vmd

k"

Example format: -VMDKPath "[storage1] /VM01/VM01.vmdk"

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-Owner <String>

Specifies the owner of a VMM object in the form of a valid domain user

account.

Example format: -Owner “Contoso\RachelValdez”

Example format: -Owner “RachelValdez@Contoso”

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SourceVMHost <Host>

Specifies the source virtual machine host object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Convert a VMware .vmdk file in the VMM library to a Windows-based .vhd f

ile on a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $LibServ = Get-LibraryServer -ComputerName "FileServer01.Contoso.co

m"

PS C:\> $VMHost = Get-VMHost –ComputerName “VMMHost01.Contoso.com”

PS C:\> Copy-VMDK -LibraryServer $LibServ -VMDKPath "\\FileServer01\MSSCVMM

Library\VMDKS\VM01.vmdk" –VMHost $VMHost -Path "C:\StoredWindowsVMs"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the library server named

FileServer01 from the VMM database and stores the library object in variab

le $LibServ.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in variable $VMHost. VMHost01 is either a Hyper-V

host or a Virtual Server host (but not a VMware ESX host).

The last command copies and converts the .vmdk file located at the specifie

d path (\\FileServer01\MSSCVMMLibrary\VMDKS\VM01.vmdk) on the library serve

r and stores the resulting .vhd file at the specified path (C:\StoredWindow

sVMs) on VMHost01. Note that the -Path parameter, when used with Copy-VMDK,

cannot take a UNC path.

IMPORTANT:

The Copy-VMDK cmdlet takes as its input the .vmdk file that the .vmx file p

oints to:

\* The .vmx file points to a .vmdk file that contains metadata, which in tur

n points to the binary .vmdk file.

\* The .vmdk file that you specify with the Copy-VMDK cmdlet is the .vmdk fi

le that contains the metadata (not the binary .vmdk file).

2: Convert a VMware-based .vmdk file on an ESX host to a Windows-based .vhd

file and move it to a Windows host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName “WindowsVMHost02.Contoso.com”

PS C:\> $SourceVMHost = Get-VMHost -ComputerName “ESXVMHost02”

PS C:\> Copy-VMDK -SourceVMHost $SourceVMHost -VMDKPath "[storage1] /VM01/V

M01.vmdk" -VMHost $VMHost -Path "C:\VM01"

The first command connects to VMMServer1.

The second command gets the object that represents the Windows host named W

indowsVMHost02 and stores the host object in $VMHost. WindowsVMHost02 is ei

ther a Hyper-V host or a Virtual Server host (but not a VMware ESX host).

The third command gets the object that represents the ESX host named ESXVMH

ost02 by specifying its name and stores the host object in $SourceVMHost. (

If the host object is not returned, use the IP address with the ComputerNam

e parameter. DNS must be configured to resolve computer names to the corres

ponding IP addresses in order to retrieve the server object by specifying t

he server name.)

The last command performs the following operations:

\* Copies and converts the .vmdk file located at the specified path

("[storage1] /VM02/VM02.vmdk") on the ESX host.

\* Transfers the resulting .vhd file to the specified path (C:\VM02)

on the Windows host. Note that the -Path parameter, when used

with Copy-VMDK, cannot take a UNC path.

IMPORTANT:

The Copy-VMDK cmdlet takes as its input the .vmdk file that the .vmx file p

oints to, as explained in detail in example 1.

REMARKS

For more information, type: "get-help Copy-VMDK -detailed".

For technical information, type: "get-help Copy-VMDK -full".

## VMHost

### Add-VMHost

SYNOPSIS

Adds a computer as a virtual machine host to the Virtual Machine Manager da

tabase.

SYNTAX

Add-VMHost [-ComputerName] <String> -Credential <PSCredential> -VMHostClust

er <VMHostCluster> [-AvailableForPlacement <Boolean>] [-CPUPercentageReserv

e <Int32>] [-Description <String>] [-DiskSpaceReserveMB <Int32>] [-JobVaria

ble <String>] [-MaintenanceHost] [-MaxDiskIOReservation <Int32>] [-MemoryRe

serveMB <Int32>] [-NetworkPercentageReserve <Int32>] [-PROTipID <Guid>] [-R

eassociate] [-RemoteConnectEnabled <Boolean>] [-RemoteConnectMultipleConnec

tionsEnabled <Boolean>] [-RemoteConnectPort <Int32>] [-RemoteConnectTimeout

Enabled <Boolean>] [-RemoteConnectTimeoutMinutes <Int32>] [-RunAsynchronous

ly] [-VMMServer [<String ServerConnection>]] [-VMPaths <String>] [<CommonPa

rameters>]

Add-VMHost [-ComputerName] <String> -Credential <PSCredential> -Virtualizat

ionManager <VirtualizationManager> [-AvailableForPlacement <Boolean>] [-Cer

tificate] [-CPUPercentageReserve <Int32>] [-Description <String>] [-DiskSpa

ceReserveMB <Int32>] [-JobVariable <String>] [-MaintenanceHost] [-MaxDiskIO

Reservation <Int32>] [-MemoryReserveMB <Int32>] [-NetworkPercentageReserve

<Int32>] [-PROTipID <Guid>] [-Reassociate] [-RemoteConnectEnabled <Boolean>

] [-RemoteConnectMultipleConnectionsEnabled <Boolean>] [-RemoteConnectPort

<Int32>] [-RemoteConnectTimeoutEnabled <Boolean>] [-RemoteConnectTimeoutMin

utes <Int32>] [-RunAsynchronously] [-SshPublicKey <ClientSshPublicKey>] [-S

shPublicKeyFile <String>] [-SshTcpPort <Int32>] [-TCPPort <Int32>] [-VMHost

Group <HostGroup>] [-VMMServer [<String ServerConnection>]] [-VMPaths <Stri

ng>] [<CommonParameters>]

Add-VMHost [-ComputerName] <String> [-AvailableForPlacement <Boolean>] [-CP

UPercentageReserve <Int32>] [-Description <String>] [-DiskSpaceReserveMB <I

nt32>] [-JobVariable <String>] [-MaintenanceHost] [-MaxDiskIOReservation <I

nt32>] [-MemoryReserveMB <Int32>] [-NetworkPercentageReserve <Int32>] [-PRO

TipID <Guid>] [-Reassociate] [-RemoteConnectEnabled <Boolean>] [-RemoteConn

ectMultipleConnectionsEnabled <Boolean>] [-RemoteConnectPort <Int32>] [-Rem

oteConnectTimeoutEnabled <Boolean>] [-RemoteConnectTimeoutMinutes <Int32>]

[-RunAsynchronously] [-VMHost [<String Host>]] [-VMMServer [<String ServerC

onnection>]] [-VMPaths <String>] [<CommonParameters>]

Add-VMHost [-ComputerName] <String> -Credential <PSCredential> [-AvailableF

orPlacement <Boolean>] [-CPUPercentageReserve <Int32>] [-Description <Strin

g>] [-DiskSpaceReserveMB <Int32>] [-JobVariable <String>] [-MaintenanceHost

] [-MaxDiskIOReservation <Int32>] [-MemoryReserveMB <Int32>] [-NetworkPerce

ntageReserve <Int32>] [-PROTipID <Guid>] [-Reassociate] [-RemoteConnectEnab

led <Boolean>] [-RemoteConnectMultipleConnectionsEnabled <Boolean>] [-Remot

eConnectPort <Int32>] [-RemoteConnectTimeoutEnabled <Boolean>] [-RemoteConn

ectTimeoutMinutes <Int32>] [-RunAsynchronously] [-VMHostGroup <HostGroup>]

[-VMMServer [<String ServerConnection>]] [-VMPaths <String>] [<CommonParame

ters>]

Add-VMHost [-ComputerName] <String> -Credential <PSCredential> -NonTrustedD

omainHost [-AvailableForPlacement <Boolean>] [-CPUPercentageReserve <Int32>

] [-Description <String>] [-DiskSpaceReserveMB <Int32>] [-JobVariable <Stri

ng>] [-MaintenanceHost] [-MaxDiskIOReservation <Int32>] [-MemoryReserveMB <

Int32>] [-NetworkPercentageReserve <Int32>] [-PROTipID <Guid>] [-Reassociat

e] [-RemoteConnectEnabled <Boolean>] [-RemoteConnectMultipleConnectionsEnab

led <Boolean>] [-RemoteConnectPort <Int32>] [-RemoteConnectTimeoutEnabled <

Boolean>] [-RemoteConnectTimeoutMinutes <Int32>] [-RunAsynchronously] [-VMH

ostGroup <HostGroup>] [-VMMServer [<String ServerConnection>]] [-VMPaths <S

tring>] [<CommonParameters>]

Add-VMHost [-ComputerName] <String> -EncryptionKey <PSCredential> -Perimete

rNetworkHost -SecurityFile <String> [-AvailableForPlacement <Boolean>] [-CP

UPercentageReserve <Int32>] [-Description <String>] [-DiskSpaceReserveMB <I

nt32>] [-JobVariable <String>] [-MaintenanceHost] [-MaxDiskIOReservation <I

nt32>] [-MemoryReserveMB <Int32>] [-NetworkPercentageReserve <Int32>] [-PRO

TipID <Guid>] [-Reassociate] [-RemoteConnectEnabled <Boolean>] [-RemoteConn

ectMultipleConnectionsEnabled <Boolean>] [-RemoteConnectPort <Int32>] [-Rem

oteConnectTimeoutEnabled <Boolean>] [-RemoteConnectTimeoutMinutes <Int32>]

[-RunAsynchronously] [-VMHostGroup <HostGroup>] [-VMMServer [<String Server

Connection>]] [-VMPaths <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Adds one or more computers as virtual machine hosts to the Virtual Machine

Manager database. A virtual machine host (also called a "host") is a physic

al computer managed by Virtual Machine Manager whose role is to host one or

more virtual machines.

TYPES OF VIRTUAL MACHINE HOST SUPPORTED IN VMM 2008

---------------------------------------------------

From the perspective of networking and domains, the types of host that VMM

2008 supports include:

\* Domain-joined Windows host - The host can be located in either a

trusted or untrusted domain.

\* Perimeter network Windows host - A non-domain-joined Windows host

can be managed in the same way as a perimeter network Windows

host that is located in a domain.

\* A VMware ESX host - ESX hosts do not use Windows Active Directory

domains.

From the perspective of virtualization platform and operating system, the t

ypes of host that VMM 2008 supports include:

\* Virtual Server hosts - A Windows-based server running the Microsoft

Virtual Server virtualization service. In VMM 2007, Virtual Server

hosts are the only supported type of host.

\* Hyper-V hosts - A server running Windows Server 2008 with the Hyper-V

role enabled. Hyper-V is the hypervisor-based virtualization feature

introduced in Windows Server 2008.

\* VMware ESX hosts – A VMware ESX server running proprietary software,

including a hypervisor, that is managed by a VMware VirtualCenter

Server running Windows.

VMM 2008 manages all three types of host, even though each host type implem

ents virtualization in a different way. The following sections describe eac

h type of host in more detail.

WHAT YOU NEED TO KNOW BEFORE YOU ADD A VIRTUAL SERVER HOST

----------------------------------------------------------

\* Windows Server 2003 R2, Windows Server 2003 Service Pack 2 (SP2), or

Windows Server 2008 must be installed on the computer that you want to

use as a Virtual Server host. The server operating system on a

Virtual Server host can have either a 32-bit or 64-bit CPU.

\* The current version of Virtual Server is Microsoft Virtual Server 2005

R2 Service Pack 1 (SP1) update version 1.1.629.0. For more information,

see KB 948515 at:

http://support.microsoft.com/kb/948515

If your computer does not have Virtual Server installed or if Virtual

Server 2005 R2 SP1 is installed, Virtual Machine Manager will install

the current version of Virtual Server automatically for you.

\* If you have a version of Virtual Server installed that is

earlier than Virtual Server 2005 R2 SP1, Add-VMHost will fail. You

will need to upgrade your Virtual Server to 2005 R2 SP1 with update

KB 948515, and then retry Add-VMHost.

WHAT YOU NEED TO KNOW BEFORE YOU ADD A HYPER-V HOST

---------------------------------------------------

\* Your computer must meet the following hardware virtualization

assistance requirements:

- 64-bit CPU

- VT (Virtualization Technology feature from Intel) or

AMD-V (from AMD) enabled on the BIOS

- NX/XD (No eXecute / eXecute Disable)

- Full BIOS support for hardware virtualization

\* Your computer must have Windows Server 2008 or later installed.

\* Your computer must have the Hyper-V RTM software update installed.

The Add-VMHost cmdlet will enable the Hyper-V server role for you,

but you must first configure the Virtualization option in the BIOS

manually.

WHAT YOU NEED TO KNOW BEFORE YOU ADD A WINDOWS-BASED PERIMETER NETWORK HOST

---------------------------------------------------------------------------

When you add a Windows-based host located in a perimeter network to Virtual

Machine Manager, you must specify credentials to be used as an encryption

key.

You use the Get-Credential cmdlet to store credentials to use as the encryp

tion key in a variable, and then use the -EncryptionKey parameter in conjun

ction with the -SecurityFile parameter to decrypt the contents of the secur

ity file associated with the encryption key.

Example format: -SecurityFile "C:\SecurityFile.txt" -EncryptionKey $Key

The user name stored in the variable (such as $Key shown in the example) ca

n be any user name, but the password must be the same encryption key that w

as used when the Virtual Machine Manager agent was installed on the perimet

er network computer.

The source of the encryption key and of the path to the SecurityFile.txt fi

le are as follows:

\* To manage a Windows-based host in a perimeter network, you must first

install the Virtual Machine Manager agent locally on that host.

\* When you run VMM Setup and choose the Setup option to install the host

in a perimeter network, the Setup wizard prompts you to:

- Provide an encryption key for the security file.

- Specify where you want to store (export) the security file.

\* After you run Setup and then use the Add-VMHost cmdlet to add a host

located in a perimeter network to the Virtual Machine Manager database,

you must specify this same encryption key and security file (as shown in

the example format) to complete the operation.

WHAT YOU NEED TO KNOW BEFORE YOU ADD A VMWARE ESX HOST

------------------------------------------------------

\* You must have added at least one VMware VirtualCenter Server to VMM.

\* Your VMware VirtualCenter Server must be running version 2.0 or above.

\* Your VMware ESX host must be running version 3.0, 3.5, or 3i.

For more information about hosts, type:

Get-Help about\_VMM\_2008\_Multiple\_Host\_Platforms

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-EncryptionKey <PSCredential>

Specifies credentials to be used as an encryption key when you add a Wi

ndows-based host (a Hyper-V host or a Virtual Server host) located in a

perimeter network to VMM.

Example format: -SecurityFile "C:\SecurityFile.txt" -EncryptionKey $Key

-NonTrustedDomainHost

Specifies that the host to be added to VMM belongs to a non-trusted dom

ain.

-PerimeterNetworkHost

Specifies that this host is located in a perimeter network.

-SecurityFile <String>

Specifies the path to a file that contains the certificate and credenti

als to use for authentication of a Hyper-V host or a Virtual Server hos

t located in a perimeter network.

Example format: -SecurityFile "C:\SecurityFile.txt" -EncryptionKey $Key

Note: This parameter does not apply to a VMware ESX host.

-VirtualizationManager <VirtualizationManager>

Specifies a virtualization manager object currently managed by VMM. VMM

2008 supports the following non-Microsoft virtualization managers:

-VMware VirtualCenter 2.0: manages hosts running ESX Server 3.0 or 3.5

-VMware VirtualCenter 2.5: manages hosts running ESX Server 3.0, 3.5, o

r 3i

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

-AvailableForPlacement <Boolean>

Specifies (when set to TRUE) that the VMM placement process will consid

er this host server (when used with Add-VMHost or Set-VMHost) or this v

olume on a host (when used with Set-VMHostVolume) to be eligible as a p

ossible location on which to deploy virtual machines. If this parameter

is set to FALSE, you can, optionally, choose to deploy virtual machine

s on this host or volume anyway. The default value is TRUE.

-Certificate

Specifies a security certificate object.

-CPUPercentageReserve <Int32>

Specifies the percentage of CPU to set aside for the use of the host op

erating system on the physical host computer. If you do not use this pa

rameter to specify the reserve, the default setting for the host group

is used: 10 percent. The Virtual Machine Manager placement process will

not recommend that a virtual machine be placed on a host unless its re

source requirements can be met without using host reserves.

-Description <String>

Specifies a description for the specified object.

-DiskSpaceReserveMB <Int32>

Specifies, in megabytes (MB), the amount of disk space to set aside for

the use of the host operating system on the physical host computer. If

you do not use this parameter to specify the reserve, the default sett

ing for the host group is used: 100 MB. The Virtual Machine Manager pla

cement process will not recommend that a virtual machine be placed on a

host unless its resource requirements can be met without using host re

serves.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-MaintenanceHost

This parameter is obsolete. Use AvailableForPlacement instead.

-MaxDiskIOReservation <Int32>

Specifies the maximum number of disk I/O operations per second (IOPS) f

or the operating system on the physical host computer. If you do not us

e this parameter to specify the reserve, the default setting for the ho

st group is used: 10000 IOPS. The Virtual Machine Manager placement pro

cess will not recommend that a virtual machine be placed on a host unle

ss its resource requirements can be met without using host reserves.

-MemoryReserveMB <Int32>

Specifies, in megabytes (MB), the amount of memory to set aside for the

use of the host operating system on the physical host computer. If you

do not use this parameter to specify the reserve, the default setting

for the host group is used: 256 MB. The Virtual Machine Manager placeme

nt process will not recommend that a virtual machine be placed on a hos

t unless its resource requirements can be met without using host reserv

es.

-NetworkPercentageReserve <Int32>

Specifies the percentage of network capacity to set aside for the use o

f the host operating system on the physical host computer. If you do no

t use this parameter to specify the reserve, the default setting for th

e host group is used: 10 percent. The Virtual Machine Manager placement

process will not recommend that a virtual machine be placed on a host

unless its resource requirements can be met without using host reserves

.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-Reassociate

Reassociates a host currently managed by one VMM server with another VM

M server.

-RemoteConnectEnabled <Boolean>

Enables (when set to TRUE) a connection on a host server that lets user

s connect to their virtual machines remotely.

TYPE OF HOST TYPE OF REMOTE CONNECTION

------------ -------------------------

Virtual Server Enables Virtual Machine Remote Control (VMRC)

Hyper-V Enables Virtual Machine Connection (VMConnect)

VMware ESX This parameter does not apply to VMs on ESX hosts

Note: In VMM 2007, this parameter was named VMRCEnabled.

-RemoteConnectMultipleConnectionsEnabled <Boolean>

Enables (when set to TRUE) two or more simultaneous remote connections

between virtual machine and its Virtual Server host. This lets users es

tablish multiple remote connections to the same virtual machine on a Vi

rtual Server host. This parameter does not apply to Hyper-V hosts or VM

ware ESX Server hosts.

Note: In VMM 2007, this parameter was named VMRCMultipleConnectionsEnab

led.

-RemoteConnectPort <Int32>

Specifies a default value for the TCP port to use when a remote user co

nnects to a virtual machine. Typically, the default port for a Virtual

Server host is 5900 (for a VMRC connection), and the default port for a

Hyper-V host is 2179 (for a VMConnect connection). This parameter does

not apply to a VMware ESX host.

Note: In VMM 2007, the Virtual Server default remote connect port param

eter was named VMRCPort.

-RemoteConnectTimeoutEnabled <Boolean>

Enables (when set to TRUE) timing out a remote connection between a vir

tual machine and its a Virtual Server host. This parameter does not app

ly to Hyper-V hosts or VMware ESX Server hosts.

Note: In VMM 2007, this parameter was named VMRCTimeoutEnabled.

-RemoteConnectTimeoutMinutes <Int32>

Specifies the number of minutes that a remote connection between a virt

ual machine and its Virtual Server host can remain idle before it is di

sconnected. This parameter does not apply to Hyper-V hosts or VMware ES

X Server hosts.

Note: In VMM 2007, this parameter was named VMRCTimeoutMinutes.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SshPublicKey <ClientSshPublicKey>

Specifies the public key used by Secure Shell (SSH) communications.

-SshPublicKeyFile <String>

Specifies the path to the public key file for establishing a secured SS

H channel with the target hosts.

-SshTcpPort <Int32>

Specifies the TCP port number used by the Secure Shell (SSH) protocol.

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VMPaths <String>

Specifies a set of default paths (as strings separated by the pipeline

operator) on a host where virtual machine files can be stored.

Example format: -VMPaths "C:\Folder1|C:\Folder2|C:\Folder3"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a host in the same domain as the VMM server to VMM.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Add-VMHost "VMHost01.Contoso.com" -Description "This is my new host

" -RemoteConnectEnabled $TRUE -RemoteConnectPort 5900 -Credential $Credenti

al

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are either a local Administrator account

or a domain account with administrator rights on the computer that you want

to add as a host.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following command uses this

server by default.

The last command adds the host object that represents VMHost01 in the Conto

so domain to the VMM database as a managed host, specifies a description, e

nables remote connections, and specifies that TCP port 5900 will be used fo

r remote connections to VMHost01. As the last command is processed, $Creden

tial provides your credentials to Add-VMHost.

2: Add a host located in a perimeter network to VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Key = Get-Credential

PS C:\> Add-VMHost "VMHost02" -Description "This is my new perimeter networ

k host" -RemoteConnectEnabled $FALSE -PerimeterNetworkHost -SecurityFile "C

:\SecurityFile.txt" -EncryptionKey $Key

The first command connects to VMMServer1.

The second command prompts you for a user name and password and stores the

credentials in $Key. The user name can be any user name, but the password m

ust be the same encryption key that was used when the VMM agent was install

ed on this computer. The VMM agent must be installed locally on a computer

located in a perimeter network by choosing the local agent option when you

run Setup. You specify the encryption key for the security file on the Secu

rity File Folder page of the System Center Virtual Machine Manager Agent Se

tup wizard.

The last command adds a host object that represents the computer named VMHo

st02 to the VMM database as a managed host. The command adds a description,

disables remote connections, and specifies that this host is located in a

perimeter network. This command uses the credentials stored in $Key to decr

ypt the contents of SecurityFile.txt (which, in this example, is located at

C:\) and then uses the contents of SecurityFile.txt to authenticate the ne

w host.

3: Add a host located in a non-trusted domain to VMM.

PS C:\> $Credential = Get-Credential

PS C:\> Add-VMHost "VMHost03.NonTrustedDomain.com" -VMMServer VMMServer1.Co

ntoso.com –NonTrustedDomainHost –Credential $Credential

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in $Credential. The required creden

tials for this operation are an account with administrator rights to add a

host located in the non-trusted domain to the VMM server in the Contoso.com

domain.

The second command adds VMHost03, located in a domain that is not trusted b

y Contoso.com, to the VMM database as a managed host. As this command is pr

ocessed, $Credential provides your credentials to Add-VMHost.

4: Add an ESX Server as a host to VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Manager = Get-VirtualizationManager -ComputerName "VirtualCenterSe

rver04"

PS C:\> $HostGroup = Get-VMHostGroup "HostGroup04"

PS C:\> $Credential = Get-Credential

PS C:\> Add-VMHost -ComputerName "nnn.nnn.nnn.nnn" –Credential $Credential

–VMHostGroup $HostGroup –VirtualizationManager $Manager

The first command connects to VMMServer1.

The second command gets the object that represents VirtualCenterServer04 an

d stores the object in $Manager.

The third command gets the object that represents HostGroup04 and stores th

e object in $HostGroup.

The fourth command uses Get-Credential to prompt you to supply a user name

and password and stores your credentials in $Credential. The required crede

ntials for this operation are an account with administrator rights to add a

n ESX server managed by VirtualCenterServer04 to the VMM database.

The last command adds an ESX Server, which is managed by VirtualCenterServe

r04, as a host to HostGroup04 in VMM. The command identifies the ESX server

by specifying its IP address (indicated by "nnn.nnn.nnn.nnn") and provides

the credentials stored in $Credential required to let you add this server

to VMM.

REMARKS

For more information, type: "get-help Add-VMHost -detailed".

For technical information, type: "get-help Add-VMHost -full".

### Associate-VMHost

SYNOPSIS

Associates a VMware ESX Server with Virtual Machine Manager as a virtual ma

chine host and specifies credentials to manage this host.

SYNTAX

Associate-VMHost [-VMHost] [<String Host>] [-Certificate] [-Credential <PSC

redential>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously]

[-SshPublicKey <ClientSshPublicKey>] [-SshPublicKeyFile <String>] [-SshTcp

Port <Int32>] [-TCPPort <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Associates a VMware ESX Server with Virtual Machine Manager as a virtual ma

chine host and specifies the credentials to use with this ESX host when it

is managed by Virtual Machine Manager.

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-Certificate

Specifies a security certificate object.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SshPublicKey <ClientSshPublicKey>

Specifies the public key used by Secure Shell (SSH) communications.

-SshPublicKeyFile <String>

Specifies the path to the public key file for establishing a secured SS

H channel with the target hosts.

-SshTcpPort <Int32>

Specifies the TCP port number used by the Secure Shell (SSH) protocol.

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Set the credentials for a specific VMware ESX Server host.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $ESXHost = Get-VMHost -ComputerName "nnn.nnn.nnn.nnn"

PS C:\> Associate-VMHost -VMHost $ESXHost -Credential $Credential

The first command uses Get-Credential to prompt you to supply a user name a

nd password, and stores these credentials in variable $Credential. The requ

ired credentials for this operation are either a root account (root/<passwo

rd>) or the account for the VMware delegated administrator defined earlier

in VirtualCenter Server for this ESX host.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands use this

server by default.

The third command gets the object that represents a VMware ESX Server named

ESXHost01 by specifying its IP address (indicated by "nnn.nnn.nnn.nnn") an

d stores the ESX host object in variable $ESXHost.

The last command associates this VMware ESX Server with VMM as a managed ho

st, and it specifies that the credentials used to access ESXHost01 are thos

e stored in $Credential.

NOTE: You can use this command to set the credentials initially, when VMM f

irst starts to manage this server; or, alternatively, to change the credent

ials at a later time.

2: Set the credentials, public key, and certificate for a specific VMware E

SX Server host.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $ESXHost = Get-VMHost -ComputerName "nnn.nnn.nnn.nnn"

PS C:\> $Cert = Get-Certificate -ComputerName "nnn.nnn.nnn.nnn"

PS C:\> $PublicKey = Get-SshPublicKey -ComputerName "nnn.nnn.nnn.nnn"

PS C:\> Associate-VMHost -VMHost $ESXHost -Credential $Credential -Certific

ate $Cert -SshPublicKey $PublicKey

The first command uses Get-Credential to prompt you to supply an appropriat

e user name and password, and stores these credentials in $Credential. The

required credentials for this operation are either a root account (root/<pa

ssword>) or the account for the VMware delegated administrator defined earl

ier in VirtualCenter Server for this ESX host..

The second command connects to VMMServer1.

The third command gets the object that represents a VMware ESX Server named

ESXHost02 by specifying its IP address (indicated by "nnn.nnn.nnn.nnn") an

d stores the ESX host object in $ESXHost.

The fourth command uses the Get-Certificate cmdlet to get the certificate o

bject from ESXHost02 and stores the certificate object in $Cert.

The fifth command uses the Get-SshPublicKey cmdlet to get the public key ob

ject from ESXHost02 and stores the public key object in $PublicKey.

The last command associates this VMware ESX Server with VMM as a managed ho

st and specifies that the credentials used to access ESXHost02 are those st

ored in $Credential.

NOTE: You can use this command to set the credentials initially, when VMM f

irst starts to manage this server; or, alternatively, to change the credent

ials at a later time.

REMARKS

For more information, type: "get-help Associate-VMHost -detailed".

For technical information, type: "get-help Associate-VMHost -full".

### Get-VMHost

SYNOPSIS

Gets virtual machine host objects from the Virtual Machine Manager database

.

SYNTAX

Get-VMHost [[-ComputerName] <String>] [-ID <Guid>] [-VMMServer [<String Ser

verConnection>]] [<CommonParameters>]

Get-VMHost [[-ComputerName] <String>] -VMHostCluster <VMHostCluster> [<Comm

onParameters>]

Get-VMHost [[-ComputerName] <String>] -VMHostGroup <HostGroup> [<CommonPara

meters>]

DETAILED DESCRIPTION

Gets one or more objects that represent virtual machine hosts from the Virt

ual Machine Manager database. Virtual machine hosts are physical computers

that are managed by Virtual Machine Manager on which you can deploy virtual

machines.

Virltual Machine Manager supports three types of host:

\* Virtual Server hosts

\* Hyper-V hosts

\* VMware ESX Server hosts

For more information about virtual machine hosts in Virtual Machine Manager

, type:

Get-Help Add-VMHost -detailed

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all hosts managed by this VMM server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets all host objects for all hosts (on which you deploy

virtual machines) currently managed by VMMServer1 and displays the host pr

operties to the user.

2: Get a host by name.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHost -ComputerName "VMHost01.Contoso.com"

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost01

in the Contoso.com domain and displays the host properties to the user.

3: Get all hosts in a specific host group and display information about the

m to the user.

PS C:\> $LabHG = Get-VMHostGroup -Name "Lab" -VMMServer VMMServer1.Contoso.

com

PS C:\> $HostsInLabHG = Get-VMHost -VMHostGroup $LabHG

PS C:\> $HostsInLabHG | Format-Table -property Name, VMs

The first command gets the object that represents the host group named Lab

from VMMServer1 and stores the host group object in $LabHG.

The second command gets all objects that represent hosts in the Lab host gr

oup and stores the host objects in $HostsInLabHG.

The last command passes all host objects stored in $HostsInLabHG to the For

mat-Table cmdlet, which displays the name of each host and the virtual mach

ines deployed on that host in a table.

4: Get all hosts in a specific host cluster and display information about t

hem to the user.

PS C:\> $Cluster = Get-VMHostCluster -Name "Cluster04.Contoso.com" -VMMServ

er VMMServer1.Contoso.com

PS C:\> $HostsInCluster = Get-VMHost -VMHostCluster $Cluster

PS C:\> $HostsInCluster | Format-Table -property Name, VirtualizationPlatfo

rm

The first command gets the object that represents the host cluster named Cl

uster04 from VMMServer1 and stores the cluster object in $Cluster.

The second command gets all objects that represent hosts in Cluster04 and s

tores the host objects in $HostsInCluster.

The last command passes all host objects stored in $HostsInCluster to the F

ormat-Table cmdlet, which displays the name and virtualization platform of

each host in Cluster04.

5: Get a specific host located on a perimeter network by its IP address.

PS C:\> $VMHost = Get-VMHost -ComputerName 10.199.53.5 -VMMServer VMMServer

1.Contoso.com

PS C:\> $VMHost | Select-Object -property ComputerName, OperatingSystem

The first command gets the object that represents a host located on a perim

eter network whose IP address is 10.199.53.5 from VMMServer1 and stores the

host object in $VMHost.

The second command passes the host object in $VMHost to the Select-Object c

mdlet, which displays the computer name and operating system for this host.

REMARKS

For more information, type: "get-help Get-VMHost -detailed".

For technical information, type: "get-help Get-VMHost -full".

### Move-VMHost

SYNOPSIS

Moves a virtual machine host object managed by Virtual Machine Manager from

one host group to another.

SYNTAX

Move-VMHost [-VMHost] [<String Host>] -ParentHostGroup [<String HostGroup>]

[-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonPa

rameters>]

DETAILED DESCRIPTION

Moves one or more objects that represent virtual machine hosts managed by V

irtual Machine Manager from the current host group to a new parent host gro

up. Before you can move a host, its new parent host group must exist.

If the host is a computer that is managed by members of a Self Service User

or Delegated Administrator user role, moving the host from one host group

to another might affect the roles that have access to the host or to virtua

l machines on that host.

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-ParentHostGroup [<String HostGroup>]

Specifies the parent host group that contains one or more hosts, host g

roups, or host clusters.

Note: In VMM 2007, this parameter was named ParentVMHostGroup.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Move a single host to a new parent host group.

PS C:\> Get-VMMServer –ComputerName "VMMServer1.Contoso.com"

PS C:\> $MyHost = Get-VMHost | where { $\_.Name -eq "VMHost01" }

PS C:\> $NewHG = Get-VMHostGroup | where {$\_.Name -eq "MyNewGroup"}

PS C:\> Move-VMHost -VMHost $MyHost -ParentHostGroup $NewHG

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents a host named VMHost01 fr

om the VMM database and stores the host object in variable $MyHost.

The third command gets the object that represents the host group named MyNe

wGroup and stores the host group object in variable $NewHG.

The last command moves VMHost01 (represented by $MyHost) from its current l

ocation to MyNewGroup (represented by $NewHG).

2: Move all hosts to a new parent host group.

PS C:\> Get-VMMServer –ComputerName VMMServer1.Contoso.com

PS C:\> $AllHosts = Get-VMHost

PS C:\> $NewHG = Get-VMHostGroup | where { $\_.Name -eq "MyNewGroup" }

PS C:\> $AllHosts | Move-VMHost -ParentHostGroup $NewHG

The first command connects to VMMServer1.

The second command gets all objects that represent hosts and stores the hos

t objects in $AllHosts.

The third command gets the object that represents the host group named MyNe

wGroup and stores the host group object in $NewHG.

The last command moves each host object in $AllHosts to a new parent host g

roup called MyNewGroup (represented by $NewHG).

3: Move a set of hosts from one host group to a new parent host group.

PS C:\> Get-VMMServer –ComputerName "VMMServer1.Contoso.com"

PS C:\> $SpecificHosts = Get-VMHost | where { $\_.VMHostGroup -like "\*OldGro

up\*" }

PS C:\> $NewHG = Get-VMHostGroup | where { $\_.Name -eq "MyNewGroup" }

PS C:\> $SpecificHosts | Move-VMHost -ParentHostGroup $NewHG

The first command connects to VMMServer1.

The second command gets all objects that represent hosts whose host group c

ontains the string "OldGroup" and stores these host objects in $SpecificHos

ts.

The third command gets the object that represents the host group named MyNe

wGroup and stores the host group object in $NewHG.

The last command moves each host object in $SpecificHosts to a new parent h

ost group called MyNewGroup (represented by $NewHG).

REMARKS

For more information, type: "get-help Move-VMHost -detailed".

For technical information, type: "get-help Move-VMHost -full".

### Refresh-VMHost

SYNOPSIS

Refreshes virtual machine host properties in the Virtual Machine Manager Ad

ministrator Console.

SYNTAX

Refresh-VMHost [-VMHost] [<String Host>] [-JobVariable <String>] [-PROTipID

<Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Refreshes the properties of a virtual machine host so that the Virtual Mach

ine Manager Administrator Console displays updated information about the ho

st.

Host properties that this cmdlet updates include:

- Name

- Operating system

- Status (such as Responding)

- Host volumes (typically, addition or removal of drive letters,

mount points, as well as used and available space)

- Network adapters (addition or removal)

- If the host is a Virtual Server host, changes in VMRC settings

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Refresh information about a specific host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> Refresh-VMHost -VMHost $VMHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost.

The last command refreshes the properties for VMHost01 so that current info

rmation about this host will appear in the Administrator Console.

2: Refresh information about all hosts.

Get-VMHost -VMMServer VMMServer1.Contoso.com | Refresh-VMhost

Refreshes information about all hosts currently managed by VMMServer1 so th

at current information about each host will appear in the Administrator Con

sole.

REMARKS

For more information, type: "get-help Refresh-VMHost -detailed".

For technical information, type: "get-help Refresh-VMHost -full".

### Remove-VMHost

SYNOPSIS

Removes a virtual machine host object from Virtual Machine Manager.

SYNTAX

Remove-VMHost [-VMHost] [<String Host>] [-Confirm] [-Credential <PSCredenti

al>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<Comm

onParameters>]

Remove-VMHost [-VMHost] [<String Host>] -Force [-Confirm] [-JobVariable <St

ring>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent virtual machine hosts from the V

irtual Machine Manager database.

The Remove-VMHost cmdlet operates as follows:

\* HOST SERVER ONLY - If this computer is a Windows-based virtual machine

host server (Virtual Server or Hyper-V) but is not also a library

server, the host is removed. That is, the host object is removed from

the Virtual Machine Manager database, and the Virtual Machine Manager

agent software is uninstalled from the physical host server.

If the host is a VMware ESX host, the host object is removed from the

Virtual Machine Manager database. Virtual Machine Manager does not

install an agent on ESX hosts.

\* HOST AND LIBRARY SERVER - If this computer is a Windows-based virtual

machine host (Virtual Server or Hyper-V) and is also a library server,

this command removes only the host functionality but leaves the

library server component in place. That is, the host object is

removed from the Virtual Machine Manager database, but the Virtual

Machine Manager agent software is not uninstalled from the physical

server. The library server object remains in the database.

If the host is a VMware ESX host, it can function only as a virtual

machine host in Virtual Machine Manager. It cannot be both a host

and a library server.

\* CREDENTIALS - If a Windows-based host (Virtual Server or Hyper-V) is

joined to an Active Directory domain, you must provide credentials for

an account with appropriate permissions to remove that host computer

from Virtual Machine Manager.

You do not need to provide Active Directory credentials to remove a

perimeter network host or VMware ESX host from Virtual Machine Manager.

\* VIRTUAL MACHINES - When you remove a host, the host is no longer

managed by Virtual Machine Manager. However, any virtual machines

currently on the host server will not be removed or disassociated

from the server. Any running virtual machines are not shut down.

Although the virtual machines are no longer managed by Virtual

Machine Manager, they are not affected in any other way.

\* FORCED REMOVAL - You can use the Force parameter with the Remove-VMHost

cmdlet to remove a virtual machine host from Virtual Machine Manager

when you do not have appropriate credentials to manage that host or

when the Virtual Machine Manager server can no longer communicate with

that host.

When you specify the Force parameter, Virtual Machine Manager will

not ask or check for credentials, nor will Virtual Machine Manager

attempt to connect to the host and uninstall the Virtual Machine

Manager agent. Hence, using the Force parameter is recommended only

when cleaning up stale host records from the Virtual Machine Manager

database.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-Confirm

Prompts for confirmation before running the command.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific domain-joined host from VMM.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> Remove-VMHost -VMHost $VMHost -Credential $Credential -Confirm

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are a domain account with administrator r

ights to remove a Windows-based host server (a Virtual Server host or a Hyp

er-V host) joined to an Active Directory domain from VMM.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands use this

server by default.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in variable $VMHost.

The last command removes the VMHost01 object from the VMM database. As this

command is processed, $Credential provides your credentials to Remove-VMHo

st, and the Confirm parameter prompts you to confirm that you do want to re

move this host from VMM.

2: Remove all hosts that are not nodes in a host cluster from VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHost | where {$\_.HostCluster -eq $NULL} | where {$\_.Virtualiz

ationPlatform -eq "VMwareESX" -or $\_.PerimeterNetworkHost -eq 1 -or $\_.Non

TrustedDomainHost -eq 1} | Remove-VMHost -Confirm

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMHost | where {$\_.HostCluster -eq $NULL -and $\_.Virtualization

Platform -ne "VMwareESX" -and $\_.PerimeterNetworkHost -eq 0 -and $\_.NonTru

stedDomainHost -eq 0} | Remove-VMHost -Credential $Credential -Confirm

The first command connects to VMMServer1.

The second command gets all host objects; excludes any hosts that are nodes

in a host cluster; selects only those objects that represent VMware ESX ho

sts, perimeter network hosts, or non-trusted domain hosts; and removes thos

e objects from VMM if you confirm that you want to remove them. Credentials

are not required to remove these hosts.

The third command prompts you to supply a user name and password for an acc

ount with permissions to remove domain-joined Windows hosts from VMM and st

ores your credentials in $Credential.

The last command selects the objects that represent all hosts except cluste

red hosts, ESX hosts, perimeter network hosts, or non-trusted domain hosts

-- that is, the command selects domain-joined Windows-based hosts that are

not part of a host cluster -- and passes these objects to the Remove-VMHost

cmdlet. As this command is processed, $Credential provides your credential

s to Remove-VMHost, and the Confirm parameter prompts you to confirm that y

ou do want to remove these hosts from VMM.

3: Remove a specific host that you can no longer access from VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03"

PS C:\> Remove-VMHost -VMHost $VMHost -Force -Confirm

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost03

and stores the host object in $VMHost.

The last command switches on the Force parameter to ensure that the object

that represents VMHost03 is removed from the VMM database. Credentials are

not needed for this operation. The Confirm parameter prompts you to confirm

that you do want to remove this host.

NOTE:

You can use the Force parameter to remove a host from VMM when you do not h

ave the credentials for that host or when the VMM server can no longer comm

unicate with that host.

REMARKS

For more information, type: "get-help Remove-VMHost -detailed".

For technical information, type: "get-help Remove-VMHost -full".

### Set-VMHost

SYNOPSIS

Changes properties of a virtual machine host managed by Virtual Machine Man

ager.

SYNTAX

Set-VMHost [-VMHost] [<String Host>] [-AvailableForPlacement <Boolean>] [-C

PUPercentageReserve <Int32>] [-Custom1 <String>] [-Custom10 <String>] [-Cus

tom2 <String>] [-Custom3 <String>] [-Custom4 <String>] [-Custom5 <String>]

[-Custom6 <String>] [-Custom7 <String>] [-Custom8 <String>] [-Custom9 <Stri

ng>] [-Description <String>] [-DiskSpaceReserveMB <Int32>] [-JobGroup <Guid

>] [-JobVariable <String>] [-MaintenanceHost] [-MaxDiskIOReservation <Int32

>] [-MemoryReserveMB <Int32>] [-NetworkPercentageReserve <Int32>] [-PROTipI

D <Guid>] [-RemoteConnectCertificatePath <String>] [-RemoteConnectEnabled <

Boolean>] [-RemoteConnectMultipleConnectionsEnabled <Boolean>] [-RemoteConn

ectPort <Int32>] [-RemoteConnectTimeoutEnabled <Boolean>] [-RemoteConnectTi

meoutMinutes <Int32>] [-RemoveRemoteConnectCertificate <Boolean>] [-RunAsyn

chronously] [-SecureRemoteConnectEnabled <Boolean>] [-VMHostManagementCrede

ntial <PSCredential>] [-VMPaths <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a virtual machine host managed by Virtual

Machine Manager. Settings that you can modify with the Set-VMHost cmdlet a

re summarized as follows.

AVAILABILITY AS A HOST FOR VIRTUAL MACHINES

-------------------------------------------

You can specify whether a host is currently considered by the Virtual Machi

ne Manager placement process as a candidate on which to place virtual machi

nes.

HOST RESERVE SETTINGS

---------------------

You can configure the following host reserve settings:

\* Percentage of CPU usage to set aside for use by the host.

\* Amount of disk space (MB) to set aside for use by the host.

\* Maximum number of disk I/O operations per second (IOPS) to set asside for

use by the host.

\* Amount of memory (MB) to set asside for use by the host.

\* Percentage of network capacity to set aside for use by the host.

The Virtual Machine Manager placement process will not recommend placing a

virtual machine on a host unless the resource requirements of the virtual m

achine can be met without using the host reserves. If you do not specify re

serve settings, Virtual Machine Manager uses default settings.

VIRTUAL MACHINE PATHS

---------------------

You can specify, as a set of default paths, locations on a host where virtu

al machine files can be stored.

CREDENTIAL FOR MANAGING HOSTS IN A PERIMETER NETWORK OR NON-TRUSTED DOMAIN

--------------------------------------------------------------------------

You can specify the password for an account used to manage Hyper-V or Virtu

al Server hosts that are located in a perimeter network or in a non-trusted

domain.

REMOTE CONNECTION SETTINGS

--------------------------

You can configure remote connection settings for Hyper-V hosts (VMConnect)

or Virtual Server hosts (VMRC) that enable users to connect to virtual mach

ines remotely. This setting does not apply to virtual machines on VMware ES

X hosts.

For Virtual Server hosts only, you can configure a set of additional remote

connection settings, including specifying timeout settings, enabling multi

ple connections, and configuring settings that enable secure connections.

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-AvailableForPlacement <Boolean>

Specifies (when set to TRUE) that the VMM placement process will consid

er this host server (when used with Add-VMHost or Set-VMHost) or this v

olume on a host (when used with Set-VMHostVolume) to be eligible as a p

ossible location on which to deploy virtual machines. If this parameter

is set to FALSE, you can, optionally, choose to deploy virtual machine

s on this host or volume anyway. The default value is TRUE.

-CPUPercentageReserve <Int32>

Specifies the percentage of CPU to set aside for the use of the host op

erating system on the physical host computer. If you do not use this pa

rameter to specify the reserve, the default setting for the host group

is used: 10 percent. The Virtual Machine Manager placement process will

not recommend that a virtual machine be placed on a host unless its re

source requirements can be met without using host reserves.

-Custom1 <String>

Specifies a custom property on a VMM object.

-Custom10 <String>

Specifies a custom property on a VMM object.

-Custom2 <String>

Specifies a custom property on a VMM object.

-Custom3 <String>

Specifies a custom property on a VMM object.

-Custom4 <String>

Specifies a custom property on a VMM object.

-Custom5 <String>

Specifies a custom property on a VMM object.

-Custom6 <String>

Specifies a custom property on a VMM object.

-Custom7 <String>

Specifies a custom property on a VMM object.

-Custom8 <String>

Specifies a custom property on a VMM object.

-Custom9 <String>

Specifies a custom property on a VMM object.

-Description <String>

Specifies a description for the specified object.

-DiskSpaceReserveMB <Int32>

Specifies, in megabytes (MB), the amount of disk space to set aside for

the use of the host operating system on the physical host computer. If

you do not use this parameter to specify the reserve, the default sett

ing for the host group is used: 100 MB. The Virtual Machine Manager pla

cement process will not recommend that a virtual machine be placed on a

host unless its resource requirements can be met without using host re

serves.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-MaintenanceHost

This parameter is obsolete. Use AvailableForPlacement instead.

-MaxDiskIOReservation <Int32>

Specifies the maximum number of disk I/O operations per second (IOPS) f

or the operating system on the physical host computer. If you do not us

e this parameter to specify the reserve, the default setting for the ho

st group is used: 10000 IOPS. The Virtual Machine Manager placement pro

cess will not recommend that a virtual machine be placed on a host unle

ss its resource requirements can be met without using host reserves.

-MemoryReserveMB <Int32>

Specifies, in megabytes (MB), the amount of memory to set aside for the

use of the host operating system on the physical host computer. If you

do not use this parameter to specify the reserve, the default setting

for the host group is used: 256 MB. The Virtual Machine Manager placeme

nt process will not recommend that a virtual machine be placed on a hos

t unless its resource requirements can be met without using host reserv

es.

-NetworkPercentageReserve <Int32>

Specifies the percentage of network capacity to set aside for the use o

f the host operating system on the physical host computer. If you do no

t use this parameter to specify the reserve, the default setting for th

e host group is used: 10 percent. The Virtual Machine Manager placement

process will not recommend that a virtual machine be placed on a host

unless its resource requirements can be met without using host reserves

.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RemoteConnectCertificatePath <String>

Specifies the path to the certificate to use for a secure remote connec

tion between a user and a virtual machine on a Virtual Server host. To

generate a request for a signed certificate to use with this parameter,

use the New-VMRCCertificateRequest cmdlet. Use the file generated by t

his cmdlet to submit a request to your Certification Authority (CA), an

d then, after receiving a certificate, use the certificate path with th

is parameter. This parameter does not apply to Hyper-V hosts or VMware

ESX Server hosts.

Note: In VMM 2007, this parameter was named VMRCCertificatePath.

-RemoteConnectEnabled <Boolean>

Enables (when set to TRUE) a connection on a host server that lets user

s connect to their virtual machines remotely.

TYPE OF HOST TYPE OF REMOTE CONNECTION

------------ -------------------------

Virtual Server Enables Virtual Machine Remote Control (VMRC)

Hyper-V Enables Virtual Machine Connection (VMConnect)

VMware ESX This parameter does not apply to VMs on ESX hosts

Note: In VMM 2007, this parameter was named VMRCEnabled.

-RemoteConnectMultipleConnectionsEnabled <Boolean>

Enables (when set to TRUE) two or more simultaneous remote connections

between virtual machine and its Virtual Server host. This lets users es

tablish multiple remote connections to the same virtual machine on a Vi

rtual Server host. This parameter does not apply to Hyper-V hosts or VM

ware ESX Server hosts.

Note: In VMM 2007, this parameter was named VMRCMultipleConnectionsEnab

led.

-RemoteConnectPort <Int32>

Specifies a default value for the TCP port to use when a remote user co

nnects to a virtual machine. Typically, the default port for a Virtual

Server host is 5900 (for a VMRC connection), and the default port for a

Hyper-V host is 2179 (for a VMConnect connection). This parameter does

not apply to a VMware ESX host.

Note: In VMM 2007, the Virtual Server default remote connect port param

eter was named VMRCPort.

-RemoteConnectTimeoutEnabled <Boolean>

Enables (when set to TRUE) timing out a remote connection between a vir

tual machine and its a Virtual Server host. This parameter does not app

ly to Hyper-V hosts or VMware ESX Server hosts.

Note: In VMM 2007, this parameter was named VMRCTimeoutEnabled.

-RemoteConnectTimeoutMinutes <Int32>

Specifies the number of minutes that a remote connection between a virt

ual machine and its Virtual Server host can remain idle before it is di

sconnected. This parameter does not apply to Hyper-V hosts or VMware ES

X Server hosts.

Note: In VMM 2007, this parameter was named VMRCTimeoutMinutes.

-RemoveRemoteConnectCertificate <Boolean>

Removes the current VMRC-based remote connect certificate for virtual m

achines on Virtual Server hosts. This parameter does not apply to Hyper

-V hosts or VMware ESX Server hosts.

Note: In VMM 2007, this parameter was named RemoveVMRCCertificate.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SecureRemoteConnectEnabled <Boolean>

Enables (when set to TRUE) a secure remote connection between a user an

d a virtual machine on a Virtual Server host. By default, an unsigned c

ertificate is used to encrypt the remote connection. Alternatively, you

can use this parameter with the RemoteConnectCertificatePath parameter

to specify a signed certificate. This parameter does not apply to Hype

r-V hosts or VMware ESX Server hosts.

Note: In VMM 2007, this parameter was named SecureVMRCEnabled.

-VMHostManagementCredential <PSCredential>

Specifies the password for an account that has permission to manage Win

dows-based hosts managed by VMM that are located in a perimeter network

or in a non-trusted domain.

Note: In VMM 2007, this parameter was named PerimeterNetworkHostCredent

ial.

-VMPaths <String>

Specifies a set of default paths (as strings separated by the pipeline

operator) on a host where virtual machine files can be stored.

Example format: -VMPaths "C:\Folder1|C:\Folder2|C:\Folder3"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Make a host available for placement.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> Set-VMHost -VMHost $VMHost -AvailableForPlacement $TRUE

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host named VMHost01

from the VMM database and stores the host object in variable $VMHost.

The last command makes VMHost01 available as a host for virtual machines. S

etting the parameter AvailableForPlacement to TRUE enables the VMM placemen

t process to evaluate this host as a possible candidate on which to deploy

virtual machines.

2: Enable remote connections on a Virtual Server host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02"

PS C:\> Set-VMHost -VMHost $VMHost -RemoteConnectEnabled $TRUE -RemoteConne

ctPort 5900 -RemoteConnectTimeoutEnabled $TRUE -RemoteConnectTimeoutMinutes

35

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost02

and stores the host object in $VMHost.

The last command enables remote connections on VMHost02 and sets the follow

ing options:

\* Sets the port used for remote connections to 5900

\* Enables the remote connecton timeout feature

\* Specifies that remote connections time out (disconnect) at 35 minutes

Enabling remote connections on this Virtual Server host lets users access a

nd manage their virtual machines deployed on this host remotely.

3. Update the virtual machine paths for a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03"

PS C:\> Set-VMHost -VMHost $VMHost -VMPaths "Z:\MySpecialVMPath1|Z:MySpecia

lVMPath2"

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost03

and stores the host object in $VMHost.

The last command updates the VMPaths property for the object that represent

s VMHost03 in the VMM database by adding the path "Z:\MySpecialVMPath" to t

he list of virtual machine paths on that host.

4. Update the resource reserves for a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost04"

PS C:\> Set-VMHost -VMHost $VMHost -CPUPercentageReserve 40 -DiskSpaceReser

veMB 1024 -MaxDiskIOReservation 500 -MemoryReserve 1024 -NetworkPercentageR

eserve 40

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost04

and stores the host object in $VMHost.

The last command updates the specified properties (for CPU, disk, memory, a

nd network reserve settings) for VMHost04.

REMARKS

For more information, type: "get-help Set-VMHost -detailed".

For technical information, type: "get-help Set-VMHost -full".

### Update-VMHost

SYNOPSIS

Updates Virtual Server 2005 R2 software installed on a virtual machine host

to the latest version of Virtual Server supported by Virtual Machine Manag

er.

SYNTAX

Update-VMHost [-VMHost] [<String Host>] [-JobVariable <String>] [-PROTipID

<Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Updates Virtual Server 2005 R2 Service Pack 1 (SP1) software installed on a

virtual machine host to the latest version of Virtual Server supported by

Virtual Machine Manager.

You need to use this cmdlet only if you have added a server to Virtual Mach

ine Manager as a host that is running Virtual Server 2005 R2 SP1 and that h

as not been updated.

If the server is running Virtual Server 2005 R2, the cmdlet will block the

update and request that you manually upgrade to Virtual Server 2005 R2 SP1,

and then run Update-VMHost again.

CAUTION: When you run the Update-VMHost cmdlet to update the Virtual Server

software on a host, the cmdlet puts all virtual machines running on the ho

st into a stopped state and might reboot the host server.

For more information about hosts supported by VMM 2008, including Virtual S

erver hosts, type: Get-Help Add-VMHost -detailed

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Update a host running Virtual Server 2005 R2 SP1 to the latest version o

f Virtual Server supported by VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> Update-VMHost -VMHost $VMHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost.

The last command updates the Virtual Server 2005 R2 SP1 software installed

on VMHost01 to the latest version of Virtual Server supported by VMM.

2: Update all hosts running running Virtual Server 2005 R2 SP1 to the lates

t version of Virtual Server supported by VMM.

PS C:\> Get-VMHost -VMMServer VMMServer1.Contoso.com | where {$\_.VirtualSer

verVersionState -eq "UpgradeAvailable"} | Update-VMHost -RunAsynchronously

This command performs the following actions:

\* Gets all objects that represent hosts managed by VMMServer1.

\* Selects only those host objects whose version state property indicates

that an upgrade for Virtual Server is available.

\* Updates the Virtual Server 2005 R2 SP1 software installed on each host

to the latest version of Virtual Server supported by Virtual Machine

Manager. Specifying the -RunAsynchronously parameter returns control

to the command shell immediately (before the command completes).

REMARKS

For more information, type: "get-help Update-VMHost -detailed".

For technical information, type: "get-help Update-VMHost -full".

## VMHostCluster

### Add-VMHostCluster

SYNOPSIS

Adds an object to Virtual Machine Manager that represents a host cluster.

SYNTAX

Add-VMHostCluster [-Name] <String> -Credential <PSCredential> [-AddVMHostJo

bsListVariable <JobVariable[]>] [-ClusterReserve <Int32>] [-Description <St

ring>] [-JobVariable <String>] [-PROTipID <Guid>] [-Reassociate] [-RemoteCo

nnectEnabled <Boolean>] [-RemoteConnectPort <Int32>] [-RunAsynchronously] [

-VMHostGroup <HostGroup>] [-VMMServer [<String ServerConnection>]] [-VMPath

s <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Adds an existing Windows Server 2008 failover host cluster to the Virtual M

achine Manager database so that Virtual Machine Manager can start managing

the host cluster.

Before you can use the Add-VMHostCluster cmdlet to add a Windows Server 200

8 host cluster to Virtual Machine Manager, you must use the Failover Cluste

r Management tool to create and configure the host cluster.

You cannot use the Add-VMHostCluster cmdlet to add a VMware host cluster to

Virtual Machine Manager. Instead, you use the Add-VirtualizationManager cm

dlet to add a VMware VirtualCenter Server to your Virtual Machine Manager e

nvironment and import its data. After adding the VirtualCenter Server to Vi

rtual Machine Manager, Virtual Machine Manager can manage VMware ESX hosts

(including host clusters) and the virtual machines deployed on those hosts.

For more information, type:

Get-Help about\_VMM\_2008\_Failover\_Clusters

Get-Help Add-VirtualizationManager -detailed

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-AddVMHostJobsListVariable <JobVariable[]>

Returns an array of job variable objects for jobs that are created for

each node when hosts in a host cluster are added to VMM. VMM uses these

job variables to track the progress of each job individually.

-ClusterReserve <Int32>

Specifies the number of host failures that a host cluster can sustain b

efore VMM designates the cluster as over-committed. The default value i

s 1.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-Reassociate

Reassociates a host currently managed by one VMM server with another VM

M server.

-RemoteConnectEnabled <Boolean>

Enables (when set to TRUE) a connection on a host server that lets user

s connect to their virtual machines remotely.

TYPE OF HOST TYPE OF REMOTE CONNECTION

------------ -------------------------

Virtual Server Enables Virtual Machine Remote Control (VMRC)

Hyper-V Enables Virtual Machine Connection (VMConnect)

VMware ESX This parameter does not apply to VMs on ESX hosts

Note: In VMM 2007, this parameter was named VMRCEnabled.

-RemoteConnectPort <Int32>

Specifies a default value for the TCP port to use when a remote user co

nnects to a virtual machine. Typically, the default port for a Virtual

Server host is 5900 (for a VMRC connection), and the default port for a

Hyper-V host is 2179 (for a VMConnect connection). This parameter does

not apply to a VMware ESX host.

Note: In VMM 2007, the Virtual Server default remote connect port param

eter was named VMRCPort.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VMPaths <String>

Specifies a set of default paths (as strings separated by the pipeline

operator) on a host where virtual machine files can be stored.

Example format: -VMPaths "C:\Folder1|C:\Folder2|C:\Folder3"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a failover cluster to VMM.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostGroup = Get-VMHostGroup | where {$\_.Path -eq "All Hosts"}

PS C:\> Add-VMHostCluster -Name "VMHostCluster01.Contoso.com" -VMHostGroup

$VMHostGroup -RemoteConnectEnabled $TRUE -RemoteConnectPort 5900 -Credentia

l $Credential

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are a domain account with administrator r

ights on all the nodes of the failover cluster that you want to add.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands use this

server by default.

The third command gets the object that represents the host group "All Hosts

". This is the host group that will be the container for the nodes in this

host cluster, which are added to VMM in the last command.

The last command adds the failover cluster VMHostCluster01 in the Contoso d

omain to the VMM database, specifies "All Hosts" as the host group, enables

remote connections, and specifies that TCP port 5900 will be used for remo

te connections to each node of the cluster. As the last command is processe

d, variable $Credential provides your credentials to Add-VMHostCluster.

REMARKS

For more information, type: "get-help Add-VMHostCluster -detailed".

For technical information, type: "get-help Add-VMHostCluster -full".

### Get-VMHostCluster

SYNOPSIS

Gets a host cluster object, or an array of host cluster objects, from the V

irtual Machine Manager database.

SYNTAX

Get-VMHostCluster [[-Name] <String>] -VMHostGroup <HostGroup> [<CommonParam

eters>]

Get-VMHostCluster [[-Name] <String>] [-VMMServer [<String ServerConnection>

]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets a host cluster object, or an array of host cluster objects, from the V

irtual Machine Manager database.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all host clusters managed by this VMM server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHostCluster

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets from the VMM database all objects that represent al

l host clusters currently managed by VMMServer1 and displays information ab

out each host cluster object to the user.

2: Get all host clusters, and display the host cluster name and virtualizat

ion platform.

PS C:\> Get-VMHostCluster -VMMServer VMMServer1.Contoso.com | select -prope

rty Name, VirtualizationPlatform

Gets all objects that represent host clusters from VMMServer1 and passes ea

ch host cluster object to "select" (the alias for the Select-Object cmdlet)

, which displays the name of each host cluster and the virtualization platf

orm of the hosts that belong to that host cluster.

3: Get a host cluster by name.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHostCluster -Name "VMHostCluster03.Contoso.com"

The first command connects to VMMServer1.

The second command gets fthe host cluster object that represents VMHostClus

ter03.

4: Display the object type, methods, and properties for a host cluster mana

ged by VMM.

PS C:\> $VMHostCluster = Get-VMHostCluster -VMMServer VMMServer1.Contoso.co

m

PS C:\> $VMHostCluster | Get-Member

The first command gets the object that represents a host cluster from VMMSe

rver1 and stores the host cluster object in $VMHostCluster. This example as

sumes that only one host cluster is managed by Virtual Machine Manager.

The second command passes the host cluster object to the Get-Member cmdlet,

which displays the .NET type for a host cluster object:

TypeName: Microsoft.SystemCenter.VirtualMachineManager.HostCluster

The command also displays a list of methods and properties associated with

this object.

REMARKS

For more information, type: "get-help Get-VMHostCluster -detailed".

For technical information, type: "get-help Get-VMHostCluster -full".

### Move-VMHostCluster

SYNOPSIS

Moves a Windows Server 2008 host cluster object managed by Virtual Machine

Manager from one host group to another.

SYNTAX

Move-VMHostCluster [-VMHostCluster] <VMHostCluster> -ParentHostGroup [<Stri

ng HostGroup>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronous

ly] [<CommonParameters>]

DETAILED DESCRIPTION

Moves an object that represents a Windows Server 2008 host cluster managed

by Virtual Machine Manager from one host group to another.

You cannot use the Move-VMHostCluster cmdlet to move a VMware host cluster.

Instead, use VirtualCenter Server to move a VMware host cluster.

PARAMETERS

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

-ParentHostGroup [<String HostGroup>]

Specifies the parent host group that contains one or more hosts, host g

roups, or host clusters.

Note: In VMM 2007, this parameter was named ParentVMHostGroup.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Move a single host cluster to a new parent host group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostCluster = Get-VMHostCluster -Name "VMHostCluster01.Contoso.c

om"

PS C:\> $DestinationHostGroup = Get-VMHostGroup | where {$\_.Name -eq "MyNew

Group"}

PS C:\> Move-VMHostCluster -VMHostCluster $VMHostCluster -ParentHostGroup $

DestinationHostGroup

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents failover cluster VMHostC

luster01.Contoso.com from the VMM database and stores the host cluster obje

ct in variable $VMHostCluster.

The third command gets the object that represents the host group named MyNe

wGroup and stores the host group object in variable $DestinationHostGroup.

The last command moves host cluster VMHostCluster01.Contoso.com (represente

d by $VMHostCluster) from its current host group to a new parent host group

called MyNewGroup (represented by $DestinationHostGroup).

REMARKS

For more information, type: "get-help Move-VMHostCluster -detailed".

For technical information, type: "get-help Move-VMHostCluster -full".

### Refresh-VMHostCluster

SYNOPSIS

Refreshes host cluster properties in the Virtual Machine Manager Administra

tor Console.

SYNTAX

Refresh-VMHostCluster [-VMHostCluster] <VMHostCluster> [-JobVariable <Strin

g>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Refreshes host cluster properties so that the Virtual Machine Manager Admin

istrator Console displays updated information about the host cluster.

PARAMETERS

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Refresh information about a specific host cluster.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostCluster = Get-VMHostCluster -Name "VMHostCluster01.Contoso.C

om"

PS C:\> Refresh-VMHostCluster -VMHostCluster $VMHostCluster

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents failover cluster VMHostC

luster01.Contoso.com from the VMM database and stores the host cluster obje

ct in variable $VMHostCluster.

The last command refreshes the properties for VMHostCluster01 so that curre

nt information about this host cluster will appear in the Administrator Con

sole.

REMARKS

For more information, type: "get-help Refresh-VMHostCluster -detailed".

For technical information, type: "get-help Refresh-VMHostCluster -full".

### Remove-VMHostCluster

SYNOPSIS

Removes a host cluster object from Virtual Machine Manager.

SYNTAX

Remove-VMHostCluster [-VMHostCluster] <VMHostCluster> [-Confirm] [-Credenti

al <PSCredential>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchro

nously] [<CommonParameters>]

Remove-VMHostCluster [-VMHostCluster] <VMHostCluster> -Force [-Confirm] [-J

obVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParame

ters>]

DETAILED DESCRIPTION

Removes one or more host cluster objects from Virtual Machine Manager.

CAUTION:

Do not use this cmdlet to remove a VMware host cluster from Virtual Machine

Manager unless you also want to remove the cluster from VirtualCenter Serv

er.

The cmdlet removes the VMware host cluster from VirtualCenter Server at the

same time that it removes the cluster from Virtual Machine Manager.

PARAMETERS

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

-Force

Forces the removal of an object from the VMM database and removes any a

ssociation between this object and other objects.

-Confirm

Prompts for confirmation before running the command.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove a specific host cluster from VMM.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostCluster = Get-VMHostCluster -Name "VMHostCluster01.Contoso.c

om"

PS C:\> Remove-VMHostCluster -VMHostCluster $VMHostCluster -Credential $Cre

dential -Confirm

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in variable $Credential. The requir

ed credentials for this operation are a domain account with administrator r

ights on all of the nodes of the failover cluster that you want to remove.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands use this

server by default.

The third command gets the object that represents failover cluster VMHostCl

uster01.Contoso.com from the VMM database and stores the host cluster objec

t in variable $VMHostCluster.

The last command removes the VMHostCluster01 object from the VMM database a

nd stops managing that host cluster. It does not modify the host cluster se

ttings or its existing virtual machines in any way. As this command is proc

essed, $Credential provides your credentials to Remove-VMHostCluster. This

example assumes that there is only one host cluster object. For an example

that shows how to change the syntax to remove multiple cluster objects, see

the next example.

CAUTION:

Do not use this cmdlet to remove a VMware host cluster from Virtual Machine

Manager unless you also want to remove the cluster from VirtualCenter Serv

er. The cmdlet removes the VMware host cluster from VirtualCenter Server at

the same time that it removes the cluster from Virtual Machine Manager.

NOTE: You cannot use the Remove-VMHost cmdlet to remove individual hosts fr

om a host cluster. VMM does not allow partial management of host clusters -

you can manage all of the hosts in the host cluster or none of them. The R

emove-VMHostCluster cmdlet removes the host cluster and its nodes from VMM.

If your goal is to remove the host cluster entirely but continue to use VM

M to manage the hosts, use these steps:

1. Use Remove-VMHostCluster to remove the host cluster

from VMM (as shown in this example).

2. On the server on which the host cluster was originally

created, used the appropriate tool (such as the Windows

Server 2008 Failover Cluster Management console or the

VMware Virtual Infrastructure Client) to remove the host

cluster entirely.

3. In VMM, use Add-VMHost to add the hosts (which were nodes

in the host cluster) to VMM. Virtual machines on those

hosts will remain intact throughout this process.

2: Remove all host clusters from VMM.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHostCluster | Remove-VMHostCluster -Credential $Credential -C

onfirm

The first command uses Get-Credential to prompt you to supply a user name a

nd password and stores your credentials in $Credential. The required creden

tials for this operation are a domain account with administrator rights on

all of the nodes of the failover cluster that you want to remove.

The second command connects to VMMServer1.

The last command gets all host cluster objects and passes those host cluste

r objects to the Remove-VMHostCluster cmdlet, which removes each host clust

er object from Virtual Machine Manager and stops managing the corresponding

host cluster. The command does not modify the host cluster settings or its

existing virtual machines in any way. As this command is processed, $Crede

ntial provides your credentials to Remove-VMHostCluster.

CAUTION:

Do not use this cmdlet to remove a VMware host cluster from Virtual Machine

Manager unless you also want to remove the cluster from VirtualCenter Serv

er. The cmdlet removes the VMware host cluster from VirtualCenter Server at

the same time that it removes the cluster from Virtual Machine Manager.

REMARKS

For more information, type: "get-help Remove-VMHostCluster -detailed".

For technical information, type: "get-help Remove-VMHostCluster -full".

### Set-VMHostCluster

SYNOPSIS

Modifies the properties of a virtual machine host cluster managed by Virtua

l Machine Manager.

SYNTAX

Set-VMHostCluster [-VMHostCluster] <VMHostCluster> [-ClusterReserve <Int32>

] [-Description <String>] [-InheritPROSettings <Boolean>] [-JobGroup <Guid>

] [-JobVariable <String>] [-PROAutomationLevel <Int32>] [-PROMonitoringLeve

l <Int32>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Modifies the properties of a host cluster managed by Virtual Machine Manage

r. Properties that you can modify include changing the cluster reserve sett

ing and enabling Performance and Resource Optimization (PRO) settings.

CLUSTER RESERVE

--------------------

The cluster reserve setting specifies the number of host failures that a ho

st cluster can sustain before Virtual Machine Manager considers the cluster

to be over-committed. An over-committed host cluster is one that cannot wi

thstand the specified number of host failures and still keep all of the vir

tual machines in the cluster running.

Virtual Machine Manager uses the following processes to determine over-comm

itment:

\* Host Placement. The placement process calculates whether adding a new

virtual machine to the host cluster will over-commit the host cluster

and, if so, placement stops recommending the deployment of additional

virtual machines on hosts in that cluster.

\* Cluster Refresher. The host cluster refresher calculates, at periodic

intervals, whether a host cluster is over-committed or not based on any

of the following events:

- A change in the value specified for -ClusterReserve.

- The failure or removal of nodes from the host cluster.

- The addition of nodes to the host cluster.

- The discovery of new virtual machines on nodes in the host cluster.

The following examples illustrate how over-commitment works:

\* Example of over-commitment when all nodes are functioning:

If you specify a cluster reserve of 2 for an 8-node host cluster,

and all 8 nodes are functioning, the host cluster is over-committed if

any combination of 6 (8-2) nodes lacks the capacity to accommodate

existing virtual machines.

\* Example of over-commitment when some nodes are not functioning:

If you specify a cluster reserve of 2 for an 8-node host cluster,

but only 5 nodes are functioning, the host cluster is over-committed if

any combination of 3 (5-2) nodes lacks the capacity to accommodate

existing virtual machines.

PRO MONITORING AND RECOMMENDED ACTIONS

--------------------------------------

PRO can, if enabled, monitor workload- and application-aware resources for

hosts in a host cluster (and for virtual machines deployed on those hosts)

and can provide recommended actions (such as move a virtual machine to a ne

w host, or add a CPU to the virtual machine) that you can implement manuall

y or automatically.

You can use the Set-VMHostCluster cmdlet with the PROAutomationLevel and PR

OMonitoringLevel parameters to enable PRO monitoring and tip implementation

for a host cluster only if Virtual Machine Manager is configured to use Sy

stem Center Operations Manager 2007 to manage those hosts and if the System

Center Virtual Machine Manager 2008 Management Pack has been imported into

that Operations Manager Server.

You can use the InheritPROSettings parameter with this cmdlet to specify wh

ether child host clusters will inherit the same PRO monitoring and PRO auto

mation settings configured for the parent host cluster.

PARAMETERS

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

-ClusterReserve <Int32>

Specifies the number of host failures that a host cluster can sustain b

efore VMM designates the cluster as over-committed. The default value i

s 1.

-Description <String>

Specifies a description for the specified object.

-InheritPROSettings <Boolean>

Specifies (when set to TRUE) that the host servers in a child host grou

p or child host cluster will have the same values for the PROMonitoring

Level and PROAutomationLevel parameters as the values specified for tho

se parameters for the parent container.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROAutomationLevel <Int32>

Specifies the severity level of tips that PRO will implement.

Valid values:

0 (Off) Respond to a PRO tip manually

1 (Critical) Implement critical PRO tips automatically

2 (CriticalAndWarning) Implement critical or warning PRO tips automatic

ally

Note: Applies only to Hyper-V or VMware ESX hosts or to VMs deployed on

those hosts.

-PROMonitoringLevel <Int32>

Specifies the severity level of tips that PRO will monitor.

Valid values:

0 (Off) Do not monitor PRO tips

1 (Critical) Monitor critical PRO tips

2 (CriticalAndWarning) Monitor critical and warning PRO tips

Note: Applies only to Hyper-V or VMware ESX hosts or to VMs deployed on

those hosts.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the setting for the cluster reserve for a host cluster.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostCluster = Get-VMHostCluster -Name "VMHostCluster01.Contoso.c

om"

PS C:\> Set-VMHostCluster -VMHostCluster $VMHostCluster -ClusterReserve 2

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents failover cluster VMHostC

luster01.Contoso.com from the VMM database and stores the host cluster obje

ct in variable $VMHostCluster.

The last command changes value for the cluster reserve for host cluster VMH

ostCluster01 to 2 by setting the parameter ClusterReserve to 2.

2: Enable PRO for a specific host cluster.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostCluster = Get-VMHostCluster -Name "VMHostCluster02.Contoso.c

om"

PS C:\> Set-VMHostCluster -VMHostCluster $VMHostCluster -PROMonitoringLevel

2 -PROAutomationLevel 1

The first command connects to VMMServer1.

The second command gets the object that represents failover cluster VMHostC

luster02.Contoso.com and stores the host cluster object in variable $VMHost

Cluster.

The last command enables PRO Monitoring on the cluster for both warning ale

rts and critical alerts, and it turns on automatic resolution for critical

alerts.

REMARKS

For more information, type: "get-help Set-VMHostCluster -detailed".

For technical information, type: "get-help Set-VMHostCluster -full".

## VMHostDisk

### Get-VMHostDisk

SYNOPSIS

Gets a hard disk drive object for the specified host from the Virtual Machi

ne Manager database.

SYNTAX

Get-VMHostDisk [[-Name] <String>] -VMHost [<String Host>] [<CommonParameter

s>]

DETAILED DESCRIPTION

Gets one or more hard disk drive objects for the specified host from the Vi

rtual Machine Manager database. You can use this cmdlet with the New-Virtua

lDiskDrive cmdlet to attach a pass-through disk on a virtual machine to a p

hysical hard disk on the host on which that virtual machine is deployed.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all hard disk drives on the specified host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> Get-VMHostDisk -VMHost $VMHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object for VMHost01 from the VMM database and s

tores the host object in variable $VMHost.

The last command gets all hard disk drive objects from VMHost01 and display

s information about those objects to the user.

2. Get a specific hard disk drive on the host by name.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostDisk = Get-VMHost -ComputerName "VMHost02.Contoso.com" | Get

-VMHostDisk –Name "\\.\PhysicalDrive0"

PS C:\> $VMHostDisk

The first command connects to VMMServer1.

The second command gets the object from VMHost02 that represents the host h

ard disk drive with a name that matches the pattern "PhysicalDrive0" and st

ores the hard disk drive object in $VMHostDisk.

The last command displays the contents of $VMHostDisk to the user.

REMARKS

For more information, type: "get-help Get-VMHostDisk -detailed".

For technical information, type: "get-help Get-VMHostDisk -full".

## VMHostGroup

### Get-VMHostGroup

SYNOPSIS

Gets a host group object from the Virtual Machine Manager database.

SYNTAX

Get-VMHostGroup [[-Name] <String>] [-VMMServer [<String ServerConnection>]]

[<CommonParameters>]

Get-VMHostGroup [[-Name] <String>] -ID <Guid> [<CommonParameters>]

Get-VMHostGroup [[-Name] <String>] -ParentHostGroup [<String HostGroup>] [<

CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent host groups from the Virtual Machin

e Manager database. Virtual Machine Manager provides a default parent host

group, called All Hosts. Host groups within All Hosts are user-created cont

ainers that can contain any of the following:

\* A host or set of hosts

\* A host group or set of host groups, and hosts within those host groups

\* A host cluster, and hosts (called nodes) within that host cluster

For more information about host groups, type:

Get-Help New-VMHostGroup -detailed

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-ID <Guid>

Specifies the numerical identifier (as a globally unique identifier, or

GUID) for a specific object.

-ParentHostGroup [<String HostGroup>]

Specifies the parent host group that contains one or more hosts, host g

roups, or host clusters.

Note: In VMM 2007, this parameter was named ParentVMHostGroup.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all host groups at the specified path.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHostGroup | where { $\_.Path -eq "All Hosts\Production" }

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets the objects that represent one or more host groups

located at host path "All Hosts\Production" and displays information about

these host groups to the user.

2: Display the name and path properties for all host groups.

PS C:\> Get-VMHostGroup -VMMServer VMMServer1 | select -Property Name,Path

Gets all host group objects from VMMServer1, selects the name and host grou

p path properties, and displays those properties to the user.

REMARKS

For more information, type: "get-help Get-VMHostGroup -detailed".

For technical information, type: "get-help Get-VMHostGroup -full".

### Move-VMHostGroup

SYNOPSIS

Moves a host group object managed by Virtual Machine Manager from the curre

nt location to a new location under a different host group parent.

SYNTAX

Move-VMHostGroup [-VMHostGroup] <HostGroup> -ParentHostGroup [<String HostG

roup>] [-JobGroup <Guid>] [-JobVariable <String>] [-PROTipID <Guid>] [-RunA

synchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Moves one or more objects that represent host groups (which contain virtual

machine hosts managed by Virtual Machine Manager) from the current locatio

n to a new location under a different host group parent. You can locate hos

t groups under the default root host group or under any other host group cr

eated by an administrator.

All hosts within a moved host group acquire a new host path relative to the

root host group. Changing the structure of host groups might change which

Self Service User or Delegated Administrator user roles have access to the

hosts contained within the affected host groups, or to the virtual machines

deployed on those hosts.

PARAMETERS

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-ParentHostGroup [<String HostGroup>]

Specifies the parent host group that contains one or more hosts, host g

roups, or host clusters.

Note: In VMM 2007, this parameter was named ParentVMHostGroup.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Move one host group to a new parent host group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $OldHostGroup = Get-VMHostGroup | where { $\_.Name -eq "MyOldGroup"

}

PS C:\> $HostGroup = Get-VMHostGroup | where { $\_.Name -eq "MyNewGroup" }

PS C:\> Move-VMHostGroup -VMHostGroup $OldHostGroup -ParentHostGroup $HostG

roup

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host group named MyO

ldGroup from the VMM database and stores the host group object in variable

$OldHostGroup.

The third command gets the object that represents the host group named MyNe

wGroup and stores this host group object in variable $HostGroup.

The last command moves the host group MyOldGroup (represented by $OldHostGr

oup) to a location under its new parent host group, MyNewGroup (represented

by $HostGroup).

2: Move all host groups to a new parent host group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $AllGroups = Get-VMHostGroup

PS C:\> $HostGroup = New-VMHostGroup -Name "Lab04"

PS C:\> $AllGroups | Move-VMHostGroup -ParentHostGroup $HostGroup

The first command connects to VMMServer1.

The second command gets all objects that represent host groups and stores t

hese host group objects in $AllGroups. $AllGroups contains the object for e

very host group -- including the default parent host group (AllHosts).

The third command creates a host group object called Lab04 and stores the n

ew host group object in $HostGroup.

The last command passes each host group object in $AllGroups to the Move-VM

HostGroup cmdlet, which moves each host group object to a new location unde

r $HostGroup - except for AllHosts because AllHosts is the default parent h

ost group and cannot be moved.

REMARKS

For more information, type: "get-help Move-VMHostGroup -detailed".

For technical information, type: "get-help Move-VMHostGroup -full".

### New-VMHostGroup

SYNOPSIS

Creates a Virtual Machine Manager host group object that can contain virtua

l machine host servers, other host groups, or host clusters.

SYNTAX

New-VMHostGroup [-Name] <String> [-Description <String>] [-JobVariable <Str

ing>] [-ParentHostGroup [<String HostGroup>]] [-PROTipID <Guid>] [-RunAsync

hronously] [-VMMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a Virtual Machine Manager host group object that can contain host s

ervers on which one or more virtual machines are deployed, and that might c

ontain other host groups or host clusters.

Virtual Machine Manager provides a default parent host group called All Hos

ts, to which you can add child host groups. A new host group is empty until

you move hosts into it and/or create one or more child host groups under t

hat host group. Host groups are organized into a hierarchical and customiza

ble tree structure. In the host group tree, the parent of a new host group

is either the default root host group (All Hosts) or a user-created host gr

oup.

A host group can be a parent container for any of the following:

\* A host or set of hosts

\* A host group or set of host groups, and hosts within those host groups

\* A host cluster, and hosts (called nodes) within that host cluster

Hosts contained in a specific host group have a host path property that sho

ws the location of that host in the host group hierarchy, as illustrated in

the following table.

Name Path

---- ----

All Hosts All Hosts

ChildHostGroup1 All Hosts\ChildHostGroup1

ChildHostGroup2 All Hosts\ChildHostGroup2

New Datacenter All Hosts\New Datacenter

nested1 All Hosts\New Datacenter\nested1

nested2 All Hosts\New Datacenter\nested\nested2

For an example that shows you how to display the host group path property,

type:

Get-Help Get-VMHostGroup -example

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-ParentHostGroup [<String HostGroup>]

Specifies the parent host group that contains one or more hosts, host g

roups, or host clusters.

Note: In VMM 2007, this parameter was named ParentVMHostGroup.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a host group for virtual machines under the root host group.

PS C:\> New-VMHostGroup -VMMServer "VMMServer1.Contoso.com" -Name "HostGrou

p01"

Creates a host group named HostGroup01 and stores its object in the VMM dat

abase on VMMServer1 in the Contoso.com domain.

By default, VMM places this host group under the root host group, which is

called All Hosts.

2: Create a host group for virtual machines under the specified parent host

group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $ParentGroup = Get-VMHostGroup | where { $\_.Name -eq "HostGroup02"

}

PS C:\> New-VMHostGroup -Name "ChildGroup02" -ParentHostGroup $ParentGroup

The first command connects to VMMServer1.

The second command gets the object that represents the host group named Hos

tGroup02 and stores the host group object in $ParentGroup.

The third command creates a host group called ChildGroup02 and places it un

der the parent host group, HostGroup (represented by $ParentGroup).

REMARKS

For more information, type: "get-help New-VMHostGroup -detailed".

For technical information, type: "get-help New-VMHostGroup -full".

### Remove-VMHostGroup

SYNOPSIS

Removes a host group object from Virtual Machine Manager.

SYNTAX

Remove-VMHostGroup [-VMHostGroup] <HostGroup> [-JobVariable <String>] [-PRO

TipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent host groups from the Virtual Mac

hine Manager database. This cmdlet also deletes a host group's child host g

roups (if any exist) if the host group and its child host groups do not con

tain any virtual machine hosts.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the specified host group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $HostGroup = Get-VMHostGroup | where { $\_.Name -eq "MyNewGroup" }

PS C:\> Remove-VMHostGroup -VMHostGroup $HostGroup

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host group named MyN

ewGroup from the VMM database and stores the host group object in variable

$HostGroup.

The third command removes the object that represents the host group stored

in $HostGroup from the VMM database.

REMARKS

For more information, type: "get-help Remove-VMHostGroup -detailed".

For technical information, type: "get-help Remove-VMHostGroup -full".

### Set-VMHostGroup

SYNOPSIS

Changes properties of a host group in Virtual Machine Manager.

SYNTAX

Set-VMHostGroup [-VMHostGroup] <HostGroup> [-ApplyReservesToChildHostGroups

<Boolean>] [-CPUPercentageReserve <Int32>] [-Description <String>] [-DiskS

paceReserveMB <Int32>] [-InheritPROSettings <Boolean>] [-JobGroup <Guid>] [

-JobVariable <String>] [-MaxDiskIOReservation <Int32>] [-MemoryReserveMB <I

nt32>] [-Name <String>] [-NetworkPercentageReserve <Int32>] [-PROAutomation

Level <Int32>] [-PROMonitoringLevel <Int32>] [-PROTipID <Guid>] [-RunAsynch

ronously] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of a host group that contains hosts managed

by Virtual Machine Manager. Properties that you can change include settings

for name, description, multiple settings that specify host reserves, and s

ettings that enable computers in the host group to be monitored by Performa

nce and Resource Optimization (PRO) and to implement actions recommended by

PRO.

HOST RESERVES

-------------

Host reserve settings that you can specify for hosts in a host group by usi

ng the Set-VMHostGroup cmdlet include the percentage of CPU, memory, disk s

pace, and network throughput that are set aside for the use of the host ope

rating system on the physical host computer. A virtual machine cannot be pl

aced on a host if the virtual machine’s resource requirements cannot be met

without using host reserves.

You can also specify whether child host groups will inherit the same host r

eserve settings configured for the parent host group.

PRO MONITORING AND RECOMMENDED ACTIONS

--------------------------------------

PRO can, if enabled, monitor workload- and application-aware resources for

hosts in a host group (and virtual machines deployed on those hosts) and ca

n provide recommended actions (such as move a virtual machine to a new host

, or add a CPU to the virtual machine) that you can implement manually or a

utomatically.

You can use the Set-VMHostGroup cmdlet with the PROAutomationLevel and PROM

onitoringLevel parameters to enable PRO monitoring and tip implementation f

or a host group only if VMM is configured to use System Center Operations M

anager 2007 to manage those hosts and if the System Center Virtual Machine

Manager 2008 Management Pack has been imported into that Operations Manager

Server.

You can use the InheritPROSettings parameter with this cmdlet to specify wh

ether child host groups will inherit the same PRO monitoring and PRO automa

tion settings configured for the parent host group.

PARAMETERS

-VMHostGroup <HostGroup>

Specifies a virtual machine host group object.

-ApplyReservesToChildHostGroups <Boolean>

Specifies (when set to TRUE) that child host groups will inherit the se

ttings for host reserves specified for the parent host group, and overw

rites any previous values.

-CPUPercentageReserve <Int32>

Specifies the percentage of CPU to set aside for the use of the host op

erating system on the physical host computer. If you do not use this pa

rameter to specify the reserve, the default setting for the host group

is used: 10 percent. The Virtual Machine Manager placement process will

not recommend that a virtual machine be placed on a host unless its re

source requirements can be met without using host reserves.

-Description <String>

Specifies a description for the specified object.

-DiskSpaceReserveMB <Int32>

Specifies, in megabytes (MB), the amount of disk space to set aside for

the use of the host operating system on the physical host computer. If

you do not use this parameter to specify the reserve, the default sett

ing for the host group is used: 100 MB. The Virtual Machine Manager pla

cement process will not recommend that a virtual machine be placed on a

host unless its resource requirements can be met without using host re

serves.

-InheritPROSettings <Boolean>

Specifies (when set to TRUE) that the host servers in a child host grou

p or child host cluster will have the same values for the PROMonitoring

Level and PROAutomationLevel parameters as the values specified for tho

se parameters for the parent container.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-MaxDiskIOReservation <Int32>

Specifies the maximum number of disk I/O operations per second (IOPS) f

or the operating system on the physical host computer. If you do not us

e this parameter to specify the reserve, the default setting for the ho

st group is used: 10000 IOPS. The Virtual Machine Manager placement pro

cess will not recommend that a virtual machine be placed on a host unle

ss its resource requirements can be met without using host reserves.

-MemoryReserveMB <Int32>

Specifies, in megabytes (MB), the amount of memory to set aside for the

use of the host operating system on the physical host computer. If you

do not use this parameter to specify the reserve, the default setting

for the host group is used: 256 MB. The Virtual Machine Manager placeme

nt process will not recommend that a virtual machine be placed on a hos

t unless its resource requirements can be met without using host reserv

es.

-Name <String>

Specifies the name of a VMM object.

-NetworkPercentageReserve <Int32>

Specifies the percentage of network capacity to set aside for the use o

f the host operating system on the physical host computer. If you do no

t use this parameter to specify the reserve, the default setting for th

e host group is used: 10 percent. The Virtual Machine Manager placement

process will not recommend that a virtual machine be placed on a host

unless its resource requirements can be met without using host reserves

.

-PROAutomationLevel <Int32>

Specifies the severity level of tips that PRO will implement.

Valid values:

0 (Off) Respond to a PRO tip manually

1 (Critical) Implement critical PRO tips automatically

2 (CriticalAndWarning) Implement critical or warning PRO tips automatic

ally

Note: Applies only to Hyper-V or VMware ESX hosts or to VMs deployed on

those hosts.

-PROMonitoringLevel <Int32>

Specifies the severity level of tips that PRO will monitor.

Valid values:

0 (Off) Do not monitor PRO tips

1 (Critical) Monitor critical PRO tips

2 (CriticalAndWarning) Monitor critical and warning PRO tips

Note: Applies only to Hyper-V or VMware ESX hosts or to VMs deployed on

those hosts.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change host reserves on an existing host group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostGroup = Get-VMHostGroup | where { $\_.Name -eq "HostGroup01"

}

PS C:\> Set-VMHostGroup -VMHostGroup $HostGroup -CPUPercentageReserve 10 -D

iskSpaceReserveMB 200 -MemoryReserveMB 512

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host group named Hos

tGroup01 and stores the host group object in variable $HostGroup.

The last command changes the value of the following host reserves for all h

osts in $HostGroup:

\* Sets the percentage of CPU reserved for each host to 10 percent.

\* Sets the amount of disk space reserved for each host to 200 MB.

\* Sets the amount of memory reserved for each host to 512 MB.

A virtual machine cannot be placed on a host in this host group if the virt

ual machine’s resource requirements cannot be met without using host reserv

es.

REMARKS

For more information, type: "get-help Set-VMHostGroup -detailed".

For technical information, type: "get-help Set-VMHostGroup -full".

## VMHostNetworkAdapter

### Add-VMHostNetworkAdapter

SYNOPSIS

Adds a physical network adapter on a host server managed by Virtual Machine

Manager to a virtual network.

SYNTAX

Add-VMHostNetworkAdapter [-VMHostNetworkAdapter] <HostNetworkAdapter[]> -Vi

rtualNetwork <VirtualNetwork> [-JobGroup <Guid>] [-JobVariable <String>] [-

PROTipID <Guid>] [-RunAsynchronously] [-VLANEnabled] [-VLANID <Int32>] [-VL

ANMode <String>] [-VLANTrunkID <Int32[]>] [<CommonParameters>]

DETAILED DESCRIPTION

Adds a physical network adapter (also called a network interface card, or N

IC) on a host managed by Virtual Machine Manager to a virtual network. Each

virtual machine on that host can also connect (through a virtual network a

dapter) to that virtual network.

A virtual network configured on a host can connect to multiple virtual netw

ork adapters on virtual machines deployed on that host. However, the number

of physical host network adapters that connect to a virtual network on the

host varies by the type of host.

TYPE OF HOST NUMBER OF HOST NIC CONNECTIONS TO VIRTUAL NETWORKS

-------------- --------------------------------------------------

Virtual Server 1 physical adapter can connect to multiple virtual networks

Hyper-V 1 physical adapter can connect to 1 virtual network

VMware ESX Multiple physical adapters can connect to 1 virtual network

VMM 2008 extends virtual networking support to include support for configur

ing one or more virtual area networks (VLANs) on a host. You can use the Ad

d-VMHostNetworkAdapter cmdlet (or the Set-VMHostNetworkAdapter cmdlet) to c

onfigure a single VLAN or multiple VLANs on a host. To configure correspond

ing VLAN settings on a virtual machine, use the New-VirtualNetworkAdapter o

r the Set-VirtualNetworkAdapter cmdlet.

For an illustration of each type of VLAN, see the examples for this cmdlet.

For more information about VLANs, type:

Get-Help about\_VMM\_2008\_Virtual\_Networking

PARAMETERS

-VMHostNetworkAdapter <HostNetworkAdapter[]>

Specifies an array of one or more physical network adapter objects on a

host to which virtual machines deployed on that host can connect.

Example format: -VMHostNetworkAdapters $VMHostNICs

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VLANEnabled

Enables a virtual LAN (a VLAN) for use by virtual machines on a Hyper-V

or VMware ESX host.

Example format for a single VLAN:

-VLANEnabled -VLANMode "Access" -VLANID 35

Example format for multiple VLANs:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

-VLANID <Int32>

Assigns a numerical identifier in the range 1-4094 to a virtual network

adapter on a virtual machine or to a physical network adapter on a Hyp

er-V or VMware ESX host.

Configure a VLANID on a Hyper-V or VMware ESX host:

\* On an externally bound physical network adapter.

Configure a VLANID on a virtual network adapter of a virtual machine:

\* Bound to a physical network adapter on the host, or

\* Bound to an internal virtual network on the host.

Example format:

-VLANEnabled

-VLANMode "Access" -VLANID 35

-VLANMode <String>

Specifies whether a virtual LAN (a VLAN) on a Hyper-V or VMware ESX hos

t supports traffic across a single VLAN ("Access" mode) or across multi

ple VLANs ("Trunk" mode).

Valid values: "Trunk" or "Access"

-VLANTrunkID <Int32[]>

Assigns a list of numerical identifiers in the range 1-4094 to a physic

al network adapter on a Hyper-V host.

Example format:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add a physical host network adapter to a virtual network.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Externa

lNetwork01"

PS C:\> $VMHostNetworkAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost -N

ame "HostAdapter01"

PS C:\> Add-VMHostNetworkAdapter -VirtualNetwork $VirtualNetwork -VMHostNet

workAdapter $VMHostNetworkAdapter

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the host named VMHost01

from the VMM database and stores the host object in variable $VMHost.

The third command gets the object that represents a virtual network on VMHo

st01 named ExternalNetwork01 and stores the virtual network object in varia

ble $VirtualNetwork.

The fourth command gets the object that represents the physical network ada

pter on VMHost01 named HostAdapter01 and stores the adapter object in varia

ble $VMHostNetworkAdapter.

The last command adds HostAdapter01 to the virtual network named ExternalNe

twork01.

NOTE: You can add only one physical host adapter per virtual network, so th

e last command will fail if an adapter is already associated with the speci

fied virtual network. If that is the case, you must first remove the existi

ng host adapter from the virtual network and then retry the command.

2: Add a physical host network adapter to a VLAN that uses "Trunk" mode.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Externa

lNetwork02"

PS C:\> $VMHostNetworkAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost -N

ame "HostAdapter02"

PS C:\> Add-VMHostNetworkAdapter -VirtualNetwork $VirtualNetwork -VMHostNet

workAdapter $VMHostNetworkAdapter –VLANEnabled –VLANMode "Trunk" –VLANTrunk

ID 1,2,100,200,1124

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost02

and stores the host object in $VMHost.

The third command gets the object that represents a virtual network on VMHo

st02 called ExternalNetwork02 and stores the virtual network object in $Vir

tualNetwork.

The fourth command gets the object that represents the physical network ada

pter on VMHost02 named HostAdapter02 and stores the adapter object in $VMHo

stNetworkAdapter.

The last command adds HostAdapter02 to the ExternalNetwork02 virtual networ

k and enables access from ExternalNetwork02 to an external networking devic

e using 802.1Q tagged VLANs 1, 2, 100, 200, and 1124.

NOTE: You can add only one host adapter per virtual network, so the last co

mmand will fail if an adapter is already associated with the specified virt

ual network.

3: Add a physical host network adapter to a VLAN that uses "Access" mode.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Externa

lNetwork03"

PS C:\> $VMHostNetworkAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost -N

ame "HostAdapter03"

PS C:\> Add-VMHostNetworkAdapter -VirtualNetwork $VirtualNetwork -VMHostNet

workAdapter $VMHostNetworkAdapter –VLANEnabled –VLANMode "Access" –VLANID 2

2

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost03

and stores the host object in $VMHost.

The third command gets the object that represents a virtual network on VMHo

st03 called ExternalNetwork03 and stores the virtual network object in $Vir

tualNetwork.

The fourth command gets the object that represents the physical network ada

pter on VMHost03 named HostAdapter03 and stores the adapter object in $VMHo

stNetworkAdapter.

The last command adds HostAdapter03 to the ExternalNetwork03 virtual networ

k and restricts access to ExternalNetwork03 to VLANID 22.

NOTE: You can add only one host adapter per virtual network, so the last co

mmand will fail if an adapter is already associated with the specified virt

ual network.

CAUTION: This example assumes that that your host is already connected to a

VLAN or, if not, that your host has two network adapters. If your host has

a single network adapter (as might be the case if you are experimenting wi

th VMM cmdlets in a lab setting), assigning the adapter to a VLAN that is u

navailable to the VMM server will prevent VMM from managing the host. You c

an perform the steps in this example on a host that has only one network ad

apter if you first install the Microsoft Loopback Adapter on your server.

REMARKS

For more information, type: "get-help Add-VMHostNetworkAdapter -detailed".

For technical information, type: "get-help Add-VMHostNetworkAdapter -full".

### Get-VMHostNetworkAdapter

SYNOPSIS

Gets physical network adapter objects on a host managed by Virtual Machine

Manager.

SYNTAX

Get-VMHostNetworkAdapter [[-Name] <String>] -VMHost [<String Host>] [<Commo

nParameters>]

Get-VMHostNetworkAdapter [[-Name] <String>] [-VMMServer [<String ServerConn

ection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent physical network adapters on a host

managed by Virtual Machine Manager.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all physical network adapters on the specified host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> $HostAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost

PS C:\> $HostAdapter

PS C:\> $HostAdapter | select -property Name, ConnectionState

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost.

The third command gets from VMHost01 all objects that represent physical ne

twork adapters on this host and stores the adapter objects in variable $Hos

tAdapter.

The fourth command displays the properties of all of the adapters in $HostA

dapter.

The last command displays only the name and connection state for each adapter.

2: Get all physical network adapters in the VMM database.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMHostNetworkAdapter | Format-List Name, MacAddress, VMHost, Ma

xBandwidth

The first command connects to VMMServer1.

The second command gets all objects that represent physical network adapter

s on all hosts managed by VMMServer1 and displays each adapter's name, its

MAC address, its host name, and its maximum bandwidth.

3: Get a physical network adapter by name from a specific host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost03"

PS C:\> $HostAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost -Name "Host

Adapter03"

PS C:\> $HostAdapter | Format-List -property Name,NetworkLocation,VLANEnabl

ed,VLANMode

The first command connects to VMMServer1.

The second command gets the object that represents VMHost03 and stores the

host object in $VMHost.

The second command gets from VMHost03 the object that represents the networ

k adapter named HostAdapter03 and stores the adapter object in $HostAdapter

.

The last command passes the adapter object stored in $HostAdapter to the Fo

rmat-List cmdlet, which displays the name, network location, whether or not

a virtual LAN is enabled, and the current value for the VLAN mode (either

Trunk or Access).

4: Get each host network adapter that includes "Broadcom" in its name.

PS C:\> Get-VMHostNetworkAdapter -VMMServer VMMServer1.Contoso.com | where

{ $\_.Name -match "Broadcom" } | format-list -property Name,IPAddresses

Gets from VMMServer1 objects for each host network adapter that includes th

e string "Broadcom" in its name and displays the name and IP addresses for

each adapter.

REMARKS

For more information, type: "get-help Get-VMHostNetworkAdapter -detailed".

For technical information, type: "get-help Get-VMHostNetworkAdapter -full".

### Remove-VMHostNetworkAdapter

SYNOPSIS

Removes a physical host network adapter object from a virtual network that

is configured on a host managed by Virtual Machine Manager.

SYNTAX

Remove-VMHostNetworkAdapter [-VirtualNetwork] <VirtualNetwork> -Confirm -VM

HostNetworkAdapter <HostNetworkAdapter[]> [-JobGroup <Guid>] [-JobVariable

<String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Removes one or more objects that represent physical host network adapters f

rom a virtual network that is configured on a host managed by Virtual Machi

ne Manager.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VirtualNetwork <VirtualNetwork>

Specifies a virtual network object.

-Confirm

Prompts for confirmation before running the command.

-VMHostNetworkAdapter <HostNetworkAdapter[]>

Specifies an array of one or more physical network adapter objects on a

host to which virtual machines deployed on that host can connect.

Example format: -VMHostNetworkAdapters $VMHostNICs

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the physical host network adapter from a specific virtual network.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> $VirtualNetwork = Get-VirtualNetwork -VMHost $VMHost -Name "Externa

lNetwork01"

PS C:\> Remove-VMHostNetworkAdapter -VirtualNetwork $VirtualNetwork -Confir

m

The first command connects to VMMServer1 in the Contoso.com domain and retr

ieves the server object from the VMM database. The following commands use t

his server by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost.

The third command gets from VMHost01 the object that represents the virtual

network named ExternalNetwork01 and stores the virtual network object in v

ariable $VirtualNetwork.

The last command removes the host network adapter from ExternalNetwork01. T

he Confirm parameter prompts you to confirm whether you want to delete the

adapter from VMM.

REMARKS

For more information, type: "get-help Remove-VMHostNetworkAdapter -detailed

".

For technical information, type: "get-help Remove-VMHostNetworkAdapter -ful

l".

### Set-VMHostNetworkAdapter

SYNOPSIS

Changes network-related properties of the specified physical network adapte

r on a host managed by Virtual Machine Manager.

SYNTAX

Set-VMHostNetworkAdapter [-VMHostNetworkAdapter] <HostNetworkAdapter[]> [-D

escription <String>] [-JobGroup <Guid>] [-JobVariable <String>] [-NetworkLo

cation <String>] [-OverrideNetworkLocation <Boolean>] [-PROTipID <Guid>] [-

RunAsynchronously] [-VLANEnabled] [-VLANID <Int32>] [-VLANMode <String>] [-

VLANTrunkID <Int32[]>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes network-related properties of the specified physical network adapte

r on a host managed by Virtual Machine Manager.

Properties that you can change with this cmdlet include:

\* NETWORK LOCATION: You can use the OverrideNetworkLocation parameter

with the NetworkLocation parameter to change the default network

location of the specified physical network adapter. See example 1.

\* VLAN SETTINGS: You can use the VLAN parameters to create or modify a

single VLAN or multiple VLANs. For an illustration of how to specify

VLAN settings, see examples 2 and 3.

For more information about VLANs and additional examples that illustrate VL

AN settings, type:

Get-Help Add-VMHostNetworkAdapter -detailed

PARAMETERS

-VMHostNetworkAdapter <HostNetworkAdapter[]>

Specifies an array of one or more physical network adapter objects on a

host to which virtual machines deployed on that host can connect.

Example format: -VMHostNetworkAdapters $VMHostNICs

-Description <String>

Specifies a description for the specified object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-NetworkLocation <String>

Specifies the network location for a physical network adapter or for a

virtual network adapter, or changes the default network location of a h

ost's physical network adapter.

Example formats:

-NetworkLocation $NetLoc ($NetLoc might contain "Corp.Contoso.com")

-OverrideNetworkLocation $TRUE –NetworkLocation "HostNICNewLocation.Co

ntoso.com"

-OverrideNetworkLocation <Boolean>

Changes the discovered network location for a host's physical network a

dapter to a new network location.

Example format:

-OverrideNetworkLocation $TRUE –NetworkLocation "HostNICNewLocation.Con

toso.com"

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VLANEnabled

Enables a virtual LAN (a VLAN) for use by virtual machines on a Hyper-V

or VMware ESX host.

Example format for a single VLAN:

-VLANEnabled -VLANMode "Access" -VLANID 35

Example format for multiple VLANs:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

-VLANID <Int32>

Assigns a numerical identifier in the range 1-4094 to a virtual network

adapter on a virtual machine or to a physical network adapter on a Hyp

er-V or VMware ESX host.

Configure a VLANID on a Hyper-V or VMware ESX host:

\* On an externally bound physical network adapter.

Configure a VLANID on a virtual network adapter of a virtual machine:

\* Bound to a physical network adapter on the host, or

\* Bound to an internal virtual network on the host.

Example format:

-VLANEnabled

-VLANMode "Access" -VLANID 35

-VLANMode <String>

Specifies whether a virtual LAN (a VLAN) on a Hyper-V or VMware ESX hos

t supports traffic across a single VLAN ("Access" mode) or across multi

ple VLANs ("Trunk" mode).

Valid values: "Trunk" or "Access"

-VLANTrunkID <Int32[]>

Assigns a list of numerical identifiers in the range 1-4094 to a physic

al network adapter on a Hyper-V host.

Example format:

-VLANEnabled -VLANMode "Trunk" -VLANTrunkID 1,2,100,200,1124

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Change the network location on a host network adapter.

PS C:\> Get-VMMServer –Computername “VMMServer1.Contoso.com”

PS C:\> $VMHost = Get-VMHost -Computername "VMHost01.Contoso.com"

PS C:\> $VMHostNIC = Get-VMHostNetworkAdapter -VMHost $VMHost

PS C:\> Set-VMHostNetworkAdapter -VMHost $VMHost -VMHostNetworkAdapter $VMH

ostNIC -OverrideNetworkLocation $TRUE –NetworkLocation "Corp.Contoso.com"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost.

The third command gets the object that represents the host network adapter

on VMHost01 (this example assumes that VMHost01 has only one network adapte

r) and stores the adapter object in variable $VMHostNIC.

The last command overrides the default network location for this network ad

apter (which this example assumes is Contoso.com) and resets the network lo

cation to Corp.Contoso.com.

2: Create a new virtual network on a host network adapter and specify a VLA

N ID for the virtual network.

PS C:\> Get-VMMServer -Computername “VMMServer1.Contoso.com”

PS C:\> $VMHost = Get-VMHost -Computername "VMHost02.Contoso.com"

PS C:\> if ($VMHost.VirtualizationPlatformString -eq "Virtual Server")

{

Write-Warning "Set-VMHostNetworkAdapter does not support adding VLANs t

o a Virtual Server host."

}

Else

{

$HostAdapter = Get-VMHostNetworkAdapter -VMHost $VMHost | where {$\_.Na

me -like "Intel(R) PRO/1000\*" }

New-VirtualNetwork -Name "Network1" -VMHost $VMHost -VMHostNetworkAdapt

er $HostAdapter

Set-VMHostNetworkAdapter -VMHostNetworkAdapter $HostAdapter -VLANEnable

d -VLANMode "Access" -VLANID 35

}

The first command connects to VMMServer1.

The second command gets the object that represents VMHost02 and stores the

host object in $VMHost.

The third command uses an If statement to check the virtualization platform

:

\* If the platform is Virtual Server, the fourth command displays a

warning to the user that this host does not support VLANs and stops

execution.

\* If the platform is not Virtual Server (that is, the host is a Hyper-V

or VMware host), the following commands execute:

The Get-VMHostNetworkAdapter cmdlet gets the object that represents a

specific host network adapter on VMHost02 (this example assumes that

VMHost02 has only one network adapter) with a name that starts with

"Intel(R) PRO/1000") and stores the adapter object in $HostAdapter.

The New-VirtualNetwork cmdlet creates a new virtual network on

VMHost02 that is bound to the host adapter, and names the virtual

network "Network1".

The Set-VMHostNetworkAdapter cmdlet enables a VLAN, sets the mode to

"Access" (which routes traffic internally within a single VLAN), and

assigns a VLAN ID of 35 on this network adapter.

CAUTION: This example assumes that that your host is already connected to a

VLAN or, if not, that your host has two network adapters. If your host has

a single network adapter (as might be the case if you are experimenting wi

th VMM cmdlets in a lab setting), assigning the adapter to a VLAN that is u

navailable to the VMM server will prevent VMM from managing the host. You c

an perform the steps in this example on a host that has only one network ad

apter if you first install the Microsoft Loopback Adapter on your server an

d then substitute "Microsoft\*" in step 3.

3: Add additional VLAN tags to a host network adapter configured in "Trunk"

mode.

PS C:\> Get-VMMServer –Computername “VMMServer1.Contoso.com”

PS C:\> $VMHost = Get-VMHost -Computername "VMHost03.Contoso.com"

PS C:\> $VMHostNIC = Get-VMHostNetworkAdapter -VMHost $VMHost -Name "Broadc

om BCM5708C NetXtreme II GigE (NDIS VBD Client) #3"

PS C:\> $NewVlanTags = $VMHostNIC.VlanTags + @(177,355,1012)

PS C:\> Set-VMHostNetworkAdapter -VMHostNetworkAdapter $VMHostNIC -VLANEnab

led $TRUE -VLANMode "Trunk" -VLANTrunkID $NewVLANTags

The first command connects to VMMServer1.

The second command gets the object that represents VMHost03 and stores the

host object in $VMHost.

The third command gets the object that represents the host network adapter

by specifying the adapter name and stores the adapter object in $VMHostNIC.

The fourth command uses the VlanTags property of the object that represents

the host network adapter ($VMHostNIC.VlanTags) and concatenates a new arra

y. Concatenation retains exisiting VlanTags and adds additional tags to th

e array. The results of the concatenation are assigned to $NewVlanTags.

The last command passes the new list of VLAN tags to the -VLANTrunkID param

eter of the Set-VMHostNetworkAdapter command. The parameter -VLANMode must

be specified with the value "Trunk" whenever the -VLANTrunkID parameter is

used to modify the list of VLAN trunk numerical identifiers.

REMARKS

For more information, type: "get-help Set-VMHostNetworkAdapter -detailed".

For technical information, type: "get-help Set-VMHostNetworkAdapter -full".

## VMHostRating

### Get-VMHostRating

SYNOPSIS

Calculates the placement rating for one or more virtual machine hosts manag

ed by Virtual Machine Manager on which you might want to deploy a specific

virtual machine.

SYNTAX

Get-VMHostRating -DiskSpaceGB <Int32> -HardwareProfile <HardwareProfile> -V

MHostGroup [<HostGroup[] HostGroup>] -VMName <String> [-CPUPriority <Int32>

] [-DiskPriority <Int32>] [-ExpectedCPUUtilization <Int32>] [-ExpectedDiskI

O <Int32>] [-ExpectedNetworkUtilization <Int32>] [-IsMigration] [-JobGroup

<Guid>] [-MemoryPriority <Int32>] [-NetworkPriority <Int32>] [-OperatingSys

tem <OperatingSystem>] [-PlacementGoal <EnginePlacementGoals>] [-RequiredLU

NCount <Int32>] [-UseDefaultPath] [-VirtualizationPlatform <VirtualizationP

latform>] [<CommonParameters>]

Get-VMHostRating -VMHost [<Host[] Host>] -VMList <VM[]> [-CPUPriority <Int3

2>] [-DiskPriority <Int32>] [-ExpectedCPUUtilization <Int32>] [-ExpectedDis

kIO <Int32>] [-IsMigration] [-MemoryPriority <Int32>] [-NetworkPriority <In

t32>] [-PlacementGoal <EnginePlacementGoals>] [-UseDefaultPath] [<CommonPar

ameters>]

Get-VMHostRating -VM [<String VM>] -VMHost [<Host[] Host>] [-CPUPriority <I

nt32>] [-DiskPriority <Int32>] [-ExpectedCPUUtilization <Int32>] [-Expected

DiskIO <Int32>] [-ExpectedNetworkUtilization <Int32>] [-IsMigration] [-Memo

ryPriority <Int32>] [-NetworkPriority <Int32>] [-PlacementGoal <EnginePlace

mentGoals>] [-UseDefaultPath] [<CommonParameters>]

Get-VMHostRating -VM [<String VM>] -VMHostGroup [<HostGroup[] HostGroup>] [

-CPUPriority <Int32>] [-DiskPriority <Int32>] [-ExpectedCPUUtilization <Int

32>] [-ExpectedDiskIO <Int32>] [-ExpectedNetworkUtilization <Int32>] [-IsMi

gration] [-MemoryPriority <Int32>] [-NetworkPriority <Int32>] [-PlacementGo

al <EnginePlacementGoals>] [-UseDefaultPath] [<CommonParameters>]

Get-VMHostRating -DiskSpaceGB <Int32> -Template [<Template String>] -VMHost

[<Host[] Host>] -VMName <String> [-CPUPriority <Int32>] [-DiskPriority <In

t32>] [-ExpectedCPUUtilization <Int32>] [-ExpectedDiskIO <Int32>] [-Expecte

dNetworkUtilization <Int32>] [-IsMigration] [-MemoryPriority <Int32>] [-Net

workPriority <Int32>] [-PlacementGoal <EnginePlacementGoals>] [-UseDefaultP

ath] [-VirtualizationPlatform <VirtualizationPlatform>] [<CommonParameters>

]

Get-VMHostRating -DiskSpaceGB <Int32> -HardwareProfile <HardwareProfile> -V

MHost [<Host[] Host>] -VMName <String> [-CPUPriority <Int32>] [-DiskPriorit

y <Int32>] [-ExpectedCPUUtilization <Int32>] [-ExpectedDiskIO <Int32>] [-Ex

pectedNetworkUtilization <Int32>] [-IsMigration] [-JobGroup <Guid>] [-Memor

yPriority <Int32>] [-NetworkPriority <Int32>] [-OperatingSystem <OperatingS

ystem>] [-PlacementGoal <EnginePlacementGoals>] [-RequiredLUNCount <Int32>]

[-UseDefaultPath] [-VirtualizationPlatform <VirtualizationPlatform>] [<Com

monParameters>]

Get-VMHostRating -DiskSpaceGB <Int32> -Template [<Template String>] -VMHost

Group [<HostGroup[] HostGroup>] -VMName <String> [-CPUPriority <Int32>] [-D

iskPriority <Int32>] [-ExpectedCPUUtilization <Int32>] [-ExpectedDiskIO <In

t32>] [-ExpectedNetworkUtilization <Int32>] [-IsMigration] [-MemoryPriority

<Int32>] [-NetworkPriority <Int32>] [-PlacementGoal <EnginePlacementGoals>

] [-UseDefaultPath] [-VirtualizationPlatform <VirtualizationPlatform>] [<Co

mmonParameters>]

DETAILED DESCRIPTION

Calculates the placement rating for one or more virtual machine hosts manag

ed by Virtual Machine Manager on which you might want to deploy a specific

virtual machine.

The rating indicates the suitability of a computer to serve as a host for a

virtual machine that requires a specific hardware configuration. The ratin

g can be computed by individual host, for an array of hosts, or for each ho

st that belongs to a specific host group or set of host groups.

When you run the Get-VMHostRating cmdlet, Virtual Machine Manager returns a

n object that represents the virtual machine host rating (VMHostRating) for

each of the specified hosts based on the hardware configuration that you w

ant on the virtual machine. You can also specify additional placement optio

ns in order to modify how the ratings are calculated.

PARAMETERS

-DiskSpaceGB <Int32>

Specifies, in gigabytes (GB), the amount of hard disk space on the host

that can be used by a specific virtual machine.

Example: -DiskSpaceGB 20 (to specify 20 GB of disk space)

-HardwareProfile <HardwareProfile>

Specifies a hardware profile object.

-Template [<Template String>]

Specifies a VMM template object used to create virtual machines.

-VM [<String VM>]

Specifies a virtual machine object.

-VMHost [<Host[] Host>]

Specifies a virtual machine host object or an array of host objects. VM

M 2008 supports Hyper-V hosts, Virtual Server hosts, and VMware ESX Ser

ver hosts. For more information about each type of host, type: Get-Help

Add-VMHost -detailed. See the examples for a specific cmdlet to determ

ine how that cmdlet uses this parameter.

-VMHostGroup [<HostGroup[] HostGroup>]

Specifies a virtual machine host group object or an array of host group

objects.

-VMList <VM[]>

Specifies a list of virtual machines (an object array) to rate hosts ag

ainst when VMM calculates the placement rating. The host rating returne

d is based on placing all listed virtual machines on the host.

Example format: -VMList $MyVMs

-VMName <String>

Specifies the name of a virtual machine to be placed on a physical host

server. Use this parameter to verify that another virtual machine with

the same name is not already deployed on that host.

-CPUPriority <Int32>

Specifies the relative importance of CPU utilization for a virtual mach

ine on a host. To make CPU utilization a higher priority relative to ot

her factors (such as disk I/O performance, memory utilization, and netw

ork utilization), set this value to a higher number.

Default value: 5. Range: 0 through 10.

-DiskPriority <Int32>

Specifies the relative importance of disk input/output (I/O) performanc

e for a virtual machine on a host. To make disk I/O performance a highe

r priority relative to other factors (such as CPU utilization, memory u

tilization, and network utilization), set this value to a higher number

.

Default value: 2. Range: 0 through 10.

-ExpectedCPUUtilization <Int32>

Specifies (as a percentage) the amount of CPU on the host that you expe

ct this virtual machine to use. This value is used only when VMM determ

ines a suitable host for the virtual machine.

-ExpectedDiskIO <Int32>

Specifies the number of disk input/output operations per second (IOPS)

that you expect this virtual machine to use.

Example: -DiskIO 1500 (to specify 1500 IOPS).

-ExpectedNetworkUtilization <Int32>

Specifies, in megabits per second (Mbps), the amount of traffic on the

host’s physical network that you expect this virtual machine to use.

-IsMigration

Specifies that a rating indicating a computer’s suitability as a host t

o which to move a virtual machine will be calculated even if the source

and destination host is the same computer.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-MemoryPriority <Int32>

Specifies the relative importance of memory utilization by a virtual ma

chine on a host. To make memory utilization a higher priority relative

to other factors (such as CPU utilization, disk I/O performance, and ne

twork utilization), set this value to a higher number.

Default value: 8. Range: 0 through 10.

-NetworkPriority <Int32>

Specifies the relative importance of network utilization by a virtual m

achine on a host. To make network utilization a higher priority relativ

e to other factors (such as CPU utilization, disk I/O performance, and

memory utilization), set this value to a higher number.

Default value: 2. Range: 0 through 10.

-OperatingSystem <OperatingSystem>

Specifies the type of operating system for a virtual machine. To list t

he names of all available operating systems in VMM, type: Get-Operating

System

-PlacementGoal <EnginePlacementGoals>

Specifies the placement algorithm to use when VMM selects the most suit

able host on which to deploy a virtual machine. Load balancing among ho

sts lets VMM minimize the processing load on any one host. Consolidatio

n lets VMM maximize resources by combining multiple low-utilization wor

kloads on a single host.

Valid values: LoadBalance or Consolidate

-RequiredLUNCount <Int32>

Specifies the number of LUNs required by a virtual machine when evaluat

ing which computers (among all available computers) are suitable hosts

on which to deploy this virtual machine.

-UseDefaultPath

Specifies that only volumes for which a default path has been set on th

e host will be evaluated as possible candidates for virtual machine pla

cement. If you omit this parameter or if no default paths are set on th

e host, all volumes will be evaluated by the placement process.

Note: In VMM 2007, this parameter was plural: UseDefaultPaths.

-VirtualizationPlatform <VirtualizationPlatform>

Specifies the virtualization platform of a virtual machine host managed

by VMM.

Valid values: VirtualServer, Hyper-V, or VMwareESX.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Calculate host ratings for a specific server as a possible host for an e

xisting VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> $HostRating = Get-VMHostRating -VM $VM -VMHost $VMHost

PS C:\> Write-Output $HostRating

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents a virtual machine named

VM01 from the VMM database and stores the virtual machine object in variabl

e $VM.

The third command gets the object that represents the host named VMHost01 a

nd stores the host object in variable $VMHost.

The fourth command returns the placement rating for VMHost01 that indicates

its suitability as a host for VM01 and stores the rating information in va

riable $HostRating.

The last command displays the host ratings stored in $HostRating to the use

r.

2: Calculate host ratings for each server in a host group as a possible hos

t for an existing VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM | where { $\_.Name -eq "VM02" }

PS C:\> $VMHostGroup = Get-VMHostGroup -Name "HostGroup02"

PS C:\> $HostRatings = Get-VMHostRating -VM $VM -VMHostGroup $VMHostGroup

PS C:\> Write-Output $HostRatings

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM02 and stores the virtual machine object in $VM.

The third command gets the object that represents the host group named Host

Group02 and stores the host group object in $VMHostGroup.

The fourth command returns the placement ratings for all hosts in the speci

fied host group and indicates the suitability of each host in that host gro

up as a host for VM02. The command stores the rating information in $HostRa

tings.

The last command displays the host ratings stored in $HostRating to the use

r.

3: Calculate host ratings for each server in a host group as a possible hos

t for a new VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHostGroup = Get-VMHostGroup -Name "HostGroup03"

PS C:\> $HWProfile = Get-HardwareProfile -Name "MyHWProfile-1"

PS C:\> $HostRatings = Get-VMHostRating -VMHostGroup $VMHostGroup -Hardware

Profile $HWProfile -DiskSpaceGB 20 -VMName "MyVM-1" -CPUPriority 8 -MemoryP

riority 5 -DiskPriority 3 -NetworkPriority 1

PS C:\> Write-Output $HostRatings

The first command connects to VMMServer1.

The second command gets the object that represents the host group named Hos

tGroup03 and stores the host group object in $VMHostGroup.

The third command gets the hardware profile object that represents the hard

ware profile named "MyHWProfile-1" and stores the profile object in $HWProf

ile.

The fourth command returns the placement ratings for all hosts in the speci

fied host group for a new virtual machine and stores the placement ratings

in $HostRatings. Before determining the host ratings, this command modifies

the priorities for various factors by using the following parameters to sp

ecify these values:

\* DiskSpaceGB. The amount of hard disk space, in gigabytes, needed on

the host for a new virtual machine's hard disk drive is set to 200.

\* CPUPriority. The relative importance of CPU utilization on the host

for a new virtual machine is set to 8.

\* MemoryPriority. The relative importance of memory utilzation on the

host for a new virtual machine is set to 5.

\* DiskPriority. The relative importance of disk input/output (I/O)

performance on the host for a new virtual machine is set to 3.

\* NetworkPriority. The relative importance of network utilization on

the host for a new virtual machine is set to 1.

NOTE: For more information about each of these settings and default values,

see the description for each parameter.

The last command displays the host ratings stored in $HostRatings to the us

er.

4: Calculate host ratings for each host in an array as a possible host for

a VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $OS = Get-OperatingSystem | where {$\_.Name -eq "64-bit edition of W

indows Server 2008 Standard"}

PS C:\> $JobGroupID = [guid]::NewGuid().ToString()

PS C:\> New-VirtualDiskDrive -SCSI -Fixed -Bus 0 -Lun 2 -Size 10 -JobGroup

$JobGroupID -FileName "TestDiskDrive"

PS C:\> $VMHosts = Get-VMHost

PS C:\> $HWProfile = Get-HardwareProfile | where { $\_.Name -eq "MyHWProfile

-1" }

PS C:\> $HostRatings = Get-VMHostRating -DiskSpaceGB 10 -HardwareProfile $

HWProfile -VMHost $VMHosts -VMName "MyNewVM" -OperatingSystem $OS -JobGroup

$JobGroupID

PS C:\> Write-Output $HostRatings

The first command connects to VMMServer1.

The second command gets the operating system object that represents a 64-bi

t edition of Windows Server 2008 Standard edition and stores the object in

$OS.

The third command generates a GUID and stores the GUID string in $JobGroupI

D. The job group ID functions as an identifier that groups subsequent comma

nds into a single job group.

The fourth command will create a new virtual disk drive with the specified

properties, but uses the job group ID to specify that the virtual disk driv

e is not created until just before the Get-VMHostRating cmdlet (in the last

command) runs.

The fifth and sixth commands retrieve an array of host objects and a specif

ic hardware profile object to pass into the Get-VMHostRating cmdlet in the

next command.

The seventh command returns the placement ratings for all hosts in the spec

ified host list and indicates the suitability of each host in that host lis

t for the new virtual machine with the specified characteristics. The comma

nd stores the rating information in $HostRatings.

Before the Get-VMHostRating cmdlet returns the host ratings, the command us

es the JobGroup parameter to run the earlier New-VirtualDiskDrive command s

o that the Get-VMHostRating cmdlet includes the virtual disk drive and its

settings when calculating placement ratings.

The last command displays the host ratings stored in $HostRatings to the us

er.

5: Calculate host ratings for specific server as a possible host for a list

of VMs.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMList = Get-VM | where { $\_.Name -like "\*VM0\*" }

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost05.Contoso.com"

PS C:\> $HostRatings = Get-VMHostRating -VMHost $VMHost -VMList $VMList

PS C:\> Write-Output $HostRatings

The first command connects to VMMServer1.

The second command gets all objects that represent virtual machines whose n

ames include the string "VM0" and stores the virtual machine objects in $VM

List.

The third command gets the object that represents the host named VMHost05 a

nd stores the host object in $VMHost.

The fourth command returns the placement ratings for this host if all the v

irtual machines represented in the array object $VMList are to be placed on

this host. The command stores the rating information in $HostRatings.

The last command displays the host ratings stored in $HostRatings to the us

er.

6: Calculate host ratings for a specific server as a possible host for an e

xisting VM based on specific characteristics.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM06"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost06.Contoso.com"

PS C:\> $HostRating = Get-VMHostRating -VM $VM -VMHost $VMHost -CPUPriority

6 -DiskPriority 5 -MemoryPriority 4 -NetworkPriority 4 -PlacementGoal "Con

solidate"

PS C:\> Write-Output $HostRating

The first command connects to VMMServer1.

The second command gets the object that represents the virtual machine name

d VM06 and stores the virtual machine object in $VM.

The third command gets the object that represents the host named VMHost06 a

nd stores the host object in $VMHost.

The fourth command returns the placement rating for VMHost06 that indicates

its suitability as a host for VM06 based on a particular set of customized

priority ratings and based on consolidation as the placement goal (as oppo

sed to the default, load balancing). The command stores the rating informat

ion in $HostRating.

The last command displays the host rating stored in $HostRating to the user.

7: Calculate host ratings for a VM to be created based on a specific template.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MyTemplate = Get-Template | where {$\_.Name -eq "Template1"}

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> $HostRating = Get-VMHostRating -DiskSpaceGB 5 -Template $MyTemplate

-VMHost $VMHost -VMName "VM05"

PS C:\> Write-Output $HostRating

The first command connects to VMMServer1.

The second command gets the object that represents the template named "Temp

late1" and stores the template object in $MyTemplate.

The third command gets the object represents the host machine VMHost07 and

stores the host object in $VMHost.

The fourth command returns the placement ratings for a new VM if it were cr

eated by using Template1 and if it were to be placed on host VMHost01. The

command stores the ratings in $HostRating.

The last command displays the host ratings stored in $HostRating to the use

r.

NOTE: The DiskSpaceGB parameter is required even though the template might

already have a virtual hard disk with a specified amount of disk space. Req

uiring the DiskSpaceGB parameter ensures that a certain minumum amount of h

ard disk space is available on the host that can be used by the virtual mac

hine. If the amount of space specified for the virtual hard disk on the tem

plate is larger than the size specified by using the DiskSpaceGB parameter,

the larger of the two sizes is taken into consideration when computing the

host ratings.

8. Calculate host ratings for a specific server as a possible host for ever

y VM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost08.Contoso.com"

PS C:\> $VMArray = Get-VM

PS C:\> $RatingArray = @( ForEach ($VM in $VMArray) {Get-VMHostRating -VM $

VM -VMHost $VMHost} )

PS C:\> Write-Output $RatingArray

The first command connects to VMMServer1.

The second command gets the object that represents the host named VMHost08

and stores the host object in $VMHost.

The third command gets all virtual machine objects in your environment and

saves these virtual machine objects in $VMArray (an object array). (If your

environment has a very large number of VMs, you might want to use a filter

to select a subset of virtual machines.)

The fourth command returns the placement ratings for VMHost08 that indicate

s its suitability as a host for each of the virtual machine objects in $VMA

rray and stores the rating information in $RatingArray.

The final command displays the ratings calculated in the preceding command.

NOTE: This example computes the ratings for each virtual machine individual

ly on a host. If you want to place multiple virtual machines on a host, see

the example that uses VMList instead.

NOTE: For more information about the standard Windows PowerShell foreach lo

op statement, type: Get-Help about\_ForEach. The foreach loop statement is n

ot the same as the Foreach-Object cmdlet, which uses “foreach” as an alias.

REMARKS

For more information, type: "get-help Get-VMHostRating -detailed".

For technical information, type: "get-help Get-VMHostRating -full".

## VMHostVolume

### Get-VMHostVolume

SYNOPSIS

Gets drive volume objects from a host managed by Virtual Machine Manager.

SYNTAX

Get-VMHostVolume [[-Name] <String>] -VMHost [<String Host>] [<CommonParamet

ers>]

DETAILED DESCRIPTION

Gets one or more objects that represent drive volumes on a virtual machine

host managed by Virtual Machine Manager.

The information returned includes:

\* Name - The name of each host volume (such as C:\, D:\, E:\).

\* HostVolumeID - The volume ID (a GUID) for each host volume.

The host volume ID is unique across your Virtual Machine Manager

environment.

\* MountPoints - The mount points for each volume.

A single volume, such as C:\, can contain multiple mount points.

\* Capacity - The storage capacity of each volume.

\* FreeSpace - The amount of free space on each volume.

\* Volume Label - A user-defined label for this volume (if any).

\* IsSANMigrationPossible - A flag indicating whether or not SAN

migration is available.

\* IsClustered - A flag indicating whether the volume is local storage or

shared storage (that is, uses external storage, such as SAN or iSCSI)

and a clustered disk resource exists for this volume.

\* InUse - A flag that is set to TRUE when one of the highly available

virtual machines managed by Virtual Machine Manager is using this

volume.

\* VMHost - The FQDN name of the host on which each volume resides.

\* IsAvailableForPlacement - A flag indicating whether or not this volume

is available for as a location on which to deploy virtual machines

on this host.

\* ServerConnection - The Virtual Machine Manager server connection

that is managing the host that this volume belongs to.

\* ID - The ID (a hexedecimal number) for each volume. This

ID is an index number for each volume on the current system.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all volumes on the specified host server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01.Contoso.com"

PS C:\> Get-VMHostVolume -VMHost $VMHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost.

The last command gets from VMHost01 all objects that represent drive volume

s on this host and displays information about these volumes to the user.

NOTE: To translate the capacity and free space from bytes into larger units

of measure, divide the number of bytes by 1024 to get kilobytes (KB); divi

de the result by 1024 to get megabytes (MB); and divide that result by 1024

to get gigabytes (GB).

2: Get the specified volume on a host.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02.Contoso.com"

PS C:\> Get-VMHostVolume -VMHost $VMHost -Name "C:\"

The first command connects to VMMServer1.

The second command gets the object that represents VMHost02 and stores the

host object in $VMHost.

The last command gets from VMHost02 the object that represents the drive vo

lume named Volume1 and displays information about this volume to the user.

3: Get all volumes on VMware ESX hosts that contain the string "SharedStora

ge" in the volume name.

PS C:\> $VMHost = Get-VMHost -VMMServer VMMServer1.Contoso.com | where { $\_

.VirtualizationPlatform -eq "VMwareESX" }

PS C:\> $VMHost | Get-VMHostVolume | select -property Name, VMHost | where

{ $\_.Name -match "SharedStorage" }

The first command gets all host objects from VMMServer1, selects only those

host objects whose virtualization platform is VMware ESX, and then stores

those host objects in $VMHost.

NOTE: This example assumes that the names of all volumes on these ESX Serve

rs include the string "storage", but that only some of those volumes' names

include the string "SharedStorage."

The second command passes each ESX host object in $VMHost to the Get-VMHost

Volume cmdlet, which gets the objects that represent each volume on these h

osts and then, in turn, passes the volume objects to "select" (the alias fo

r the Select-Object cmdlet). The Select-Object cmdlet displays the volume n

ame and the host that volume resides on for those volumes whose name contai

ns the string "SharedStorage".

REMARKS

For more information, type: "get-help Get-VMHostVolume -detailed".

For technical information, type: "get-help Get-VMHostVolume -full".

### Set-VMHostVolume

SYNOPSIS

Modifies the setting for a volume on a host server that enables Virtual Mac

hine Manager to evaluate that volume as available storage during the virtua

l machine placement process.

SYNTAX

Set-VMHostVolume [-VMHostVolume] <VMHostVolume> -AvailableForPlacement <Boo

lean> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<Com

monParameters>]

DETAILED DESCRIPTION

Modifies the setting that determines whether or not Virtual Machine Manager

will evaluate a specific volume on a host server as available storage duri

ng the virtual machine placement process.

During the placement process, Virtual Machine Manager evaluates managed hos

ts, including the volumes on those managed hosts, when calculating a recomm

endation for the best location on which to deploy a virtual machine. If you

specify that a volume on the host will not be included when Virtual Machin

e Manager performs its automatic placement calculation, you can still, opti

onally, choose to manually deploy a virtual machine on that volume.

PARAMETERS

-VMHostVolume <VMHostVolume>

Specifies an object that represents a volume on a specific virtual mach

ine host.

-AvailableForPlacement <Boolean>

Specifies (when set to TRUE) that the VMM placement process will consid

er this host server (when used with Add-VMHost or Set-VMHost) or this v

olume on a host (when used with Set-VMHostVolume) to be eligible as a p

ossible location on which to deploy virtual machines. If this parameter

is set to FALSE, you can, optionally, choose to deploy virtual machine

s on this host or volume anyway. The default value is TRUE.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Make a volume (on a host) available for placement.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> $VMHostVol = Get-VMHostVolume -VMHost $VMHost

PS C:\> Set-VMHostVolume -VMHostVolume $VMHostVol -AvailableForPlacement $T

RUE

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object for VMHost01 (in the Contoso.com domain)

from the VMM database and stores the host object in variable $VMHost.

This example assumes that VMHost01 has only one volume, so the third comman

d gets the object that represents that volume and stores the volume object

in variable $VMHostVol.

The last command makes the volume on VMHost01 available for placement. Sett

ing the parameter AvailableForPlacement to TRUE enables the VMM placement p

rocess to evaluate this volume on VMHost01 as a possible candidate to host

virtual machines.

2: Make the second volume (on a host) available for placement.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost02"

PS C:\> $VMHostVols = Get-VMHostVolume -VMHost $VMHost

PS C:\> Set-VMHostVolume -VMHostVolume $VMHostVols[1] -AvailableForPlacemen

t $TRUE

The first command connects to VMMServer1.

The second command gets the object for VMHost02 and stores the host object

in variable $VMHost.

This example assumes that VMHost02 has at least two volumes. The third comm

and gets all objects that represent volumes on VMHost02 and stores the volu

me objects in variable $VMHostVols (an object array).

The last command makes the second volume ($VMHostVols[1]) on VMHost02 avail

able for placement.

REMARKS

For more information, type: "get-help Set-VMHostVolume -detailed".

For technical information, type: "get-help Set-VMHostVolume -full".

## VMMManagedComputer

### Get-VMMManagedComputer

SYNOPSIS

Gets managed computer objects from the Virtual Machine Manager database.

SYNTAX

Get-VMMManagedComputer [[-ComputerName] <String>] [-VMMServer [<String Serv

erConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent physical computers managed by Virtu

al Machine Manager.

Managed computers include the following types of computers:

\* VMWARE VIRTUALIZATION MANAGER. A server running VMware VirtualCenter

Server that VMM connects to in order to manage VMware ESX hosts and

the virtual machines deployed on those hosts.

\* VIRTUAL MACHINE HOST. A Virtual Server host, Hyper-V host, or

VMware ESX Server host on which you deploy virtual machines.

\* LIBRARY SERVER. A server used to make shares available to store VMM

library resources.

\* P2V SOURCE COMPUTER. Any physical computer that you want to "clone"

so that you can use its hardware and software settings to create one

or more virtual machines.

For more information, type:

Get-Help Add-VirtualizationManager -detailed

Get-Help Add-VMHost -detailed

Get-Help Add-LibraryServer -detailed

Get-Help Add-LibraryShare -detailed

Get-Help New-P2V -detailed

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all computers managed by VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMMManagedComputer

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets all objects that represent computers managed by VMM

Server1 from the VMM database and displays information about these managed

computers to the user. When you look at the output, note that the property

RoleString indicates whether the server is a library server, a host for vir

tual machines, both a library server and a host, or a VMware VirtualCenter

Server.

2: Update the agent software on all host servers managed by VMM.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMManagedComputer | Update-VMMManagedcomputer -Credential $Cre

dential -RunAsync

The first command connects to VMMServer1.

The second command uses Get-Credential to prompt you to supply a user name

and password and stores your credentials in $Credential. The required crede

ntials for this operation a domain account with rights to update software o

n computers managed by Virtual Machine Manager.

The last command performs the following operations:

\* Gets the objects that represent all computers managed by VMM.

\* Passes each managed computer objec to the

Update-VMMManagedComputer cmdlet, which updates

the VMM agent software on each computer.

As this command is processed, $Credential provides

your credentials to the Update-VMMManagedComputer cmdlet.

3: Get a specific computer managed by VMM by IP address.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMMManagedComputer -ComputerName "10.20.30.40"

The first command connects to VMMServer1.

The second command gets the object for a computer by specifying its IP addr

ess.

REMARKS

For more information, type: "get-help Get-VMMManagedComputer -detailed".

For technical information, type: "get-help Get-VMMManagedComputer -full".

### Reassociate-VMMManagedComputer

SYNOPSIS

Reassociates a managed computer on which Virtual Machine Manager agent soft

ware is installed (that is, a Windows-based host or library server) with a

different Virtual Machine Manager server.

SYNTAX

Reassociate-VMMManagedComputer [-VMMManagedComputer] <VMMManagedComputer> -

Credential <PSCredential> [-JobVariable <String>] [-PROTipID <Guid>] [-RunA

synchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Reassociates a managed computer on which Virtual Machine Manager agent soft

ware is installed (that is, a Windows-based host or library server) with a

different Virtual Machine Manager server.

When you initially add a host or library server to Virtual Machine Manager,

the host or library server is "associated" with the Virtual Machine Manage

r server that provides the Virtual Machine Manager database to which you ad

ded that host or library server. The Virtual Machine Manager database might

be installed either in a Microsoft SQL Server 2005 database on the Virtual

Machine Manager server itself or on a remote server running Microsoft SQL

Server 2005.

After a host or library server is added to (and therefore associated with)

a Virtual Machine Manager server, it cannot communicate with any other Virt

ual Machine Manager server. However, you can reassociate it with a differen

t Virtual Machine Manager server, as described in the following scenarios.

SCENARIO 1: DISASTER RECOVERY

-----------------------------

In this scenario, VMMServerA fails, or the Virtual Machine Manager service

running on VMMServerA fails. You might already have VMMServerB available as

a backup Virtual Machine Manager server. If not, you can reinstall the Vir

tual Machine Manager service on VMMServerB. At this point, the Virtual Mach

ine Manager database might be on VMMServerB, or, if you keep the database o

n a separate SQL Server, you can now point VMMServerB to the Virtual Machin

e Manager database on that separate SQL Server.

However, although you now have a functioning Virtual Machine Manager server

(VMMServerB) and database, hosts and library servers that were managed by

VMMServerA are still configured to communicate with VMMServerA. VMMServerB

recognizes these managed computers, but they are in an "Access Denied" stat

e. At this point, you can use the Reassociate-VMMManagedComputer cmdlet to

reassociate computers that were managed by VMMServerA with VMMServerB.

SCENARIO 2: RE-ORGANIZING SERVER GROUPINGS

------------------------------------------

In this scenario, VMMServerA and VMMServerB are two existing Virtual Machin

e Manager servers that manage different sets of hosts and library servers.

If, for example, VMHost01 is currently managed by VMMServerA, you can add V

MHost01 to VMMServerB by using the Add-Host cmdlet with that cmdlet's Reass

ociate parameter. If you do this, VMHost01's state on VMMServerA is now "Ac

cess Denied" and its state on VMMServerB is "Responding." VMHost01 is now m

anaged by VMMServerB, so you can remove it from VMMServerA.

Alternatively, if you decide that you want VMHost01 to be managed by VMMSer

verA after all, you can use Reassociate-VMMManagedComputer to reassociate H

ost01 with VMMServerA.

PARAMETERS

-VMMManagedComputer <VMMManagedComputer>

Specifies an object that represents a computer that is managed by VMM.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Reassociate all unassociated managed computers with a specific VMM serve

r.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMManagedComputer | where {$\_.State -eq "AccessDenied"} | Reas

sociate-VMMManagedComputer -Credential $Credential

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command uses Get-Credential to prompt you to supply a user name

and password and stores your credentials in variable $Credential. The requi

red credentials for this operation are a domain account with administrator

rights on the host server that you want to reassociate with a specific Virt

ual Machine Manager server and the password for that account.

The last command performs the following operations:

\* Gets all objects from the VMM database on VMMServer1 that

represent managed computers (both library servers and hosts).

\* Selects only objects for managed computers that are in the

Access Denied state.

\* Uses the Reassociate-VMMManagedComputer cmdlet to change the

association of any managed computer currently in the Access

Denied state to VMMServer1.

\* Uses $Credential to provide your credentials to

Reassociate-VMMManagedComputer.

REMARKS

For more information, type: "get-help Reassociate-VMMManagedComputer -detai

led".

For technical information, type: "get-help Reassociate-VMMManagedComputer -

full".

### Update-VMMManagedComputer

SYNOPSIS

Updates Virtual Machine Manager agent software installed on a Windows-based

managed computer.

SYNTAX

Update-VMMManagedComputer [-VMMManagedComputer] <VMMManagedComputer> -Crede

ntial <PSCredential> [-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynch

ronously] [<CommonParameters>]

DETAILED DESCRIPTION

Updates Virtual Machine Manager agent software installed on a Windows-based

managed computer to the current version of the software.

If you upgrade your Virtual Machine Manager server to a later version of th

e Virtual Machine Manager service, afterward you can use this command to up

date agent software on computers that are managed by that Virtual Machine M

anager server.

Managed computers that you can update by using this cmdlet include:

- Virtual Server hosts

- Hyper-V hosts

- Windows-based library servers

- Windows-based P2V source computers

You can use the Update-VMMManagedComputer cmdlet to update the Virtual Mach

ine Manager agent software on domain-joined trusted hosts and non-trusted d

omain-joined hosts, but not on hosts located on a perimeter network.

PARAMETERS

-VMMManagedComputer <VMMManagedComputer>

Specifies an object that represents a computer that is managed by VMM.

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Update all managed computers.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMMManagedComputer | foreach { Update-VMMManagedComputer -VMMMa

nagedComputer $\_ -Credential $Credential -RunAsynchronously }

The first command prompts you to provide credentials with appropriate permi

ssions to perform this operation and stores the credentials in variable $Cr

edential.

The second command connects to VMMServer1 in the Contoso.com domain and get

s the server object from the VMM database. The following commands uses this

server by default.

The last command gets all objects that represent computers that are current

ly managed by VMMServer1 from the VMM database and passes each managed comp

uter object to "foreach" (the ForEach-Object cmdlet), which uses the Update

-VMMManagedComputer cmdlet to update the agent software on each managed com

puter. As this command is processed, $Credential provides your credentials

to Update-VMMManagedComputer. Note: This example assumes that no managed co

mputers are located in a perimeter network.

NOTE: For more information about the standard Windows PowerShell ForEach-Ob

ject cmdlet (which uses "foreach" as its alias but is not the same as the W

indows PowerShell foreach loop statement), type: Get-Help ForEach-Object

2: Update a specific host.

PS C:\> $Credential = Get-Credential

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMMManagedHost = Get-VMMManagedComputer -ComputerName "VMHost02.Co

ntoso.com"

PS C:\> Update-VMMManagedComputer -VMMManagedComputer $VMMManagedHost -Cred

ential $Credential

The first command prompts you to provide credentials with appropriate permi

ssions to perform this operation and stores the credentials in $Credential.

The second command connects to VMMServer1.

The third command gets the object that represents the managed host named VM

Host02 and stores the host object in $VMMManagedHost.

The last command updates the agent software on VMHost02, obtaining your cre

dentials from $Credential.

REMARKS

For more information, type: "get-help Update-VMMManagedComputer -detailed".

For technical information, type: "get-help Update-VMMManagedComputer -full".

## VMMServer

### Backup-VMMServer

SYNOPSIS

Backs up the Virtual Machine Manager database.

SYNTAX

Backup-VMMServer -Path <String> [-JobVariable <String>] [-PROTipID <Guid>]

[-RunAsynchronously] [-VMMServer [<String ServerConnection>]] [<CommonParam

eters>]

DETAILED DESCRIPTION

Backs up the Virtual Machine Manager database on a Virtual Machine Manager

server to a local folder or to a remote network share. You must back up the

Virtual Machine Manager database to a server running Microsoft SQL Server

2005, and the folder to which you back up the database must be accessible t

o the SQL Server.

Determining Whether SQL Is Local or on a Remote Server

------------------------------------------------------

If you do not know whether the Virtual Machine Manager database is stored l

ocally or on a remote server running Microsoft SQL Server 2005, do the foll

owing:

1. On the Virtual Machine Manager server, open the Registry Editor.

2. Navigate to:

HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\

Microsoft System Center Virtual Machine Manager 2008

Server\Settings\Sql

3. Look at the value for OnRemoteServer:

\* If it is set to 1, the database is on a remote SQL server.

\* If it is set to 0, the database is on the local Virtual

Machine Manager server

Restoring the Backed-up Database

--------------------------------

After you use the Backup-VMMServer cmdlet to back up the Virtual Machine Ma

nager database (see Examples 1 and 2), you can use the SCVMMRecover.exe com

mand to restore the database (see Example 3). This command (which is not a

Windows PowerShell cmdlet) is installed with Virtual Machine Manager. By de

fault, SCVMMRecover.exe is installed at <%system-drive%>\Program Files\Micr

osoft System Center Virtual Machine Manager 2008\bin\.

CAUTION: If you backed up a Virtual Machine Manager 2007 database, you cann

ot restore it to a Virtual Machine Manager 2008 database.

IMPORTANT: To back up and restore a server functioning as a virtual machine

host or as a library server in a Virtual Machine Manager environment, use

your standard server backup procedures.

PARAMETERS

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Back up the VMM database to a local folder.

PS C:\> $VMMServer = Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMMServer | Backup-VMMServer –Path "D:\VMMBackups"

The first command connects to VMMServer1 in the Contoso.com domain, gets th

e server object from the VMM database, and stores the server object in vari

able $VMMServer.

The second command passes the VMM server object stored in $VMMServer to the

Backup-VMMServer cmdlet, which backs up the VMM database to the specified

path on VMMServer1.

IMPORTANT:

\* Backup-VMMServer must back up the database to a server running Microsoft

SQL Server. This example assumes that Microsoft SQL Server (for the

VMM database) is installed on VMMServer1 rather than on a remote server.

\* When you back up the database to a local folder, the folder must be

write-accessible to the Microsoft SQL Server service.

2: Back up the VMM database to a network share.

PS C:\> $VMMServer = Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMMServer | Backup-VMMServer –Path "\\SQLServer01\VMMBackups"

The first command connects to VMMServer1 and stores the server object in $V

MMServer.

The second command passes the VMM server object to Backup-VMMServer, which

backs up the VMM database to the specified share on a server called SQLSer

ver01.

IMPORTANT:

\* Backup-VMMServer must back up the database to a server running Microsoft

SQL Server, so this example assumes that Microsoft SQL Server (for the

VMM database) is installed on SQLServer01.

\* When you back up the database to a remote share, the share must be

write-accessible to the Microsoft SQL Server service.

3: Restore the VMM database.

C:\> SCVMMRecover.exe –Path <%backup-folder-path%>\<%backup-file-name%>.bak

-Confirm

Where:

<%backup-folder-path%> is the path on the server running Microsoft SQL Serv

er 2005 where the .bak file is saved.

<%backup-file-name%> is the name of the .bak file that was created during t

he backup operation.

Restores the VMM database to the VMM server.

NOTE: You must open a Command Prompt window (not a Windows PowerShell windo

w) and use the SCVMMRecover.exe command that is installed with VMM, but is

not a Windows PowerShell cmdlet, to perform this operation. You must run SC

VMMRecover.exe locally on the VMM server on which you want to restore the d

atabase.

This example assumes that SCVMMRecover.exe is installed in the default loca

tion for VMM at:

<%system-drive%>\Program Files\Microsoft System Center Virtual Machine Mana

ger 2008\bin\SCVMMRecover.exe

REMARKS

For more information, type: "get-help Backup-VMMServer -detailed".

For technical information, type: "get-help Backup-VMMServer -full".

### Get-VMMServer

SYNOPSIS

Connects to a Virtual Machine Manager server (if a connection does not alre

ady exist) and retrieves the object that represents this server from the Vi

rtual Machine Manager database.

SYNTAX

Get-VMMServer [-ComputerName] <String> [-ConnectAs <String>] [-Credential <

PSCredential>] [-TCPPort <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Connects to a Virtual Machine Manager server (if a connection does not alre

ady exist) and retrieves the object that represents this server from the Vi

rtual Machine Manager database. The default port used to connect to a Virtu

al Machine Manager server is TCP port 8100.

After a connection to the Virtual Machine Manager server is established, al

l future commands run at the Windows PowerShell command line that require t

he Virtual Machine Manager server object will automatically use the existin

g connection until you close that Windows PowerShell command-shell window.

The Virtual Machine Manager service running on the Virtual Machine manager

server supports the Virtual Machine Manager database. This database is stor

ed either in Microsoft SQL Server 2005 on the Virtual Machine Manager serve

r itself or on a separate server running Microsoft SQL Server 2005.

The Virtual Machine Manager service enables you to manage your virtual envi

ronment, including host servers (which host virtual machines), library serv

ers (which store Virtual Machine Manager library resources), and virtual ma

chines deployed on a host or stored in the library.

PARAMETERS

-ComputerName <String>

Specifies the name of a computer that VMM can uniquely identify on your

network.

Valid formats: FQDN, IPv4 or IPv6 address, or NetBIOS name.

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the computer name.

-ConnectAs <String>

Specifies the VMM user role to use, if the user is a member of more tha

n one role, when connecting to the VMM server from the command line.

Valid values: Administrator, DelegatedAdmin, or SelfServiceUser.

USER ROLE MEMBERSHIP PERMISSIONS

-------------------- -----------

DelegatedAdmin + Administrator Can manage all VMM objects

Multiple DelegatedAdmin roles Can manage objects from union of all sc

opes

Multiple SelfServiceUser roles Can manage objects from union of all sc

opes

-Credential <PSCredential>

Specifies an object that represents the user name and password of an ac

count that has permission to complete this task, or (in the case of Res

tart-Job) to complete a restarted task. For more information, type: Get

-Help Get-Credential

-TCPPort <Int32>

Specifies a numeric value that represents a TCP port.

Note: In VMM 2007, this parameter, when used with the Get-VMMServer cmd

let, was named Port.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Connect to a VMM server.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

Connects to the VMM server named VMMServer1 that is located in the Contoso

domain and gets the server object from the VMM database provided by VMMServ

er1.

NOTE: The VMM database is stored either in Microsoft SQL Server 2005 on the

VMM server itself or on another server running Microsoft SQL Server 2005.

2: Connect to a VMM server through a specific port.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com" -TCPPort 8100

Connects over TCP port 8100 to the VMM server named VMMServer1 and retrieve

s the server object from the VMM database.

3: Get the .NET object type, methods, and properties for the VMM server.

PS C:\> $VMMServer = Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMMServer | Get-Member

PS C:\> $VMMServer | Get-Member | Format-List

The first command gets the object that represents VMMServer1 and stores the

VMM server object in variable $VMMServer.

The second command passes the VMM server object to the Get-Member cmdlet, w

hich retrieves and displays the following:

\* TypeName: The name of the .NET type of the VMM server object:

Microsoft.SystemCenter.VirtualMachineManager.Remoting.ServerConnection

\* MemberType: A list of the name and definition for each method and each

property associated wtih this object type.

The third command retrieves and displays the same information as the second

command, except that it pipes the output to the Format-List cmdlet so that

you can see the complete definition for each method and each property for

the VMM server object.

REMARKS

For more information, type: "get-help Get-VMMServer -detailed".

For technical information, type: "get-help Get-VMMServer -full".

### Set-VMMServer

SYNOPSIS

Changes properties of the Virtual Machine Manager server.

SYNTAX

Set-VMMServer [-CEIPOptIn <Boolean>] [-CPUPriority <Int32>] [-DiskIOPriorit

y <Int32>] [-JobVariable <String>] [-LibraryRefresherEnabled <Boolean>] [-L

ibraryRefresherFrequency <Int32>] [-MemoryPriority <Int32>] [-NetworkPriori

ty <Int32>] [-OpsMgrReportingEnabled] [-OpsMgrReportingServerURL <String>]

[-OpsMgrServer] [-PhysicalAddressRangeEnd <String>] [-PhysicalAddressRangeS

tart] [-PlacementGoal <EnginePlacementGoals>] [-PROAutomationLevel <Int32>]

[-PROMonitoringLevel <Int32>] [-PROTipID <Guid>] [-RunAsynchronously] [-Se

lfServiceContactEmail <String>] [-VMConnectDefaultPort <Int32>] [-VMMServer

[<String ServerConnection>]] [-VMRCAccessAccount <String>] [-VMRCDefaultPo

rt <Int32>] [<CommonParameters>]

DETAILED DESCRIPTION

Changes one or more properties of the Virtual Machine Manager server. Virtu

al Machine Manager settings that you can change with the Set-VMMServer cmdl

et are summarized as follows.

Placement Goal

--------------

You can use the Set-VMMServer cmdlet to specify whether to optimize load ba

lancing among hosts or to consolidate multiple loads onto a single host. Th

e Virtual Machine Manager placement process takes this setting into account

when it evaluates which hosts are the most suitable servers on which to de

ploy a virtual machine.

Relative Importance of Resource Usage

-------------------------------------

You can use the Set-VMMServer cmdlet to configure settings that specify the

importance of specific resources relative to each other. Virtual Machine M

anager takes these resource priorities into account when it calculates whic

h host servers are the best potential hosts for virtual machines:

\* CPU priority: Sets a higher or lower priority for CPU usage by a

virtual machine on a host.

\* Disk I/O priority: Sets a higher or lower priority for disk

input/output (I/O) performance by a virtual machine on a host.

\* Memory priority: Sets a higher or lower priority for the amount of

memory needed by a virtual machine on a host.

\* Network priority: Sets a higher or lower priority for network

utilization by a virtual machine on a host.

Microsoft Customer Experience Improvement Program (CEIP) Participation

----------------------------------------------------------------------

You can use the Set-VMMServer cmdlet to specify whether to enable Service Q

uality Metrics (SQM) for this Virtual Machine Manager. By default, SQM is e

nabled and therefore collects information about problems customers encounte

r in order to improve the software in a later release.

Library Refresher

-----------------

You can use the Set-VMMServer cmdlet to specify whether the Virtual Machine

Manager library refresher is enabled and, if so, how often objects in the

library are refreshed.

System Center Operations Manager 2007 Reporting

-----------------------------------------------

You can use the Set-VMMServer cmdlet to enable Operations Manager reporting

, specify the Operations Manager server to which VMM connects, and specify

the URL of the Operations Manager reporting server.

For more information about using Operations Manager in conjunction with Vir

tual Machine Manager:

\* See the Operations Manager deployment and configuration URLs in

"Setting Up Reporting in VMM" (topic in the VMM UI help)

Performance and Resource Optimization (PRO)

-------------------------------------------

If your Virtual Machine Manager environment is configured to work with Oper

ations Manager and to receive Performance and Resource Optimization tips (P

RO tips), you can use the Set-VMMServer cmdlet to enable monitoring by PRO

in order to receive PRO tips. A PRO tip recommends an action in response to

an alert generated by Operations Manager. You can also specify whether and

at what level to implement PRO tips automatically.

For more information about PRO tips:

\* Type: Get-Help Get-PROTip -detailed

\* See "About Performance and Resource Optimization" (topic in the VMM UI He

lp)

MAC Address Range

-----------------

You can use the Set-VMMServer cmdlet to configure a range of physical addre

sses (MAC addresses) that Virtual Machine Manager can allocate to virtual m

achines directly or through Virtual Machine Manager templates or hardware p

rofiles used to create virtual machines.

Contact for Self-Service Users

------------------------------

You can use the Set-VMMServer cmdlet to specify the e-mail address of an ad

ministrator who supports self-service users.

Hyper-V Default Port and Virtual Server Default Port and Account

----------------------------------------------------------------

You can use the Set-VMMServer cmdlet to configure the default values for th

e following:

\* The TCP port (VMConnectDefaultPort) that Hyper-V hosts use to connect

to Virtual Machine Manager.

\* The TCP port (VMRCDefaultPort) that Virtual Server hosts use to

connect to Virtual Machine Manager.

\* The default account (VMRCDefaultPort) granted VMRC access to

virtual machines on Virtual Server hosts in a host group.

PARAMETERS

-CEIPOptIn <Boolean>

Enables Service Quality Metrics (SQM) for this server and thus particip

ate in the Microsoft Customer Experience Improvement Program (CEIP), wh

ich collects information about problems customers encounter in order to

improve the software in a later release. The default value is TRUE.

-CPUPriority <Int32>

Specifies the relative importance of CPU utilization for a virtual mach

ine on a host. To make CPU utilization a higher priority relative to ot

her factors (such as disk I/O performance, memory utilization, and netw

ork utilization), set this value to a higher number.

Default value: 5. Range: 0 through 10.

-DiskIOPriority <Int32>

Specifies the relative importance of disk input/output (I/O) performanc

e for a virtual machine on a host. To make disk I/O performance a highe

r priority relative to other factors (such as CPU utilization, memory u

tilization, and network utilization), set this value to a higher number

.

Default value: 2. Range: 0 through 10.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryRefresherEnabled <Boolean>

Specifies (when set to TRUE) that VMM library objects will be refreshed.

-LibraryRefresherFrequency <Int32>

Specifies, in hours, the frequency at which objects in the VMM library

are refreshed. The default setting is 1 hour.

-MemoryPriority <Int32>

Specifies the relative importance of memory utilization by a virtual ma

chine on a host. To make memory utilization a higher priority relative

to other factors (such as CPU utilization, disk I/O performance, and ne

twork utilization), set this value to a higher number.

Default value: 8. Range: 0 through 10.

-NetworkPriority <Int32>

Specifies the relative importance of network utilization by a virtual m

achine on a host. To make network utilization a higher priority relativ

e to other factors (such as CPU utilization, disk I/O performance, and

memory utilization), set this value to a higher number.

Default value: 2. Range: 0 through 10.

-OpsMgrReportingEnabled

Enables System Center Operations Manager reporting.

Note: In VMM 2007, this parameter was named MOMReportingEnabled.

-OpsMgrReportingServerURL <String>

Specifies the URL of a System Center Operations Manager reporting serve

r.

Default format: http://<ReportingServerName>/ReportServer

Alternatively, you can use a secure http URL (https) to connect to a re

porting site that is secured by Secure Sockets Layer (SSL).

Note: In VMM 2007, this parameter was named MOMReportingServerURL.

-OpsMgrServer

Specifies the fully qualified domain name (FQDN) of the System Center O

perations Manager server to which VMM connects.

-PhysicalAddressRangeEnd <String>

Specifies the last address in a range of static physical addresses (MAC

addresses).

-PhysicalAddressRangeStart

Specifies the first address in a range of static physical addresses (MA

C addresses).

-PlacementGoal <EnginePlacementGoals>

Specifies the placement algorithm to use when VMM selects the most suit

able host on which to deploy a virtual machine. Load balancing among ho

sts lets VMM minimize the processing load on any one host. Consolidatio

n lets VMM maximize resources by combining multiple low-utilization wor

kloads on a single host.

Valid values: LoadBalance or Consolidate

-PROAutomationLevel <Int32>

Specifies the severity level of tips that PRO will implement.

Valid values:

0 (Off) Respond to a PRO tip manually

1 (Critical) Implement critical PRO tips automatically

2 (CriticalAndWarning) Implement critical or warning PRO tips automatic

ally

Note: Applies only to Hyper-V or VMware ESX hosts or to VMs deployed on

those hosts.

-PROMonitoringLevel <Int32>

Specifies the severity level of tips that PRO will monitor.

Valid values:

0 (Off) Do not monitor PRO tips

1 (Critical) Monitor critical PRO tips

2 (CriticalAndWarning) Monitor critical and warning PRO tips

Note: Applies only to Hyper-V or VMware ESX hosts or to VMs deployed on

those hosts.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-SelfServiceContactEmail <String>

Specifies the e-mail address of a VMM administrator that self-service u

sers can contact if they encounter a problem.

-VMConnectDefaultPort <Int32>

Specifies the default value for the TCP port used for Virtual Machine C

onnection (VMConnect) sessions on all Hyper-V hosts managed by this VMM

server. Typically, the default port is 2179, but you can use this para

meter to change the default. This parameter does not apply to Virtual S

erver hosts or VMware ESX Server hosts.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VMRCAccessAccount <String>

Specifies the default account that is granted Virtual Machine Remote Co

ntrol (VMRC) access to new virtual machines on one or more Virtual Serv

er hosts in a specific host group. This parameter does not apply to Hyp

er-V hosts or VMware ESX Server hosts.

-VMRCDefaultPort <Int32>

Specifies the default value for the TCP port used for Virtual Machine R

emote Control (VMRC) sessions on all Virtual Server hosts managed by th

is VMM server. Typically, the default port used for VMRC is 5900, but y

ou can use this parameter to change the default. This parameter does no

t apply to Hyper-V hosts or VMware ESX Server hosts.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Specify the importance of CPU utilization when determining on which host

to place a virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -CPUPbriority 6

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command sets the value of the CPUPriority parameter for VMMServe

r1 to 6. This setting specifies that, when VMM calculates the most suitable

host on which to place a virtual machine, it will evaluate the importance

of CPU utilization at 6 when compared to the importance specified for other

factors (such as disk I/O performance, memory utilization, and network uti

lization).

NOTE: To make CPU utilization a higher priority relative to other factors,

set the value for CPUPriority to a higher number.

2: Set the frequency at which the library is refreshed.

PS C:\> Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -LibraryRefresherEnabled $TRUE -LibraryRefresherFrequ

ency 24

The first command connects to VMMServer1.

The second command enables library refreshing for VMMServer1 and sets the r

efreshing frequency rate to every 24 hours.

3: Set the range of available MAC addresses on the VMM server.

PS C:\> Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -VMMServer $VMMServer -PhysicalAddressRangeStart 22-2

2-22-22-22-22 -PhysicalAddressRangeEnd 22-22-22-22-22-99

The first command connects to VMMServer1.

The second command sets the global physical address (MAC address) range on

VMMServer1 to the static addresses 22-22-22-22-22-22 through 22-22-22-22-2

2-99.

After you specify the range of available MAC addresses, these addresses are

available for use on any virtual network adapter configured on any virtual

machine that is deployed on any host managed by VMMServer1, or for use on

any hardware profile or template stored on any library server managed by VM

MServer1.

4: Opt out of the Customer Experience Improvement Program.

PS C:\> Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -VMMServer $VMMServer -CEIPOptIn $FALSE

The first command connects to VMMServer1.

The second command disables the use of Service Quality Metrics (SQM) on VMM

Server1 in order to opt out of participation in the Microsoft Customer Expe

rience Improvement Program (CEIP) by setting the CEIPOptIn parameter to $FA

LSE.

5: Specify a self-service contact e-mail address.

PS C:\> $VMMServer = Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -VMMServer $VMMServer -SelfServiceContactEmail "Admin

Help@Contoso.com"

The first command connects to VMMServer1 and stores the server object in

$VMMServer.

The second command sets the self service contact e-mail address to "AdminHe

lp@Contoso.com".

6: Enable System Center Operations Manager reporting for VMM.

PS C:\> $VMMServer = Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -VMMServer $VMMServer -OpsMgrReportingEnabled $TRUE -

OpsMgrReportingServerUrl "http://OpsMgr.Contoso.com/ReportServer" -OpsMgrSe

rver "OpsMgr.Contoso.com"

The first command connects to VMMServer1 and stores the server object in $V

MMServer.

The second command enables Operations Manager reporting by:

\* Setting the value for -OpsMgrReportingEnabled to $TRUE

\* Setting the Operations Manager Reporting URL to:

http://OpsMgr.Contoso.com/ReportServer

\* Setting the Operations Manager Server to Opsmgr.Contoso.com

7: Set all placement settings back to the default settings.

PS C:\> $VMMServer = Get-VMMServer -ComputerName “VMMServer1.Contoso.com"

PS C:\> Set-VMMServer -VMMServer $VMMServer -CPUPriority 8 -DiskIOPriority

2 -MemoryPriority 8 -NetworkPriority 2

The first command connects to VMMServer1 and stores the server object in $V

MMServer.

The second command sets the values for CPUPriority to 8, DiskIOPriority to

2, MemoryPriority to 8, and Network Priority to 2. These are the factory d

efault settings for placement.

REMARKS

For more information, type: "get-help Set-VMMServer -detailed".

For technical information, type: "get-help Set-VMMServer -full".

## VMMUserRole

### Get-VMMUserRole

SYNOPSIS

Gets an object that represents a Virtual Machine Manager user role.

SYNTAX

Get-VMMUserRole [[-Name] <String>] [-VMMServer [<String ServerConnection>]]

[<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects that represent Virtual Machine Manager user roles.

VMM 2008 uses role-based security to define the boundaries within which me

mbers of a given user role can operate and the set of allowed operations me

mbers of a user role can perform.

VMM ADMINISTRATOR (Administrator)

---------------------------------

In VMM 2008, members of the Administrator user role can perform all operati

ons on all VMM objects. Only one Administrator role exists, and you cannot

create or remove this user role.

You can use the Set-VMMUserRole cmdlet to add or remove members to the Admi

nistrator user role.

DELEGATED ADMINISTRATOR (DelegatedAdmin)

----------------------------------------

Members of a Delegated Administrator user role can perform all operations o

n all VMM objects within the specified scope (one or more host groups or li

brary servers).

You can use the New-VMMUserRole cmdlet to create a new delegated administra

tor user role and the Set-VMMUserRole cmdlet to add or remove members to th

is user role.

SELF-SERVICE USER (SelfServiceUser)

-----------------------------------

Members of a Self Service User role can perform all permitted operations on

a specific set of virtual machines deployed on one or more hosts within th

e specified scope (one or more host groups) and, optionally, can store virt

ual machines in the library. The permitted operations are those used to man

age virtual machines.

You can use the New-VMMUserRole cmdlet to create a new self-service user ro

le and the Set-VMMUserRole cmdlet to add or remove members to this user rol

e and to specify or modify the list of available permissions and other opti

ons for members of this user role.

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all VMM user roles.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMMUserRole

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets all VMM user role objects on this VMM server and di

splays information about each user role.

2: Get a specific user role by filtering for a specific user role name.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMMUserRole | where { $\_.Name -eq "Administrator" }

The first command connects to VMMServer1.

The second command gets the object for all VMM user roles, selects the user

role named Administrator, and then displays information about that user ro

le to the user.

3: Get a specific user role by name.

PS C:\> Get-VMMUserRole -VMMServer VMMServer1.Contoso.com -Name "ContosoSel

fServiceUsers"

Gets the object that represents the user role named ContosoSelfServiceUsers

from VMMServer1 and displays information about this user role to the user

.

4. Display properties and other information about user role objects.

PS C:\> $UserRoles = Get-VMMUserRole -VMMServer VMMServer1.Contoso.com

PS C:\> $UserRoles | select Name, UserRoleProfile, ParentUserRole, HostGrou

p

PS C:\> $UserRoles | Get-Member

The first command gets the objects for all user roles on VMMServer1 and sto

res the user role objects in $UserRoles.

The second command passes each user role object in $UserRoles to "select" (

the alias for the Select-Object cmdlet) and then, for each user role, displ

ays the name, user role profile, parent user role, and host group.

The last command passes each user role in $UserRoles to the Get-Member cmdl

et, which displays the three .NET types for VMM user roles:

TypeName: Microsoft.SystemCenter.VirtualMachineManager.UserRole

TypeName: Microsoft.SystemCenter.VirtualMachineManager.SelfServiceUserRole

TypeName: Microsoft.SystemCenter.VirtualMachineManager.DelegatedAdminUserRole

The last command also displays the methods and properties associated with e

ach user role type.

REMARKS

For more information, type: "get-help Get-VMMUserRole -detailed".

For technical information, type: "get-help Get-VMMUserRole -full".

### New-VMMUserRole

SYNOPSIS

Creates a Self Service User role or a Delegated Administrator user role for

a group of Virtual Machine Manager users.

SYNTAX

New-VMMUserRole [-Name] <String> -UserRoleProfile <String> [-Description <S

tring>] [-JobGroup <Guid>] [-JobVariable <String>] [-ParentUserRole <UserRo

le>] [-PROTipID <Guid>] [-RunAsynchronously] [-VMMServer [<String ServerCon

nection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a Self Service User role or a Delegated Administrator user role for

a group of Virtual Machine Manager users. Only one Administrator role exis

ts; you cannot create another Administrator role or delete the existing one

.

If you are a member of a Delegated Administrator user role, you can create

a new user role with a scope that is a subset of the scope of the parent us

er role. In this case, you must use the New-VMMUserRole cmdlet with the Par

entUserRole parameter to create the new user role, or the command will fail

.

After you create a new user role, you can use the Set-VMMUserRole cmdlet to

rename the user role, to add or remove members, and to add or modify the s

cope of objects that members of the role can manage. For a self-service use

r role, you can also define a quota that limits the number of virtual machi

nes self-service users can create, and you can specify which actions member

s of a self-service user role can take on their virtual machines. Although

you cannot create or remove the Administrator role or limit its scope, you

can use Set-VMMUserRole to add or remove members to that role.

VMM 2007 ADMINS AND SELF-SERVICE USERS UPGRADE TO VMM 2008 AUTOMATICALLY

------------------------------------------------------------------------

The Active Directory security group called Virtual Machine Manager Administ

rators used in VMM 2007 is replaced by the Administrators user role in VMM

2008. The self-service policy feature used in VMM 2007 is replaced by the S

elf Service User role.

When you upgrade a VMM server from System Center Virtual Machine Manager 20

07 to System Center Virtual Machine Manager 2008:

\* The upgrade process automatically adds all Active Directory users who

were members of the Virtual Machine Manager Administrators security

group in VMM 2007 to the VMM 2008 Administrator user role. You can

then use Set-VMMUserRole to add or remove other members.

\* The upgrade process does not convert or create any Delegated

Administrator user roles because there is no counterpart to the

Delegated Administrator user role in VMM 2007

\* The upgrade process automatically converts existing VMM 2007 self-

service policies to VMM 2008 Self Service User roles. The new Self

Service User role has the same name, settings, and members as the

earlier self-service policy from which it is derived.

For example, if a self-service policy specifies a quota that limits

the number of virtual machines a self-service user can create, the

same restriction is carried over to the new Self Service User role.

For more information about user roles, type:

Get-Help about\_VMM\_2008\_Role\_Based\_Security

PARAMETERS

-Name <String>

Specifies the name of a VMM object.

-UserRoleProfile <String>

Specifies the type of profile to use as the basis for the user role.

Valid values: DelegatedAdmin or SelfServiceUser.

-Description <String>

Specifies a description for the specified object.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-ParentUserRole <UserRole>

Specifies an existing VMM user role as the parent of a new user role. T

his parameter is required when a Delegated Administrator creates a new

user role that has a scope which is a subset of the parent user role sc

ope.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Create a delegated administrator user role.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> New-VMMUserrole -Name "ContosoAdmin" -Description "Delegated Admini

strators for the Contoso.com domain" -UserRoleProfile DelegatedAdmin

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command creates a delegated administrator user role named "Conto

soAdmin", provides the description "Delegated Administrators for the Contos

o.com domain", and uses the -UserRoleProfile parameter to designate the use

r role type as delegated administrator.

2: Create a Self Service User user role whose members can manage objects in

the Lab host group.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Scope = New-VMHostGroup -Name "Lab"

PS C:\> $SelfServiceRole = New-VMMUserRole -Name "ContosoSelfServiceUsers"

-UserRoleProfile SelfServiceUser

PS C:\> Set-VMMUserRole -UserRole $SelfServiceRole -AddScope $Scope -AddMem

ber Contoso\User1,Contoso\User2

The first command connects to VMMServer1.

The second command creates a host group named "Lab" and stores the host gr

oup object in variable $Scope.

The third command creates a new user role named "ContosoSelfServiceUsers",

uses the -UserRoleProfile parameter to designate the new user role type as

Self Service User, and stores the new user role object in variable $SelfSe

rviceRole.

The last command sets the scope for ContosoSelfServiceUsers to the Lab host

group and adds User1 and User2 (both members of the Contoso.com domain) as

members to the ContosoSelfServiceUsers user role.

REMARKS

For more information, type: "get-help New-VMMUserRole -detailed".

For technical information, type: "get-help New-VMMUserRole -full".

### Remove-VMMUserRole

SYNOPSIS

Removes an existing Self Service User or Delegated Administrator user role

from Virtual Machine Manager.

SYNTAX

Remove-VMMUserRole [-UserRole] <UserRole> [-Confirm] [-JobGroup <Guid>] [-J

obVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonParame

ters>]

DETAILED DESCRIPTION

Removes an existing Self Service User or Delegated Administrator user role

from Virtual Machine Manager. You cannot remove the Administrator role.

PARAMETERS

-UserRole <UserRole>

Specifies a user role object.

-Confirm

Prompts for confirmation before running the command.

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove the specified user role.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $RemoveRole = Get-VMMUserRole | where {$\_.Name -eq "ContosoAdmin"}

PS C:\> Remove-VMMUserrole -Userrole $RemoveRole

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets all objects that represent VMM user roles; uses the

"where filter (the Where-Object cmdlet) to select the user role named "Con

toso Admin"; and then stores the object for that user role in variable $Rem

oveRole

The last command removes the "Contoso Admin" user role from the VMM database.

REMARKS

For more information, type: "get-help Remove-VMMUserRole -detailed".

For technical information, type: "get-help Remove-VMMUserRole -full".

### Set-VMMUserRole

SYNOPSIS

Modifies the settings for an existing Virtual Machine Manager user role.

SYNTAX

Set-VMMUserRole -JobGroup <Guid> [-AddMember [<NTAccount[] String>]] [-AddS

cope [<LibraryServer Template HostGroup>]] [-Description <String>] [-JobVar

iable <String>] [-LibraryStoreSharePath <String>] [-Name <String>] [-PROTip

ID <Guid>] [-QuotaPerUser <Boolean>] [-QuotaPoint <Int32>] [-RemoveMember [

<NTAccount[] String>]] [-RemoveQuotaPoint] [-RemoveScope [<Template HostGro

up LibraryServer>]] [-RunAsynchronously] [-VMMServer [<String ServerConnect

ion>]] [-VMPermission [<SelfServicePermission[] String>]] [<CommonParameter

s>]

Set-VMMUserRole -UserRole <UserRole> [-AddMember [<NTAccount[] String>]] [-

AddScope [<LibraryServer Template HostGroup>]] [-Description <String>] [-Jo

bVariable <String>] [-LibraryStoreSharePath <String>] [-Name <String>] [-PR

OTipID <Guid>] [-QuotaPerUser <Boolean>] [-QuotaPoint <Int32>] [-RemoveMemb

er [<NTAccount[] String>]] [-RemoveQuotaPoint] [-RemoveScope [<Template Hos

tGroup LibraryServer>]] [-RunAsynchronously] [-VMPermission [<SelfServicePe

rmission[] String>]] [<CommonParameters>]

DETAILED DESCRIPTION

Modifies the settings for an existing Virtual Machine Manager user role. Th

e settings that you can modify depend on the type of VMM user role.

VMM ADMINISTRATOR (Administrator)

---------------------------------

You can add or remove members to the Administrator user role, but you canno

t limit the scope of objects that members of this role can manage.

DELEGATED ADMINISTRATOR (DelegatedAdmin)

----------------------------------------

You can expand or restrict the scope for, and add to or remove members from

, a Delegated Administrator user role. You can grant members of this user r

ole permission to manage all of the objects in one or more host groups and/

or allow users to manage all of the objects stored on one or more library s

ervers. Within that framework, you cannot limit the actions that members of

the Delegated Administrator user role can perform.

SELF-SERVICE USER (SelfServiceUser)

-----------------------------------

You can add or remove members as well as expand or limit the scope and acti

ons granted to members of a Self Service User role. You can grant members o

f a self-service user role permission to manage all of the objects in one o

r more host groups; permission to create virtual machines; permission to st

ore virtual machines on a specific library share in the library; and permis

sion to use one or more template objects to create virtual machines. Within

that framework, you can grant members of a Self Service User role one or m

ore of the virtual machine permissions that define the actions that self-se

rvice users can take. You can also limit the number of virtual machines tha

t self-service users can create by setting a quota that applies to each use

r or to all users collectively.

The actions that you can grant include the virtual machine permissions call

ed Create, Start, Stop, PauseAndResume, Remove, Shutdown, Checkpoint, Store

(which allows self-service users to store their virtual machines in the li

brary and, later, to deploy a stored virtual machine on a host server, unle

ss the quota points setting blocks deploying additional virtual machines),

AllowLocalAdmin (which lets self-service users act as local Administrator o

n their virtual machines), and RemoteControl (which allows self-service use

rs to have remote access to a running virtual machine).

PARAMETERS

-JobGroup <Guid>

Specifies an identifier for a series of commands that will run as a set

just before the final command that includes the same job group identif

ier runs. For information about how VMM uses job groups, including a li

st of job groups available for specific cmdlets, type: Get-Help about\_V

MM

-UserRole <UserRole>

Specifies a user role object.

-AddMember [<NTAccount[] String>]

Adds one or more Active Directory domain users or groups to the user ro

le.

Example formats:

-AddMember Domain1\User1

-AddMember User1

-AddMember User1@Domain1

-AddMember Domain1\LabGroupAlias

-AddMember LabGroupAlias (Active Directory security group, not e-mail

alias)

-AddScope [<LibraryServer Template HostGroup>]

Adds one or more VMM objects to the scope of objects that members of th

is user role can manage.

-Description <String>

Specifies a description for the specified object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryStoreSharePath <String>

Specifies the path to a library share that members of a Self Service Us

er user role can use to store their virtual machines.

Example format: "\\LibraryServerName\LibraryShareName"

-Name <String>

Specifies the name of a VMM object.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-QuotaPerUser <Boolean>

Calculates quota points for each member of a Self-Service User role (wh

en set to TRUE) or for all members of a Self-Service User role (when se

t to FALSE). The default value is TRUE

-QuotaPoint <Int32>

Specifies a quota that limits the number of virtual machines self-servi

ce users can create.

-RemoveMember [<NTAccount[] String>]

Removes one or more Active Directory domain users or groups from the us

er role.

Example formats:

-RemoveMember Domain1\User1

-RemoveMember User1

-RemoveMember User1@Domain1

-RemoveMember Domain1\LabGroupAlias

-RemoveMember LabGroupAlias (Active Directory security group, not e-mai

l alias)

-RemoveQuotaPoint

Switches off the quota setting.

-RemoveScope [<Template HostGroup LibraryServer>]

Removes one or more VMM objects from the scope of objects that members

of this user role can manage.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VMPermission [<SelfServicePermission[] String>]

Specifies the actions that members of a Self Service User role can take

on their virtual machines.

Valid values: Create, PauseAndResume, Start, Stop, AllowLocalAdmin, Rem

oteConnect, Remove, Shutdown, Checkpoint, Store, Save.

Example format: -VMPermission "Create" | "PauseAndResume" | "Stop"

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Add the specified users to the VMM Administrator user role.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $AdminRole = Get-VMMUserRole | where { $\_.Name -eq "Administrator"

}

PS C:\> Set-VMMUserRole -UserRole $AdminRole -AddMember Contoso\User1,Conto

so\User2

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets all objects that represent VMM user roles, selects

the user role object named Administrator, and stores the Administrator obje

ct in variable $AdminRole.

The last command adds User1 and User2 (both members of the Contoso.com doma

in) to the Administrator user role.

2: Add the specified users to the Administrator role in a single command.

PS C:\> Get-VMMUserRole -VMMServer VMMServer1.Contoso.com | where { $\_.Prof

ile -eq "Administrator" } | Set-VMMUserRole -AddMember Contoso\User3

Gets all user role objects from VMMServer1; selects the user role objects w

hose profile is Administrator; and then adds User3 to the Administrator use

r role.

3: Modify an existing user role by adding a host group to its scope.

PS C:\> $HostGroupScope = Get-VMHostGroup -VMMServer VMMServer1.Contoso.com

| where { $\_.Path -eq "All Hosts\Lab" }

PS C:\> $UserRole = Get-VMMUserRole | where { $\_.Name -eq "ContosoAdmin" }

PS C:\> Set-VMMUserRole -UserRole $UserRole -AddScope $HostGroupScope

The first command gets the object that represents the host group named Lab

(a child host group located under "All Hosts") and stores the host group ob

ject in $HostGroupScope.

The second command gets all objects that represent VMM user roles, selects

the user role named ContosoLabAdmin, and stores the object for that user ro

le in $UserRole.

The last command modifies the scope of the "ContosoLabAdmin" user role by a

dding the Lab host group to its scope.

4: Remove the specified user from the Administrator user role.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $AdminRole = Get-VMMUserRole | where {$\_.Name -eq "Administrator" }

PS C:\> Set-VMMUserRole -UserRole $AdminRole -RemoveMember Contoso\User1

The first command connects to VMMServer1.

The second command gets all VMM user role objects, selects the Administrato

r user role object, and stores that object in $AdminRole.

The last command removes User1 (a member of the Contoso.com domain) from th

e Administrator user role.

5: Add multiple templates to a self-service user role.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $Template = Get-Template | where { $\_.Name -like "DevTemplate\*" }

PS C:\> Get-VMMUserRole | where { $\_.Name -like "ContosoSelfServiceUsers" }

| Set-VMMUserRole -AddScope $Template

The first command connects to VMMServer1.

The second command gets all objects that represent templates, selects only

objects for those templates that include "DevTemplate" in the template name

, and stores the selected template objects in $Template.

The last command gets all objects that represent user roles, selects only t

hose that include ContosoSelfServiceUsers" in the name of the user role, an

d then passes each user role object to the Set-VMMUserRole cmdlet. The Set-

VMMUserRole cmdlet adds the templates stored in $Templates to all user role

s that include ContosoSelfServiceUsers in the user role name.

6. Modify what actions members of a self-service user role can take on thei

r virtual machines.

PS C:\> $UserRole = Get-VMMUserRole -VMMServer "VMMServer1.Contoso.com" | w

here { $\_.Name -eq "ContosoSelfServiceUsers" }

PS C:\> Set-VMMUserRole -UserRole $UserRole -VMPermission "Create,PauseAndR

esume,Stop,AllowLocalAdmin,Store" -LibraryStoreSharePath "\\FileServer01.Co

ntoso.com\MSSCVMMLibrary\VHDs"

The first command gets all objects that represents VMM user roles from VMMS

erver1, selects the user role named ContosoSelfServiceUsers, and stores the

user role object in $UserRole.

The second command modifies the permissions for members of the "ContosoSelf

ServiceUsers" user role to allow Creation, PauseAndResume, Stop, AllowLocal

Admin and Store permissions. In addition, the command specifies that member

s of this user role can store a virtual machine (currently deployed on a ho

st) on the library server at the location specified by the -LibraryStoreSha

rePath parameter.

Note: To list all available permissions that you can specify for self-servi

ce users, type:

PS C:\> [enum]::GetValues([Microsoft.VirtualManager.Remoting.SelfServicePer

mission])

The enumeration command lists all possible permissions that you can specify

with the Set-VMMuserRole cmdlet and the -VMPermission parameter:

PERMISSION ALLOWED ACTIONS

---------- ---------------

Create Users can create a new VM

PauseAndResume Users can pause and resume operation of their VMs

Start Users can start their VMs

Stop Users can stop their VMs

AllowLocalAdmin Users can act as local Administrator on their VMs

RemoteConnect Users can access their VMs remotely

Remove Users can remove their VMs and their configuration files

Shutdown Users can shut down their VMs

Checkpoint Users can create checkpoints for their VMs

Store Users can store their VMs in the library

Save Users can put their VMs into a saved state

REMARKS

For more information, type: "get-help Set-VMMUserRole -detailed".

For technical information, type: "get-help Set-VMMUserRole -full".

## VMPerformance

### Get-VMPerformance

SYNOPSIS

Gets performance data for a specific virtual machine deployed on a host man

aged by Virtual Machine Manager.

SYNTAX

Get-VMPerformance [-VM] [<String VM>] [-JobVariable <String>] [-PROTipID <G

uid>] [-RunAsynchronously] [<CommonParameters>]

DETAILED DESCRIPTION

Gets performance data for a specific virtual machine deployed on a host man

aged by Virtual Machine Manager. The VMPerformance object returned by this

cmdlet includes information about CPU utilization history, system uptime, a

nd other data.

PARAMETERS

-VM [<String VM>]

Specifies a virtual machine object.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get performance information about a specific virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VM = Get-VM -Name "VM01"

PS C:\> Get-VMPerformance -VM $VM

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object for VM01 and stores the virtual machine

object in variable $VM.

The last command gets performance data about VM01 and displays it to the us

er. This information includes:

\* The length of time the virtual machine has been running.

\* The CPU utilization history for this virtual machine.

\* The last time this virtual machine was updated.

After this command completes successfully, current information about this v

irtual machine will appear in the Administrator Console.

NOTE: You can use this cmdlet to get performance information about a virtua

l machine if it is currently deployed on a host but not if it is stored in

the VMM library.

REMARKS

For more information, type: "get-help Get-VMPerformance -detailed".

For technical information, type: "get-help Get-VMPerformance -full".

## VMRCCertificateRequest

### New-VMRCCertificateRequest

SYNOPSIS

Generates a request for a signed certificate that Virtual Machine Manager c

an use to secure Virtual Machine Remote Control (VMRC) communications by en

crypting information exchanged between a user and a virtual machine on a Vi

rtual Server host.

SYNTAX

New-VMRCCertificateRequest -KeyLength <Int32> -Path <String> -VMHost [<Stri

ng Host>] [-City <String>] [-CountryRegion <String>] [-JobVariable <String>

] [-Organization <String>] [-OrganizationalUnit <String>] [-PROTipID <Guid>

] [-RunAsynchronously] [-State <String>] [<CommonParameters>]

DETAILED DESCRIPTION

Generates a request for a signed certificate that Virtual Machine Manager c

an use to secure Virtual Machine Remote Control (VMRC) communications by en

crypting information exchanged between a user and a virtual machine on a Vi

rtual Server host. VMRC is a feature of Microsoft Virtual Server that lets

a VMRC client connect to a host running Virtual Server so that a remote use

r can access a virtual machine deployed on that host. This cmdlet applies o

nly to Virtual Server hosts (not Hyper-V or VMware hosts).

Use the file generated by this cmdlet to submit a request to your Certifica

tion Authority (CA). After you receive a certificate, you can use the Set-V

MHost cmdlet with the RemoteConnectCertificatePath parameter to store the c

ertificate on the Virtual Server host. Next, use Set-VMHost with the Secure

RemoteConnectEnabled parameter to enable secure communications for any remo

te user who connects to a virtual machine on that host using VMRC.

The path that you specify with the New-VMRCCertificateRequest cmdlet must i

nclude the name of the file that the cmdlet will create on the client compu

ter (the computer running the Administrator Console or Windows PowerShell).

You must have access permissions that let you create a file on that path o

n the computer. You can specify a local path or a UNC path.

Example paths:

-Path "D:\Requests\MyHostCertificateRequest.txt"

-Path "\\MyServer\MyShare\MyHostCertificateRequest.txt"

PARAMETERS

-KeyLength <Int32>

Specifies the length of the encryption key for an X.509 certificate.

Valid values: 512, 1024, 2048, 4096, 8192

Note: Using 8192 might take longer and therefore might cause the cmdlet

to fail. If so, specify a different length.

-Path <String>

Specifies the destination path for the operation.

Example formats (the specific format or formats you can you use might d

iffer by cmdlet):

Local path -Path "F:\"

UNC path -Path "\\Library\Templates"

Volume GUID path -Path "\\?\Volume{4703c1ea-8ae7-11db-b473-00123f7603e

3}\"

VMware ESX path –Path "[storage1]\MyVMwareFolderForVMs\MyVM.vmx"

Note: See the examples for a specific cmdlet to determine how that cmdl

et specifies the path.

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-City <String>

Specifies the name of the city in which the organization that manages a

specific host is located.

Maximum length: 64 characters.

-CountryRegion <String>

Specifies the name of the country or region in which the organization t

hat manages a specific host is located.

Maximum length: 2 characters.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-Organization <String>

Specifies the name of the organization where a specific host is located

.

Maximum length: 64 characters.

-OrganizationalUnit <String>

Specifies the name of an organizational unit that contains a specific h

ost computer.

Maximum length: 64 characters.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-State <String>

Specifies the name of the state in which the organization that manages

a specific host is located.

Maximum length: 64 characters.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Request a VMRC certificate for a Virtual Server host and save the certif

icate to a text file.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $VMHost = Get-VMHost -ComputerName "VMHost01"

PS C:\> New-VMRCCertificateRequest -VMHost $VMHost -Path "C:\VMRCCertReqFol

der\CertificateRequest.txt" -KeyLength 4096 -Organization "Contoso" -Organi

zationalUnit "MyDepartment" -City "MyCity" -State "MyState" -CountryRegion

"US"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents VMHost01 from the VMM da

tabase and stores the host object in variable $VMHost. This example assumes

that VMHost01 is a Virtual Server host.

The last command creates a certificate request for VMHost01; specifies a ke

y length; specifies values for Organization and other identifying fields; a

nd then saves the resulting request as the CertificateRequest.txt file stor

ed in C:\VMRCCertReqFolder on VMHost01.

REMARKS

For more information, type: "get-help New-VMRCCertificateRequest -detailed".

For technical information, type: "get-help New-VMRCCertificateRequest -full".

## VMwareResourcePool

### Get-VMwareResourcePool

SYNOPSIS

Gets VMware resource pool objects from the Virtual Machine Manager database.

SYNTAX

Get-VMwareResourcePool -VMHostCluster <VMHostCluster> [<CommonParameters>]

Get-VMwareResourcePool -VMHost [<String Host>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets VMware resource pool objects from the Virtual Machine Manager database.

VMware resource pools configured on a VMwareVirtual Center server that is m

anaged by Virtual Machine Manager are imported as objects into the Virtual

Machine Manager database when the VirtualCenter Server is added to Virtual

Machine Manager.

VMware uses resource pools to group virtual machines deployed on ESX hosts,

or ESX host clusters, into an organizational hierarchy that consists of pa

rent, sibling, and child resource pools. Resources, such as CPU and memory,

are specified for virtual machines assigned to each resource pool. Adminis

tration of sets of resource pools can be delegated, in VirtualCenter Server

, to administrators by department, by geographical region, or by some other

organizational requirement.

PARAMETERS

-VMHost [<String Host>]

Specifies a virtual machine host object. VMM 2008 supports Hyper-V host

s, Virtual Server hosts, and VMware ESX Server hosts. For more informat

ion about each type of host, type: Get-Help Add-VMHost -detailed. See t

he examples for a specific cmdlet to determine how that cmdlet uses thi

s parameter.

-VMHostCluster <VMHostCluster>

Specifies a VMM host cluster object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get the VMware resource pool for an ESX host from the VMM database.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MyESXHost = Get-VMHost -ComputerName "nnn.nnn.nnn.nnn"

PS C:\> Get-VMwareResourcePool -VMHost $MyESXHost

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following commands use this s

erver by default.

The second command gets the object that represents the VMware ESX host name

d VMHost01 from the VMM database by specifying the IP address (represented

in the example by "nnn.nnn.nnn.nnn") of the ESX host. This ESX Server is ma

naged by VMM through VMware VirtualCenter Server. The command stores the ho

st object in variable $MyESXHost.

NOTE: When you use an IP address with the -ComputerName parameter to specif

y an ESX host, you must use the same IP address that was used initially to

add this host to VMM.

The last command gets the VMware resource pool information from the VMM dat

abase for the ESX host object stored in variable $VMHost and displays infor

mation about the resource pool to the user.

2: Get the VMware resource pool for an ESX host cluster from the VMM databa

se.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $MyESXCluster = Get-VMHostCluster -Name "VMHostCluster02"

PS C:\> Get-VMwareResourcePool -VMHostCluster $MyESXCluster

The first command connects to VMMServer1.

The second command gets the object that represents the host cluster named V

MHostCluster01 from the VMM database. This is a cluster of ESX hosts that i

s managed by VMM through VMware VirtualCenter Server. The command stores th

e host cluster object in $MyESXCluster.

The third command gets the VMware resource pool information from the VMM da

tabase for the ESX host cluster object stored in $MyESXCluster and displays

information about the resource pool to the user.

REMARKS

For more information, type: "get-help Get-VMwareResourcePool -detailed".

For technical information, type: "get-help Get-VMwareResourcePool -full".

## VMXMachineConfig

### Get-VMXMachineConfig

SYNOPSIS

Gets VMX machine configuration objects (from the Virtual Machine Manager da

tabase) that are associated with one or more VMware-based virtual machines.

SYNTAX

Get-VMXMachineConfig [-VMMServer [<String ServerConnection>]] [-VMXPath <St

ring>] [<CommonParameters>]

DETAILED DESCRIPTION

Gets one or more objects from the Virtual Machine Manager database that rep

resent the VMX machine configuration associated with one or more VMware vir

tual machines. Information about a virtual machine computer's hardware, dis

ks, and operating system is stored in the VMX machine configuration object.

A VMX machine configuration object is used by the New-V2V cmdlet when it co

nverts a VMware-based virtual machine deployed on an ESX host to a virtual

machine deployed on a Windows-based host (a Virtual Server or Hyper-V host).

PARAMETERS

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

-VMXPath <String>

Specifies the full UNC path to the .vmx file of a VMware virtual machin

e.

Example format: \\ServerName\VolumeName\DirectoryName\VMwareVM.vmx

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Get all machine configurations objects in your VMM environment.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> Get-VMXMachineConfig

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets all objects from the VMM database that represent VM

X machine configurations (which were created earlier by using New-VMXMachin

eConfig) and displays information about these machine configuration objects

to the user.

REMARKS

For more information, type: "get-help Get-VMXMachineConfig -detailed".

For technical information, type: "get-help Get-VMXMachineConfig -full".

### New-VMXMachineConfig

SYNOPSIS

Creates a VMX machine configuration object by gathering virtual machine con

figuration information from a virtual machine created in VMware that you pl

an to convert to a virtual machine deployed on a Windows-based host managed

by Virtual Machine Manager.

SYNTAX

New-VMXMachineConfig [-VMXPath] <String> [-JobVariable <String>] [-LibraryS

erver [<String LibraryServer>]] [-PROTipID <Guid>] [-RunAsynchronously] [-V

MMServer [<String ServerConnection>]] [<CommonParameters>]

DETAILED DESCRIPTION

Creates a VMX machine configuration object by gathering information about t

he physical characteristics of a VMware-based virtual machine (and its disk

s) that you plan to convert to a virtual machine deployed on a Windows-base

d host (a Hyper-V or Virtual Server host) managed by Virtual Machine Manage

r. This cmdlet does not collect information about the operating system or d

ata on the VMware-based virtual machine.

VMWare virtual hard disk formats supported by the New-VMXMachineConfig cmdl

et include:

- monolithicSparse

- monolithicFlat

- vmfs

- twoGbMaxExtentSparse

- twoGbMaxExtentFlat

PARAMETERS

-VMXPath <String>

Specifies the full UNC path to the .vmx file of a VMware virtual machin

e.

Example format: \\ServerName\VolumeName\DirectoryName\VMwareVM.vmx

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-LibraryServer [<String LibraryServer>]

Specifies a VMM library server object.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

-VMMServer [<String ServerConnection>]

Specifies a VMM server object.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Gather information from a VMware-based virtual machine.

PS C:\> Get-VMMServer -ComputerName "VMMServer1.Contoso.com"

PS C:\> $LibServ = Get-LibraryServer –ComputerName “FileServer01.Contoso.co

m”

PS C:\> New-VMXMachineConfig –LibraryServer $LibServ –VMXPath

"\\FileServer01\MSSCVMMLibrary\VMware\VMSource.vmx"

The first command connects to VMMServer1 in the Contoso.com domain and gets

the server object from the VMM database. The following command uses this s

erver by default.

The second command gets the object that represents the library server FileS

erver01 (in the Contoso.com domain) and stores the library object in variab

le $LibServ.

The last command gathers the machine configuration information for the .vmx

file located at "\\FileServer01\MSSCVMMLibrary\VMware\VMSource.vmx" on the

library server. The New-VMXMachineConfig cmdlet stores the resulting machi

ne configuration object associated with VMSource.vmx in the VMM database.

NOTE: If you look in Library view in the Administrator Console, you cannot

see the file VMSource.vmx because the .vmx file is part of a single virtual

machine object. What you see in Library view is the virtual machine. To fi

nd the path to a .vmdk file, view the properties for that virtual machine.

REMARKS

For more information, type: "get-help New-VMXMachineConfig -detailed".

For technical information, type: "get-help New-VMXMachineConfig -full".

### Remove-VMXMachineConfig

SYNOPSIS

Removes a VMX machine configuration object from Virtual Machine Manager.

SYNTAX

Remove-VMXMachineConfig [-VMXMachineConfig] <VMXMachineConfig> [-Confirm] [

-JobVariable <String>] [-PROTipID <Guid>] [-RunAsynchronously] [<CommonPara

meters>]

DETAILED DESCRIPTION

Removes one or more objects that represent a VMX machine configuration from

the Virtual Machine Manager database.

This cmdlet returns the object upon success (with the property MarkedForDel

etion set to TRUE) or returns an error message upon failure.

PARAMETERS

-VMXMachineConfig <VMXMachineConfig>

Specifies a VMX machine configuration for a VMware-based virtual machin

e. VMX machine configuration includes information about the virtual mac

hine's hardware, disks, and operating system.

Note: In VMM 2007, this parameter, when used with the New-V2V and Remov

e-VMXMachineConfig cmdlets, was named MachineConfig.

-Confirm

Prompts for confirmation before running the command.

-JobVariable <String>

Specifies that job progress is tracked and stored in the variable named

by this parameter.

-PROTipID <Guid>

Specifies the ID of the PRO tip that triggered this action. Allows for

future auditing of PRO tips.

-RunAsynchronously

Specifies that the job runs asynchronously so that control returns to t

he command shell immediately.

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug,

-ErrorAction, -ErrorVariable, and -OutVariable. For more information,

type, "get-help about\_commonparameters".

1: Remove all VMX machine configurations without being prompted to confirm

each deletion.

PS C:\> $VMXMachineConfigs = Get-VMXMachineConfig -VMMServer VMMServer1.Con

toso.com

PS C:\> $MachineConfigs | Remove-VMXMachineConfig

The first command retrieves all VMX machine configuration objects from the

VMM database on VMMServer1 and stores these objects in variable $VMXMachine

Configs (an object array).

The second command passes each object in $VMXMachineConfigs to Remove-VMXMa

chineConfig, which removes each VMX machine configuration object.

REMARKS

For more information, type: "get-help Remove-VMXMachineConfig -detailed".

For technical information, type: "get-help Remove-VMXMachineConfig -full".