MVMC – Virtual Machine Conversion Cmdlets

ConvertTo-MvmcAzureVirtualHardDisk

ConvertTo-MvmcAzureVirtualHardDisk

Converts VMware virtual disks to VHDs and uploads them to Windows Azure.

Syntax

```
Parameter Set: Default
ConvertTo-MvmcAzureVirtualHardDisk [-SourceConnection] <MvmcSourceConnection> [-
SubscriptionId] <String> [-Thumbprint] <String> [-StorageAccount] <String> [-GuestVmId]
<String> [[-GuestCredential] <PSCredential> ] [[-UninstallVMTools]] [[-SourceVMPowerOption]
<PowerOption> ] [ <CommonParameters>]
```

Detailed Description

The **ConvertTo-MvmcAzureVirtualHardDisk** cmdlet converts disks attached to a VMware virtual machine to one or more virtual hard disks (VHDs) and uploads them to Windows Azure. You must have a Windows Azure subscription ID, certificate thumbprint, and storage account. A subscription ID uniquely identities your Windows Azure subscription. Windows Azure uses an X.509 v3 certificate to authenticate operations. This cmdlet requires the thumbprint of the certificate.

You need a management certificate for Windows Azure to authenticate your subscription ID on Windows Azure. The certificate needs to be present in Personal store and Trusted store of the current user. For more information, see Create and Upload a Management Certificate for Windows Azure on the Microsoft Developer Network.

Create a storage account in Windows Azure before you begin. For more information, see How To Create a Storage Account on the Windows Azure site.

If you specify the *UninstallVMTools* parameter, this cmdlet removes VMware Tools from the source virtual machine before it converts the virtual machine to a VHD. In order to uninstall VMware Tools, you must provide appropriate credentials. You cannot uninstall VMware Tools from an offline source virtual machine. After this cmdlet copies disks to the computer where it converts them, it restores the source virtual machine to the snapshot, which includes VMware Tools, if present previously. You can then turn on the source virtual machine, if necessary.

Parameters

-GuestCredential<PSCredential>

Specifies credentials to establish a connection to the source virtual machine. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`.

Aliases	none
Required?	false
Position?	6
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

-GuestVmId<String>

Specifies the ID for a virtual machine. To obtain an object that contains this ID, use the **Get-MvmcSourceVirtualMachine** cmdlet.

Aliases	none
Required?	true
Position?	5
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceConnection<MvmcSourceConnection>

Specifies a connection to a VMware vSphere host or computer that runs VMware vCenter Server. To obtain a connection, use the **New-MvmcSourceConnection** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none

Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-SourceVMPowerOption<PowerOption>

Specifies a power option for the source virtual machine. Valid values are: PowerOff and PowerOn. If the source virtual machine is currently off, the PowerOn option does not cause it to turn on.

Do not specify a value of Shutdown for this parameter.

Aliases	none
Required?	false
Position?	8
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

-StorageAccount<String>

Specifies a Windows Azure storage account. This cmdlet stores VHDs in this storage account.

Aliases	none
Required?	true
Position?	4
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SubscriptionId<String>

Specifies a subscription ID for Windows Azure.

Aliases	none
Required?	true

Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-Thumbprint<String>

Specifies a certificate thumbprint for Windows Azure.

Aliases	none
Required?	true
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-UninstallVMTools

Indicates that this cmdlet unistalls VMware Tools from the source virtual machine. If you specify this parameter, you must specify appropriate credentials for the *GuestCredential* parameter. If the source virtual machine is offline, the cmdlet ignores this parameter.

Aliases	none
Required?	false
Position?	7
Default Value	none
Accept Pipeline Input?	false
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Engine.ServiceLayer.IMachineDriveCollection**

Examples

Example 1: Convert disks to VHD and upload to Windows Azure

This example converts the disks attached to a VMware virtual machine to VHDs and uploads them to the specified Storage account in Windows Azure.

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores that secure string in the $SourcePassword variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $SourcePassword, and then stores the credentials in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and then stores that connection object in the $SourceConnection variable.

The fifth command gets all of the virtual machines from the connection object stored in $SourceConnection, and then passes them to a **Where-Object** cmdlet by using the pipeline operator. That cmdlet drops all virtual machines except one named VM073. The command stores any virtual machine named VM073 in the $SourceVM variable. For more information, type `Get-Help Where-Object`.

The purpose of the fifth command is to get a virtual machine object that contains the ID of the guest virtual machine as a member.

The sixth command stores the Windows Azure subscription ID in the $SubscriptionID variable.

The seventh command stores the Windows Azure thumbprint string in the $ThumbPrint variable.

The final command converts the virtual machine to VHDs and uploads them to a storage account on Windows Azure. The command specifies the connection stored in $SourceConnection. The command uses standard dot notation to refer to the **GuestVmId** member of the object stored in $SourceVM, which identifies the virtual machine. The command specifies the Windows Azure subscription ID and thumbprint, and specifies the storage account as ContosoStore07.

```
PS C:\> $SourceUser = "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object -TypeName System.Management.Automation.PSCredential
-ArgumentList $SourceUser,$SourcePassword
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> $SourceVM = Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection |
Where-Object {$_.Name -match 'VM073'}
 PS C:\> $SubscriptionID = "<subscriptionID>"
```

```
 PS C:\> $ThumbPrint = "<thumbprint>"
 PS C:\> ConvertTo-MvmcAzureVirtualHardDisk -SourceConnection $SourceConnection -
SubscriptionId $SubscriptionID -Thumbprint $ThumbPrint -StorageAccount "ContosoStore07" -
GuestVmId $SourceVM.GuestVmId
```

Related topics

New-MvmcSourceConnection

Get-MvmcSourceVirtualMachine

ConvertTo-MvmcVirtualHardDisk

ConvertTo-MvmcVirtualHardDiskOvf

Disable-MvmcSourceVMTools

Uninstall-MvmcSourceVMTools

New-MvmcVirtualMachineFromOvf

Stop-MvmcSourceVirtualMachine

ConvertTo-MvmcVirtualHardDisk

ConvertTo-MvmcVirtualHardDisk

Converts a VMDK to a VHD.

Syntax

```
Parameter Set: Default
ConvertTo-MvmcVirtualHardDisk [-SourceLiteralPath] <String> [[-DestinationLiteralPath]
<String> ] [[-VhdType] <VhdType> ] [[-VhdFormat] <VhdFormat> ] [ <CommonParameters>]
```

Detailed Description

The **ConvertTo-MvmcVirtualHardDisk** cmdlet converts a VMware virtual disk (VMDK) to a Hyper-V™ based virtual hard disk (VHD). The source virtual disk must be on the local file system of the computer where you run this cmdlet.

Use the *VhdFormat* parameter to specify a VHD format. There are two VHD formats: VHD and VHDx. Windows Server® 2008 R2 does not support the VHDx format. If you run this cmdlet on a computer that runs Windows Server 2008 R2, it creates disks in the VHD format. If you run this cmdlet on a computer that runs Windows® 8, Windows Server® 2012, or Windows Server® 2012 R2, this cmdlet creates disks in the VHDx format, by default.

Parameters

-DestinationLiteralPath<String>

Specifies a literal path. This cmdlet stores the converted VHD in the folder that you specify. If you do not specify a directory, the cmdlet saves to the current directory.

Aliases	VhdPath
Required?	false
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceLiteralPath<String>

Specifies a literal path. The cmdlet converts the VMDK that the path specifies.

Aliases	Src
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-VhdFormat<VhdFormat>

Specifies the file format for a VHD. Valid values are: Vhd and Vhdx.

Aliases	none
Required?	false
Position?	4
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

-VhdType<VhdType>

Specifies the type for a VHD. Valid values are: FixedHardDisk and DynamicHardDisk. The default value is DynamicHardDisk.

Aliases	none
Required?	false
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Cmdlet.ConvertedDisk**

Examples

Example 1: Convert a VMDK to a VHD by using default values

This command converts the specified VMDK into a VHD. The command does not specify a destination for the VHD, so the command stores it in the current directory.

Because the command does not specify the *VhdFormat* parameter value, the format depends on the version of the Windows operating system. If you run the command on a computer that runs Windows 8, Windows Server 2012, or Windows Server® 2012 R2, it creates file in the VHDx format. If you run the command on a computer that runs Windows Server 2008 R2, it creates a file in the VHD format.

The command does not specify a value for the *VhdType* parameter, and, therefore, the command creates the default dynamic hard disk.

```
PS C:\> ConvertTo-VirtualHardDisk -SourceLiteralPath "C:\VMDKs\PattiFullerVMDK.vmdk"
```

Example 2: Convert a VMDK to a fixed VHD in VHD format

This command converts a VMDK named PattiFullerVMDK.vmdk in the specified directory to a VHD in the C:\VHDs directory. This example specifies a value of FixedHardDisk for the *VhdType* parameter, and specifies the VHD format.

```
PS C:\> ConvertTo-VirtualHardDisk -SourceLiteralPath "C:\VMDKs\PattiFullerVMDK.vmdk"-
DestinationLiteralPath "C:\VHDs" -VhdType FixedHardDisk -VhdFormat Vhd
```

Related topics

ConvertTo-MvmcAzureVirtualHardDisk

ConvertTo-MvmcVirtualHardDiskOvf

ConvertTo-MvmcVirtualHardDiskOvf

ConvertTo-MvmcVirtualHardDiskOvf

Converts VMware virtual disks to VHDs and exports configuration information as an OVF file.

Syntax

```
Parameter Set: Default
ConvertTo-MvmcVirtualHardDiskOvf [-SourceConnection] <MvmcSourceConnection> [[-
DestinationLiteralPath] <String> ] [-GuestVmId] <String> [[-GuestCredential] <PSCredential>
] [[-VhdType] <VhdType> ] [[-VhdFormat] <VhdFormat> ] [[-UninstallVMTools]] [[-
SourceVMPowerOption] <PowerOption> ] [ <CommonParameters>]
```

Detailed Description

The **Convertto-MvmcVirtualHardDiskOvf** cmdlet converts virtual disks attached to a VMware virtual machine to Hyper-V™ based virtual hard disks (VHDs) and exports the configuration of the virtual machine as an open virtualization format (OVF) file. This cmdlet creates a snapshot of a virtual machine and then shuts down the virtual machine. After the cmdlet copies disks to the computer where it converts them, the cmdlet restores the source virtual machine to the snapshot. You can then turn the source virtual machine on, if necessary.

Use the *VhdFormat* parameter to specify a VHD format. There are two VHD formats: VHD and VHDx. Windows Server® 2008 R2 does not support the VHDx format. If you run this cmdlet on a computer that runs Windows Server 2008 R2, it creates disks in the VHD format. If you run this cmdlet on a computer that runs Windows® 8, Windows Server® 2012, or Windows Server® 2012 R2, this cmdlet creates disks in the VHDx format, by default.

If you specify the *UninstallVMTools* parameter, this cmdlet removes VMware Tools from the source virtual machine before it converts the virtual machine to a VHD. In order to uninstall VMware Tools, you must provide credentials to connect to the virtual machine as administrator or root. You cannot uninstall VMware Tools from an offline source virtual machine. After the cmdlet copies disks to the computer where it converts them, it restores the source virtual machine to the snapshot, which includes VMware Tools, if present previously.

Parameters

-DestinationLiteralPath<String>

Specifies a literal path. This cmdlet stores the converted VHD or VHDs and exported OVF file in the folder that the path specifies.

Aliases	VhdPath
Required?	false
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-GuestCredential<PSCredential>

Specifies credentials to establish a connection to the source virtual machine. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`. If you specify the *UninstallVMTools* parameter, you must specify credentials for this parameter.

Aliases	none
Required?	false
Position?	4
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

-GuestVmId<String>

Specifies the ID for a virtual machine. To obtain an object that contains this ID, use the **Get-MvmcSourceVirtualMachine** cmdlet.

Aliases	none
Required?	true
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceConnection<MvmcSourceConnection>

Specifies a connection to a VMware vSphere host or computer that runs VMware vCenter Server. To obtain a connection, use the **New-MvmcSourceConnection** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-SourceVMPowerOption<PowerOption>

Specifies a power option for the source virtual machine. Valid values are: PowerOff and PowerOn. If the source virtual machine is currently off, the PowerOn option does not cause it to turn on.

Do not specify a value of Shutdown for this parameter.

Aliases	none
Required?	false
Position?	8
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

-UninstallVMTools

Indicates that this cmdlet unistalls VMware Tools from the source virtual machine. If you specify this parameter, you must specify appropriate credentials for the **GuestCredential** parameter. If the source virtual machine is offline, the cmdlet ignores this parameter.

Aliases	none
Required?	false
Position?	7
Default Value	none

Accept Pipeline Input?	false
Accept Wildcard Characters?	false

-VhdFormat<VhdFormat>

Specifies the file format for a VHD. Valid values are: Vhd and Vhdx.

Aliases	none
Required?	false
Position?	6
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

-VhdType<VhdType>

Specifies the type for a VHD. Valid values are: FixedHardDisk and DynamicHardDisk. The default value is DynamicHardDisk.

Aliases	none
Required?	false
Position?	5
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Engine.ServiceLayer.IMachineDriveCollection**

Examples

Example 1: Convert a source virtual machine to VHDs and an OVF file

This example converts a source virtual machine named VM073 to one or more VHDs and an OVF file in the directory C:\VHDs\PattiFullerVM.

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores that secure string in the $SourcePassword variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $SourcePassword, and stores the credentials in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and then stores that connection object in the $SourceConnection variable.

The fifth command gets all of the virtual machines from the connection object stored in $SourceConnection variable, and passes them to the **Where-Object** cmdlet. That cmdlet drops all virtual machines except one named VM073. The command stores any virtual machine named VM073 in the $SourceVM variable. For more information, type `Get-Help Where-Object`.

The final command converts the source virtual machine The command specifies the connection stored in $SourceConnection. The command uses standard dot notation to refer to the **GuestVmId** member of the object stored in $SourceVM. This ID specifies the source virtual machine. The command saves the VHD or VHDs and the OVF file in the directory C:\VHDs\PattiFullerVM.

```
PS C:\> $SourceUser = "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object -TypeName System.Management.Automation.PSCredential
-ArgumentList $SourceUser,$SourcePassword
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> $SourceVM = Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection |
Where-Object {$_.Name -match 'VM073'}
 PS C:\> Convertto-MvmcVirtualHardDiskOvf -SourceConnection $SourceConnection -
DestinationLiteralPath "C:\VHDs\PattiFullerVM" -GuestVmId $SourceVM.GuestVmId
```

Related topics

New-MvmcSourceConnection

Get-MvmcSourceVirtualMachine

ConvertTo-MvmcAzureVirtualHardDisk

ConvertTo-MvmcVirtualHardDisk

Disable-MvmcSourceVMTools
Uninstall-MvmcSourceVMTools
New-MvmcVirtualMachineFromOvf
Stop-MvmcSourceVirtualMachine

Disable-MvmcSourceVMTools

Disable-MvmcSourceVMTools

Disables VMware Tools for a VHD.

Syntax

```
Parameter Set: Default
Disable-MvmcSourceVMTools [-DestinationLiteralPath] <String> [ <CommonParameters>]
```

Detailed Description

The **Disable-MvmcSourceVMTools** cmdlet disables services, drivers, and programs associated with VMware Tools. You may need to disable these items for a virtual hard disk (VHD) that was converted from an offline VMware virtual machine disk (VMDK).

This cmdlet supports only VHDs that employ the Windows operating system.

Parameters

-DestinationLiteralPath<String>

Specifies a literal path. This cmdlet disables VMware Tools for the VHD or VHDx that the path specifies.

Aliases	VhdPath
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Examples

Example 1: Disable VMware Tools

This command disables services, drivers and programs associated with VMware Tools for the specified VHD.

```
PS C:\> Disable-MvmcSourceVMTools -DestinationLiteralPath "C:\VHD\VHD073.vhd"
```

Related topics

Uninstall-MvmcSourceVMTools

Get-MvmcSourceVirtualMachine

Get-MvmcSourceVirtualMachine

Gets virtual machines from a VMware server.

Syntax

```
Parameter Set: Default
Get-MvmcSourceVirtualMachine [-SourceConnection] <MvmcSourceConnection> [
<CommonParameters>]
```

Detailed Description

The **Get-MvmcSourceVirtualMachine** cmdlet gets virtual machines from a VMware vSphere host or a computer that runs VMware vCenter Server. If you get virtual machines from vCenter Server, this cmdlet gets the virtual machines from all hosts that the server manages.

Parameters

-SourceConnection<MvmcSourceConnection>

Specifies a connection to a vSphere host or computer that runs vCenter Server. To obtain connection, use the **New-MvmcSourceConnection** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Engine.ForVMWare.LightVirtualMachine**

Examples

Example 1: Get a virtual machine from a VMware host

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores the secure string in the $Password variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $Password, and then stores the credentials in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and then stores that connection object in the $SourceConnection variable.

The final command gets all of the virtual machines from the the connection object stored in $SourceConnection, and the passes them to the **Where-Object** cmdlet by using the pipeline operator. That cmdlet drops all virtual machines except one that has the name VM073. For more information, type `Get-Help Where-Object`.

```
PS C:\> $SourceUser= "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object -TypeName System.Management.Automation.PSCredential
-ArgumentList $SourceUser,$SourcePassword
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection | Where-Object
{$_.Name -match 'VM073'}
```

Related topics

New-MvmcSourceConnection

Stop-MvmcSourceVirtualMachine

New-MvmcSourceConnection

New-MvmcSourceConnection

Establishes a connection to a VMware server.

Syntax

```
Parameter Set: Default
New-MvmcSourceConnection [-Server] <String> [-SourceCredential] <PSCredential> [
<CommonParameters>]
```

Detailed Description

The **New-MvmcSourceConnection** cmdlet establishes a connection to a VMware vSphere host or computer that runs VMware vCenter Server. A virtual machine source hosts ESX or ESXi virtual machines. This cmdlet creates connections to the following sources:

-- vCenter Server 5.0

-- vCenter Server 4.1

-- vCenter 5.1

-- vCenter 4.0

-- ESXi Server 5.5 (vServer)

-- ESXi Server 5.1 (vServer)

-- ESXi Server 5.0 (vServer)

-- ESXi/ESX Server 4.1 (vServer)

-- ESXi/ESX Server 4.0 (vServer)

Parameters

-Server<String>

Specifies the name or IP address of vSphere host or computer that runs vCenter Server. Alternatively, specify ESX host name.

Aliases	none
Required?	true
Position?	1

Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceCredential<PSCredential>

Specifies an object that contains credentials to establish a connection to a vSphere host or computer that runs vCenter Server. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`.

Aliases	none
Required?	true
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Cmdlet.MvmcSourceConnection**

Examples

Example 1: Establish a connection to a VMware server

The first command stores the string ContosoAdmin in the $UserName variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores the secure string in the $Password variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $UserName and $Password, and then stores the credentials in the $Credential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to

create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The final command creates a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $Credential.

```
PS C:\> $UserName = "ContosoAdmin"
 PS C:\> $Password = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $Credential = New-Object -Typename System.Management.Automation.PSCredential -
ArgumentList $UserName,$Password
 PS C:\> New-MvmcSourceConnection -Server "ContosoVS03" -SourceCredential $Credential
```

Related topics

Get-MvmcSourceVirtualMachine

New-MvmcSourceVirtualMachineSnapshot

New-MvmcSourceVirtualMachineSnapshot

Creates a snapshot of a source virtual machine.

Syntax

```
Parameter Set: Default
New-MvmcSourceVirtualMachineSnapshot [-SourceConnection] <MvmcSourceConnection> [-GuestVmId]
<String> [-SnapshotName] <String> [ <CommonParameters>]
```

Detailed Description

The **New-MvmcSourceVirtualMachineSnapshot** cmdlet creates a snapshot of source virtual machine. Specify a connection to a VMware vSphere host or a computer that runs VMware vCenter Server.

Parameters

-GuestVmId<String>

Specifies the ID for a virtual machine. To obtain an object that contains this ID, use the **Get-MvmcSourceVirtualMachine** cmdlet.

Aliases	none
Required?	true
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SnapshotName<String>

Specifies a name for the snapshot.

Aliases	none
Required?	true
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceConnection<MvmcSourceConnection>

Specifies a connection to a vSphere host or computer that runs vCenter Server. To obtain a connection, use the **New-MvmcSourceConnection** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Engine.ForVMware.Snapshot**

Examples

Example 1: Create a snapshot of a VMware virtual machine

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores the secure string in the $SourcePassword variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $SourcePassword, and stores it in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and then stores that connection object in the $SourceConnection variable.

The fifth command gets all of the virtual machines from the connection object stored in $SourceConnection, and then passes them to the **Where-Object** cmdlet by using the pipeline operator. This cmdlet drops all virtual machines except one named VM073. The command stores any virtual machine named VM073 in the **$SourceVM** variable. For more information, type `Get-Help Where-Object`.

The final command creates a snapshot for a virtual machine. The command specifies the connection stored in $SourceConnection. The command uses standard dot notation to refer to the **GuestVmId** member of the object stored in the $SourceVM variable. This ID identifies the virtual machine. The command provides the name 201302Snapshot for the snapshot.

```
PS C:\> $SourceUser = "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object -TypeName System.Management.Automation.PSCredential
-ArgumentList $SourceUser,$SourcePassword
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> $SourceVM = Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection |
Where-Object {$_.Name -match 'VM073'}
 PS C:\> New-MvmcSourceVirtualMachineSnapshot -SourceConnection $SourceConnection -GuestVmId
$sourceVM.GuestVmId -SnapshotName "201302Snapshot"
```

Related topics

New-MvmcSourceConnection

Get-MvmcSourceVirtualMachine

Restore-MvmcSourceVirtualMachineSnapshot

New-MvmcVirtualMachineFromOvf

New-MvmcVirtualMachineFromOvf

Creates a virtual machine from an OVF configuration file.

Syntax

```
Parameter Set: Default
New-MvmcVirtualMachineFromOvf [-DestinationLiteralPath] <String> [[-DestinationServer]
<String> ] [[-DestinationServerCredential] <PSCredential> ] [[-DestinationVMPowerOption]
<PowerOption> ] [[-VMConfigurationLiteralPath] <String> ] [[-SnapshotLiteralPath] <String> ]
[ <CommonParameters>]
```

Detailed Description

The **New-MvmcVirtualMachineFromOvf** cmdlet creates a Hyper-V™ virtual machine from an open virtualization format (OVF) configuration file and attaches the converted virtual hard disks (VHD) to the new virtual machine.

Parameters

-DestinationLiteralPath<String>

Specifies a literal path. This cmdlet stores the OVF file and VHDs for the virtual machine that the path specifies.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-DestinationServer<String>

Specifies a fully qualified domain name (FQDN), server name, or IP address of the destination server that runs Hyper-V. If you do not specify a server, the cmdlet uses the local server.

Aliases	none
Required?	false
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-DestinationServerCredential<PSCredential>

Specifies an object that contains credentials to establish a connection to the server that runs Hyper-V. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`. If you do not specify credentials, the cmdlet uses network credentials.

Aliases	none
Required?	false
Position?	5
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-DestinationVMPowerOption<PowerOption>

Specifies which power option to use for the virtual machine. Valid values are: PowerOff and PowerOn. The default value is PowerOff.

Do not specify a value of Shutdown for this parameter.

Aliases	none
Required?	false
Position?	7
Default Value	none

Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SnapshotLiteralPath<String>

Specifies a path to store snapshots for the virtual machine. If you do not specify a value, the cmdlet uses the Hyper-V default path. This parameter is valid only when the Hyper-V runs on a server that runs Windows Server® 2012.

Aliases	none
Required?	false
Position?	11
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-VMConfigurationLiteralPath<String>

Specifies a path for virtual machine configuration for the virtual machine. If you do not specify a path, the cmdlet uses the default location specified on the server that runs Hyper-V.

Aliases	none
Required?	false
Position?	9
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **Microsoft.Accelerators.Mvmc.Engine.HyperV.IHyperVVirtualMachine**

Examples

Example 1: Create a virtual machine on the local server

This command creates a virtual machine on the local server that runs Hyper-V for the OVF file and VHD or VHDs in the folder C:\OVF07.

```
PS C:\> New-MvmcVirtualMachineFromOvf -DestinationLiteralPath "C:\OVF07"
```

Example 2: Create a virtual machine on a specified server

This command creates a virtual machine on the server named ContosoVMServer for the OVF file and VHD or VHDs in the shared folder \\ContosoVMServer\OVF21.

```
PS C:\> New-MvmcVirtualMachineFromOvf -DestinationLiteralPath "\\ContosoVMServer\OVF21" -
DestinationServer "ContosoVMServer"
```

Related topics

ConvertTo-MvmcVirtualHardDiskOvf

Stop-MvmcSourceVirtualMachine

Restore-MvmcSourceVirtualMachineSnapshot

Restore-MvmcSourceVirtualMachineSnapshot

Restores the source virtual machine from a snapshot.

Syntax

```
Parameter Set: Default
Restore-MvmcSourceVirtualMachineSnapshot [-Snapshot] <ISnapshot> [[-SkipDelete]] [
<CommonParameters>]
```

Detailed Description

The **Restore-MvmcSourceVirtualMachineSnapshot** cmdlet restores a source virtual machine to a snapshot. Specify a connection to a VMware vSphere server or a server that runs VMware vCenter Server. The cmdlet deletes the snapshot, unless you include the *SkipDelete* parameter.

Parameters

-SkipDelete

Indicates that this cmdlet does not delete the snapshot after it restores the virtual machine.

Aliases	none
Required?	false
Position?	2
Default Value	none
Accept Pipeline Input?	false
Accept Wildcard Characters?	false

-Snapshot<ISnapshot>

Specifies a snapshot object. To obtain a snapshot object, use the **New-MvmcSourceVirtualMachineSnapshot** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Examples

Example 1: Restore a virtual machine to a snapshot

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores the secure string in the $SourcePassword variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $SourcePassword, and then stores the credentials in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and then stores that connection object in the $SourceConnection variable.

The fifth command gets all of the virtual machines from the server specified in the connection object stored in $SourceConnection, and then passes them to a **Where-Object** filter by using the pipeline operator. That cmdlet drops all virtual machines except one named VM073. The command stores any virtual machine named VM073 in the $SourceVM variable. For more information, type `Get-Help Where-Object`.

The sixth command creates a snapshot for a virtual machine, and then stores the snapshot in the $Snapshot variable. The command specifies the connection stored in $SourceConnection. The command uses standard dot notation to refer to the **GuestVmId** member of the object stored in $SourceVM. This ID specifies the virtual machine. The command specifies the name 201302Snapshot for the snapshot.

The final command restores the snapshot specified by the object sorted in $Snapshot. This command does not use the *SkipDelete* parameter, and, therefore, the command deletes the snapshot when it finishes.

```
PS C:\> $SourceUser = "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object -TypeName System.Management.Automation.PSCredential
-ArgumentList $SourceUser,$SourcePassword
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> $SourceVM = Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection |
Where-Object {$_.Name -match 'VM073'}
 PS C:\> $Snapshot = New-MvmcSourceVirtualMachineSnapshot -SourceConnection
$SourceConnection -GuestVmId $SourceVM.GuestVmId -SnapshotName "201302Snapshot"
 PS C:\> Restore-MvmcSourceVirtualMachineSnapshot -Snapshot $Snapshot
```

Related topics

New-MvmcSourceConnection

Get-MvmcSourceVirtualMachine

New-MvmcSourceVirtualMachineSnapshot

Stop-MvmcSourceVirtualMachine

Stop-MvmcSourceVirtualMachine

Shuts down or powers off a source virtual machine.

Syntax

```
Parameter Set: Default
Stop-MvmcSourceVirtualMachine [-SourceConnection] <MvmcSourceConnection> [-GuestVmId]
<String> [-SourceVMPowerOption] <PowerOption> [ <CommonParameters>]
```

Detailed Description

The **Stop-MvmcSourceVirtualMachine** cmdlet shuts down or powers off a source virtual machine. Specify a connection to a VMware vSphere host or a computer that runs VMware vCenter Server.

Parameters

-GuestVmId<String>

Specifies the ID for a virtual machine. To obtain an object that contains this ID, use the **Get-MvmcSourceVirtualMachine** cmdlet.

Aliases	none
Required?	true
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceConnection<MvmcSourceConnection>

Specifies a connection to a vSphere host or computer that runs vCenter Server. To obtain a connection, use the **New-MvmcSourceConnection** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-SourceVMPowerOption<PowerOption>

Specifies a power option for the source virtual machine. Valid values are: PowerOff and Shutdown.

Do not specify a value of PowerOn for this parameter.

Aliases	none
Required?	true
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByValue, ByPropertyName)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **System.Object**

Examples

Example 1: Shut down a virtual machine

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores that secure string in the $Password variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $Password, and stores the credentials in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the host named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and then stores that connection object in the $SourceConnection variable.

The fifth command gets all of the virtual machines from the connection object stored in $SourceConnection, and passes them to the **Where-Object** cmdlet. That cmdlet drops all virtual machines except one that has the name VM073. The command stores any virtual machine named VM073 in the $SourceVM variable. For more information, type `Get-Help Where-Object`.

The final command shuts down a virtual machine. The command specifies the connection stored in $SourceConnection. The command uses standard dot notation to refer to the **GuestVmId** member of the object stored in the $SourceVM variable. This ID identifies the virtual machine to shut down. The *SourceVMPowerOption* parameter specifies to shut down, rather than power off, the virtual machine.

```
PS C:\> $SourceUser= "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object PSCredential ($SourceUser, $SourcePassword)
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> $SourceVM = Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection |
Where-Object {$_.Name -match 'VM073'}
 PS C:\> Stop-MvmcSourceVirtualMachine -SourceConnection $SourceConnection -GuestVmId
$SourceVM.GuestVmId -SourceVMPowerOption Shutdown
```

Related topics

New-MvmcSourceConnection

Get-MvmcSourceVirtualMachine

Uninstall-MvmcSourceVMTools

Uninstall-MvmcSourceVMTools

Uninstalls VMware Tools from a virtual machine.

Syntax

```
Parameter Set: Default
Uninstall-MvmcSourceVMTools [-SourceConnection] <MvmcSourceConnection> [-GuestVmId] <String>
[[-GuestCredential] <PSCredential> ] [ <CommonParameters>]
```

Detailed Description

The **Uninstall-MvmcSourceVMTools** cmdlet uninstalls VMware Tools from a source virtual machine. This cmdlet creates a snapshot of the virtual machine, uninstalls VMware Tools, and then restores the virtual machine to the snapshot.

Before you run this cmdlet, for a source virtual machine that runs the Windows operating system, do or verify the following actions:

-- Join the virtual machine to an Active Directory Domain Services (AD DS) domain.

-- Enable remote access through Windows Management Instrumentation (WMI) on the source virtual machine and on the Hyper-V™ host.

-- Use an account to connect to the VMware-based virtual machine that is part of an AD DS domain and is a local administrator on the virtual machine.

To uninstall VMware Tools for a computer that runs the Linux operating system, you must have root credentials. You cannot use the `su` or `sudo` commands.

To uninstall VMware Tools, you must be able to connect to the virtual machine. You cannot uninstall VMware tools for an offline source virtual machine. For computers that run the Linux operating system, enable SSH and listen on port 22.

Parameters

-GuestCredential<PSCredential>

Specifies credentials to establish a connection to the source virtual machine. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`.

Aliases	none

Required?	false
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-GuestVmId<String>

Specifies the ID for a virtual machine. To obtain an object that contains this ID, use the **Get-MvmcSourceVirtualMachine** cmdlet.

Aliases	none
Required?	true
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SourceConnection<MvmcSourceConnection>

Specifies a connection to a VMware vSphere host or computer that runs VMware vCenter Server. To obtain a connection, use the **New-MvmcSourceConnection** cmdlet.

Aliases	none
Required?	true
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

The output type is the type of the objects that the cmdlet emits.

- **System.Object**

Examples

Example 1: Uninstall VMware Tools

The first command stores the string PattiFuller in the $SourceUser variable.

The second command uses the **ConvertTo-SecureString** cmdlet to create a secure string based on the supplied password, and then stores that secure string in the $SourcePassword variable. For more information, type `Get-Help ConvertTo-SecureString`.

The third command uses the **New-Object** cmdlet to create a **PSCredential** object based on the objects stored in $SourceUser and $SourcePassword, and stores the credentials in the $SourceCredential variable. For more information, type `Get-Help New-Object`. As an alternative, use the **Get-Credential** cmdlet to create a **PSCredential** object by using a dialog box. For more information, type `Get-Help Get-Credential`.

The fourth command uses the **New-MvmcSourceConnection** cmdlet to create a connection to the server named ContosoVS03 that uses the **PSCredential** object stored in $SourceCredential, and stores that connection object in the $SourceConnection variable.

The fifth command gets all of the virtual machines from the server specified in the connection object stored in $SourceConnection, and then passes them to the **Where-Object** cmdlet. That cmdlet drops all virtual machines except one named VM073. The command stores any virtual machine named VM073 in the $SourceVM variable. For more information, type `Get-Help Where-Object`.

The final command uninstalls VMware Tools. The command specifies the connection stored in $SourceConnection. The command uses standard dot notation to refer to the **GuestVmId** member of the object stored in $SourceVM. This ID identifies the virtual machine. The command uses the **PSCredential** object stored in $SourceCredential. In this example, the same user has permissions on the VMware host and on the virtual machine itself. If this is not the case, provide a separate credential for the *GuestCredential* parameter.

```
PS C:\> $SourceUser = "PattiFuller"
 PS C:\> $SourcePassword = ConvertTo-SecureString -AsPlainText -Force -String "Password"
 PS C:\> $SourceCredential = New-Object -TypeName System.Management.Automation.PSCredential
-ArgumentList $SourceUser,$SourcePassword
 PS C:\> $SourceConnection = New-MvmcSourceConnection -Server "ContosoVS03" -
SourceCredential $SourceCredential
 PS C:\> $SourceVM = Get-MvmcSourceVirtualMachine -SourceConnection $SourceConnection |
Where-Object {$_.Name -match 'VM073'}
 PS C:\> Uninstall-MvmcSourceVMTools -SourceConnection $SourceConnection -GuestVmId
$SourceVM.GuestVmId -GuestCredential $SourceCredential
```

MVMC – Physical Machine Conversion Cmdlets

New-MvmcP2VSourceConnection

This cmdlet is used to establish a connection with the source machine that is being converted. The credentials specified with this cmdlet should have administrative rights on the source machine. When a connection is established with the source machine, an agent is installed that enables MVMC to retrieve machine details from the source.

Syntax

```
New-MvmcP2VSourceConnection [-PhysicalServer] <string> [-SourceCredential] <pscredential>
[<CommonParameters>]
```

Parameters

-PhysicalServer<String>

The name, IP address or the FQDN of the source machine to be converted.

Required?	True
Position?	0
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)

-SourceCredential<PSCredential>

Specifies credentials to establish a connection to the source virtual machine. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`.

Aliases	None
Required?	False
Position?	1

Default Value	none
Accept Pipeline Input?	True (ByValue)

<CommonParameters>

This cmdlet supports the common parameters: -Verbose, -Debug, -ErrorAction, -ErrorVariable, -OutBuffer, and -OutVariable. For more information, see about_CommonParameters.

Outputs

```
Microsoft.Accelerators.Mvmc.Cmdlet.MvmcP2VSourceConnection
```

Example

```
## Create the credentials
$user = 'username'
$pass = convertto-securestring 'password' -asplaintext -force
$cred = new-object pscredential ($user, $pass)

$SourceMachine = 'Name of the source machine'

$VMName = 'Ph-dcmrr41r03n11'

## Establish connection with Source Machine
$conn = new-mvmcp2vsourceconnection -physicalserver $SourceMachine -sourcecredential
$cred
```

Get-MvmcP2VSourceSystemInformation

This cmdlet retrieves machine information from the source machine. The agent installed on the source machine queries the information and returns it. Source information includes the drive information, network settings and OS information.

Syntax

```
Get-MvmcP2VSourceSystemInformation [-P2VSourceConnection] <MvmcP2VSourceConnection>
[<CommonParameters>]
```

Parameters

-P2VSourceConnection<MVMCP2VSourceConnection>

The source connection object returned from the New-MVMCP2VSourceConnection cmdlet is passed to this cmdlet.

Required?	True
Position?	0
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)

Outputs

```
Microsoft.Accelerators.Mvmc.Engine.VMM.Engine.P2VCommon.SystemInformation
```

Example

```
## Get system information and logical drives
$conn = new-mvmcp2vsourceconnection -physicalserver $SourceMachine -sourcecredential
$cred
$sys = Get-MvmcP2VSourceSystemInformation -P2VSourceConnection $conn
```

New-MvmcP2VRequestParam

This object is the control parameter that defines the structure of the target VM being created. A new request param is first created and the values are updated from subsequent cmdlets. The requestParam object contains the list of drives to be selected, the network settings, the virtual processor count, and the virtual memory count.

Syntax

```
New-MvmcP2VRequestParam  [<CommonParameters>]
```

Outputs

```
Microsoft.Accelerators.Mvmc.Cmdlet.MvmcP2VRequestParam
```

Example

```
## Create the P2V target VM configuration
$p2vparam = New-MvmcP2VRequestParam
$p2vparam.SelectedDrives.AddRange(Drives to be added)
$p2vparam.CpuCount = 1  ##Number of PRocessors on the destination VM
$p2vparam.StartupMemoryInMB = 512 ##Memory for the destination VM
$p2vparam.SelectedNetworkAdapters.add('adapter',  "Switch Name") ##VSwitch Name on
the HyperV Host
```

New-MvmcHyperVHostConnection

This cmdlet is used to establish a connection with the destination Hyper-V server that will host the converted VM.

Syntax

```
New-MvmcHyperVHostConnection [-HyperVServer] <string> [-HostCredential] <pscredential>
[<CommonParameters>]
```

Parameters

-HyperVServer<String>

The name, IP address or FQDN of the destination HyperV Host

Required?	True
Position?	0
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-HostCredential<PSCredential>

Specifies credentials to establish a connection to the source virtual machine. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`.

Required?	True
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

Outputs

```
Microsoft.Accelerators.Mvmc.Cmdlet.MvmcHyperVHostConnection
```

Example

```
## Establish a connection with the Destination HyperV Host
$HyperVHostName = 'Name of the host'
$HyperVHostUser = 'UserName'
$HyperVHostPass = convertto-securestring 'Password' -asplaintext -force
$HyperVHostCred = new-object pscredential ($HyperVHostUser, $HyperVHostPass)

$hvconn = New-MVMCHyperVHostConnection -HyperVServer $HyperVHostName -HostCredential
$HyperVHostCred
```

ConvertTo-MvmcP2V

This cmdlet performs the conversion of the physical machine into the virtual machine. The selected drives from the source machine are converted to VHDs and a new VM is provisioned out of these disks.

Syntax

```
ConvertTo-MvmcP2V [-SourceMachineConnection] <MvmcP2VSourceConnection> [-
DestinationLiteralPath] <string> [-DestinationHyperVHostConnection]
<MvmcHyperVHostConnection> [-TempWorkingFolder] <string> [[-VmName] <string>] [-
P2VRequestParam] <MvmcP2VRequestParam> [<CommonParameters>]
```

Parameters

-SourceMachineConnection<MVMCP2VSourceConnection>

The sourceMachineConnection obtained from the NewMvmcP2VSoureConnection cmdlet.

Required?	True
Position?	0
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-DestinationLiteralPath<String>

This is the final destination of the disks of the virtual machine. This location should have sufficient space to house all of the selected disks being converted.

Required?	True
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-DestinationHyperVHostConnection<MVMCHyperVHostConnection>

The HyperVConnection obtained from the New-HyperVHostConnection cmdlet.

Required?	True
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-TempWorkingFOlder<String>

A temporary workspace on the MVMC converter that will be used for preparing the disks before a virtual machine is provisioned.

Required?	True
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-VMName<String>

An optional parameter to specify the name of the destination virtual machine.

Required?	False
Position?	4
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-P2VRequestParam<MVMCP2VRequestParam>

All of the VM parameters are collected in this param object. This object is created using the New-MvmcP2VRequestParam cmdlet, and is updated based on the needs of the virtual machine.

Required?	True
Position?	5
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

Outputs

`Microsoft.Accelerators.Mvmc.Engine.HyperV.IHyperVVirtualMachine`

Example1

```
$DestinationPath = 'destination' #THis can be a local path (c:\VMPath), if the converter and host are the same machine, else only a share path (\\Server\Share)
$TempWorkingFolder = 'temp folder' #this path is used for disk fixups, and must be a local path (c:\temp)

## P2V conversion
ConvertTo-MvmcP2V -SourceMachineConnection $conn -DestinationLiteralPath $DestinationPath -DestinationHyperVHostConnection $hvconn -TempWorkingFolder $TempWorkingFolder -VmName $VMName -P2VRequestParam $p2vparam -Verbose -Debug
```

Example2 – Complete Script

This script is a representative sample of the entire flow to perform a physical to virtual machine conversion.

```
## Create the credentials
$user = 'User Name'
$pass = convertto-securestring 'Password' -asplaintext -force
```

```
$cred = new-object pscredential ($user, $pass)

## Import the module
Import-Module "C:\Program Files\Microsoft Virtual Machine Converter\MvmcCmdlet.psd1"

$SourceMachine = 'SourceMachineName'
## Get system information and logical drives
$conn = new-mvmcp2vsourceconnection -physicalserver $SourceMachine -sourcecredential
$cred
$sys = Get-MvmcP2VSourceSystemInformation -P2VSourceConnection $conn
$lcs = $sys.LogicalDrives
$lcs | ft driveletter
$nads = $sys.NetworkAdapters

## Create the P2V target VM configuration
$p2vparam = New-MvmcP2VRequestParam

## Disks created for the VM are “Dynamic” by default. TO explicity change the
disk(s) to be fixed disks, the following step should be done.

## $lcs[0].IsFixed = $true


$p2vparam.SelectedDrives.AddRange($lcs)
$p2vparam.CpuCount = 1  ##Number of PRocessors on the destination VM
$p2vparam.StartupMemoryInMB = 512 ##Memory for the destination VM
$p2vparam.UseDynamicMemory = $false ##Memory Static or Dynamic
$p2vparam.SelectedNetworkAdapters.add($nads[0], "External") ##VSwitch Name on the
HyperV Host

$HyperVHostName = 'DestinationHostName'
$HyperVHostUser = 'DestinationUserName'
$HyperVHostPass = convertto-securestring 'DestinationPassword' -asplaintext -force
$HyperVHostCred = new-object pscredential ($HyperVHostUser, $HyperVHostPass)

$hvconn = New-MVMCHyperVHostConnection -HyperVServer $HyperVHostName -HostCredential
$HyperVHostCred

$DestinationPath = 'FinalPath' #THis can be a local path (c:\VMPath), if the
converter and host are the same machine, else only a share path (\\Server\Share)
$TempWorkingFolder = 'Temp Path' #this path is used for disk fixups, and must be a
local path (c:\temp)
$VMName = 'VM Name'

## P2V conversion
ConvertTo-MvmcP2V -SourceMachineConnection $conn -DestinationLiteralPath
$DestinationPath -DestinationHyperVHostConnection $hvconn -TempWorkingFolder
$TempWorkingFolder -VmName $VMName -P2VRequestParam $p2vparam -Verbose -Debug
```

ConvertTo-MvmcP2VVirtualHardDisk

This cmdlet converts selected drives into VHDs.

Syntax

```
ConvertTo-MvmcP2VVirtualHardDisk [-SourceMachineConnection] <MvmcP2VSourceConnection> [-
DestinationHostname] <string> [-DestinationHostCredential] <pscredential> [-
DestinationLiteralPath] <string> [-SelectedDrives] <LogicalDrive[]> [<CommonParameters>]
```

Parameters

-SourceMachineConnection<MVMCP2VSourceConnection>

The sourceMachineConnection obtained from the NewMvmcP2VSoureConnection cmdlet.

Required?	True
Position?	0
Default Value	none
Accept Pipeline Input?	True (ByValue)
Accept Wildcard Characters?	false

-DestinationHostName<String>

The name of the destination server where the converted disks will be stored

Required?	True
Position?	1
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-DestinationHostCredential<PSCredential>

Specifies credentials to establish a connection to the source virtual machine. To obtain a **PSCredential** object, use the **Get-Credential** cmdlet. For more information, type `Get-Help Get-Credential`.

Required?	False
Position?	2
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-DestinationLiteralPath<String>

The destination on the host where the converted disks will be stored.

Required?	True
Position?	3
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

-SelectedDrives<LogicalDrive[]>

A list of drives that are to be converted. The list of drives can be obtained from the Get-MvmcP2VSourceSystemInformation cmdlet.

Required?	True
Position?	4
Default Value	none
Accept Pipeline Input?	True (ByPropertyName)
Accept Wildcard Characters?	false

Outputs

`System.String`