WorkshopPLUS

Microsoft Azure Infrastructure as a Service (IaaS)

Automating Virtual Machine Management with Microsoft Azure PowerShell Cmdlets (Classic)

Student Lab Manual

V1.7, September 10, 2015

Information in this document is subject to change without notice. The example companies, organizations, products, people, and events depicted herein are fictitious. No association with any real company, organization, product, person or event is intended or should be inferred. 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, trademarked, 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.

© 2010 Microsoft Corporation. All rights reserved.

Microsoft, MS-DOS, MS, Windows, Windows NT, MSDN, Active Directory, BizTalk, SQL Server, SharePoint, Outlook, PowerPoint, FrontPage, Visual Basic, Visual C++, Visual J++, Visual InterDev, Visual SourceSafe, Visual C#, Visual J#,  and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the U.S.A. and/or other countries.

Other product and company names herein may be the trademarks of their respective owners.

Contents

[Automating VM Management with Microsoft Azure PowerShell Cmdlets 4](#_Toc419982820)

[Overview 4](#_Toc419982821)

[Objectives 4](#_Toc419982822)

[Prerequisites 4](#_Toc419982823)

[Exercises 1 – Obtaining your subscriptions credentials 5](#_Toc419982824)

[Task 1 – Import Azure account information 5](#_Toc419982825)

[Task 2 - Select your Subscription 6](#_Toc419982826)

[Task 3 - Create your storage account (if you don’t already have one in the subscription) 6](#_Toc419982827)

[Exercise 2: Provisioning a Virtual Machine using Microsoft Azure PowerShell Cmdlets 7](#_Toc419982828)

[Task 1 - Provisioning a Virtual Machine 7](#_Toc419982829)

[Exercise 3: Using PowerShell CmdLets for Advanced Provisioning 10](#_Toc419982830)

[Task 1 - Performing Custom Provisioning 10](#_Toc419982831)

[Task 2 - Post Provisioning Configuration 11](#_Toc419982832)

[Task 3 - Performing Changes That Require a Reboot 14](#_Toc419982833)

[Task 4 - Managing Disk Images 15](#_Toc419982834)

[Task 5 - Imaging, Exporting and Importing Virtual Machine Configurations 17](#_Toc419982835)

[Task 6 - Exporting and Importing Virtual Machine Configuration 19](#_Toc419982836)

[Task 7 - Managing Remote Desktop (RDP) and SSH Connectivity 20](#_Toc419982837)

[Summary 21](#_Toc419982838)

# Automating VM Management with Microsoft Azure PowerShell Cmdlets

## Overview

In this hands-on lab you will understand the capabilities of automating the deployment and management of virtual machines in Microsoft Azure.

### Objectives

In this hands-on lab, you will learn how to:

* Provision virtual machines
* Perform post provisioning configuration
* Reboot or start a virtual machine
* Manage disk and image libraries
* Export and Import virtual machines
* Enable or disable virtual machine endpoints

### Prerequisites

The following is required to complete this hands-on lab:

* [Microsoft Azure PowerShell](http://msdn.microsoft.com/en-us/library/windowsazure/jj156055)
* A Microsoft Azure subscription - [sign up for a free trial](http://aka.ms/WATK-FreeTrial)

## Exercises 1 – Obtaining your subscriptions credentials

This hands-on lab includes the following exercises:

* [Provisioning a Virtual Machine using PowerShell CmdLets](http://localhost:64285/HOL.htm#Exercise1)
* [Using PowerShell CmdLets for Advanced Provisioning](http://localhost:64285/HOL.htm#Exercise2)

In order to complete this lab, you will need your subscription's secure credentials. Microsoft Azure allows you to connect to Azure from your local machine but you will need login permissions.

We will be using the Add-AzureAccount PowerShell command to retrieve information from your Azure subscription(s). When you enter this command in the Microsoft Azure PowerShell Cmdlet console, a prompt will appear that requests your login credentials. Once you log in, all subscription information related to this login will be downloaded into the PowerShell session. If you were running an automated PowerShell script and wanted to parse this information, using the Azure-AddAccount command would be the way to proceed.

### Task 1 – Import Azure account information

In this task, you will log on to the Microsoft Azure Portal and download your Azure subscription information. This information contains the secure credentials and additional information about your Microsoft Azure Subscription to use in your development environment.

1. Start **Microsoft Azure PowerShell**. If you are using Windows 8.x, you will right click on the tile and select **Run as Administrator**.
2. In the PowerShell command prompt, type the following command.

PowerShell

Add-AzureAccount

1. A dialog box will appear that will ask you for your login credentials for your Azure account. This login may have co-admin permissions to multiple Azure subscriptions. All subscription information will be download into your running PowerShell session.
2. Execute the following commands and take note of the subscription name and a storage account name you will use for the exercise. Also make note of the location (data center) of the storage account. You will later select a specific subscription name and storage account name you want to use. NOTE: A location can also be called a ‘region’ in MSDN documentation.

PowerShell

Get-AzureSubscription | select SubscriptionName

Select-AzureSubscription –SubscriptionName ‘<name of your subscription>’

Get-AzureStorageAccount | select StorageAccountName, Location

1. **When you run the Get-AzureStorageAccount command, you will get a list of storage accounts only for the subscription you entered in the Select line of code. If you ran the Get-AzureStorageAccount command right after the Get-AzureSubscription command AND you have multiple subscriptions, you would get an error.**

**NOTE:** If you do **not** have a storage account already created you can use for this exercise you should create one first by following these steps.

Run the following to determine the data center to create your storage account in. Ensure you pick a data center that shows support for **PersistentVMRole**.

PowerShell

Get-AzureLocation

### Task 2 - Select your Subscription

If you have more than one subscription, to get started you want to make sure that the PowerShell command window knows which subscription you want to work with. Note that the name of the Azure subscription is case sensitive.

PowerShell

Select-AzureSubscription –SubscriptionName ‘your subscription name’

### Task 3 - Create your storage account (if you don’t already have one in the subscription)

PowerShell

New-AzureStorageAccount -StorageAccountName '[YOUR-STORAGE-NAME]' -Location '[DC-LOCATION]'

**Note:** The name of your storage account **MUST** be in all lowercase letters/numbers.

1. Execute the following command to set your current storage account for your subscription.  Note that below we first do a 'set' to set the options for the subscription and then we 'select' the Azure subscription into memory.

PowerShell

Set-AzureSubscription -SubscriptionName '[YOUR-SUBSCRIPTION-NAME]' -CurrentStorageAccount '[YOUR-STORAGE-ACCOUNT]'

Select-AzureSubscription -SubscriptionName '[YOUR-SUBSCRIPTION-NAME]'

## Exercise 2: Provisioning a Virtual Machine using Microsoft Azure PowerShell Cmdlets

Now that you have the subscription information loaded into the running PowerShell session, you will learn how to provision a simple virtual machine in Microsoft Azure PowerShell.

### Task 1 - Provisioning a Virtual Machine

The first step to create a virtual machine in Microsoft Azure is to define the virtual machine configuration for items such as endpoints or data disks, and then define which cloud service and data center the virtual machine will reside in.

1. If not already opened, start **Microsoft Azure PowerShell** with administrator privileges.
2. Define the **$dclocation** variable with the location of the storage account you've configured in the getting started section (for example, East US).

PowerShell

$dclocation = '[YOUR-LOCATION]'

Once the location is selected, you will need to create the virtual machine configuration.

To create virtual machine you will need a few pieces of information: The cloud service name that the virtual machine will be contained in, and the virtual machine image name.

1. Select a unique name for your cloud service. To validate the name is not in use you can use **Test-AzureName** cmdlet. It will return true if the service name already exists.

PowerShell

Test-AzureName -Service '[YOUR-CLOUD-SERVICE-NAME]'

$cloudSvcName = '[YOUR-CLOUD-SERVICE-NAME]'

1. Select the virtual machine image you want to use as the basis of the virtual machine.  You must include the file extension (.vhd). To create a virtual machine with PowerShell, you need to enter the specific image name that exists in Azure to use as the template for building your machine.

The command Get-AzureVMImage will return all image names that exist in Azure. If you use this command, you would need to scroll through a long list of machine names, so instead, we’ll use a different command that will look for the latest version of Windows Server 2012 R2 Datacenter.

PowerShell

$image = (Get-AzureVMImage | Where {$\_.ImageFamily -eq "Windows Server 2012 R2 Datacenter" } | sort PublishedDate -Descending | Select-Object -First 1).ImageName

1. Next, choose the virtual machine creation script below based on whether you selected Windows or Linux.
   1. A Windows Virtual Machine from an Image.

PowerShell

$adminUserName = '[YOUR-USER-NAME]'

$adminPassword = '[YOUR-PASSWORD]'

$vmname = 'mytestvm1'

New-AzureQuickVM -AdminUserName $adminUserName -Windows -ServiceName $cloudSvcName -Name $vmname -ImageName $image -Password $adminPassword -Location $dclocation

* 1. A Linux Virtual Machine from an Image. Notice that the image has changed, and the -OS switch specifies Linux as the operating system.

PowerShell *(if you are using a Linux server ~ not necessary for lab steps)*

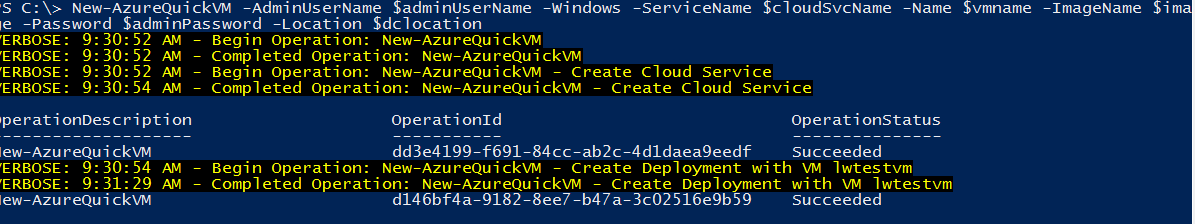
$linuxuser = '[YOUR-USER-USERNAME]'

$adminPassword = '[YOUR-PASSWORD]'

$vmname = 'mytestvm1'

New-AzureQuickVM -Linux -ServiceName $cloudSvcName -Name $vmname -ImageName $image -LinuxUser $linuxuser -Password $adminPassword -Location $dclocation

**Note:** Specifying the **-Location** parameter on **New-AzureQuickVM** or **New-AzureVM** tells the cmdlet to attempt to create a cloud service as a container for the virtual machines. Use this option when creating the first virtual machine and omit it when adding new virtual machines to the same cloud service.

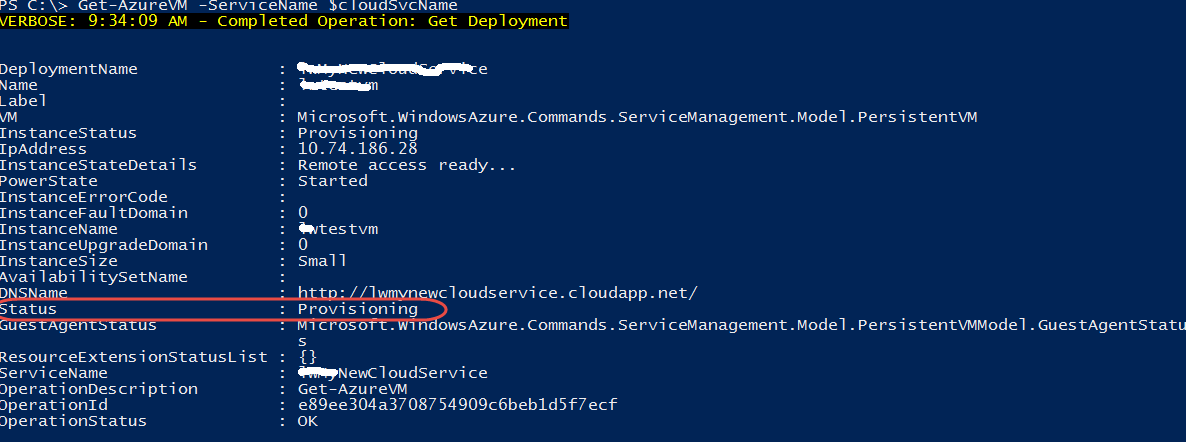


New-AzureQuickVM Cmdlet Sample Output

1. Once the virtual machine has been created you can inspect it using the **Get-AzureVM** cmdlet. The following command will enumerate the details of all the virtual machines in the cloud service.

PowerShell

Get-AzureVM -ServiceName $cloudSvcName



Get-AzureVM –ServiceName $cloudSvcName Sample Output

To be more specific you can use the -Name parameter.

PowerShell

Get-AzureVM -ServiceName $cloudSvcName -Name $vmname

1. The **Microsoft Azure PowerShell Cmdlets** support restart, stop and start operations as well using the **Restart-AzureVM, Stop-AzureVM** and **Start-AzureVM** commands.

With the following commands you will be able to start, stop and restart your virtual machine.

PowerShell

# Restart

Restart-AzureVM -ServiceName $cloudSvcName -Name $vmname

# Shutdown

Stop-AzureVM -ServiceName $cloudSvcName -Name $vmname

# Start

Start-AzureVM -ServiceName $cloudSvcName -Name $vmname

**NOTE:** Make sure your virtual machine finished provisioning before executing these commands.

## Exercise 3: Using PowerShell CmdLets for Advanced Provisioning

In addition to just creating a single uncustomized virtual machine, you can also configure data disks, disk cache settings, networking endpoints and automatically configure domain join settings at provisioning time in addition to batch creating virtual machines using the **New-AzureVMConfig/New-AzureVM** cmdlet combination.

### Task 1 - Performing Custom Provisioning

1. Run the cmdlets below to create two new virtual machines with a 50 GB data disk already attached and a load balanced endpoint open on port 80 for HTTP traffic.

For Windows:

PowerShell

$vmname2 = 'mytestvm2'

$vmname3 = 'mytestvm3'

$vm2 = New-AzureVMConfig -Name $vmname2 -InstanceSize ExtraSmall -ImageName $image | Add-AzureProvisioningConfig -Windows -AdminUserName $adminUserName -Password $adminPassword | Add-AzureDataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'datadisk1' -LUN 0 | Add-AzureEndpoint -Protocol tcp -LocalPort 80 -PublicPort 80 -Name 'lbweb' -LBSetName 'lbweb' -ProbePort 80 -ProbeProtocol http -ProbePath '/'

$vm3 = New-AzureVMConfig -Name $vmname3 -InstanceSize ExtraSmall -ImageName $image | Add-AzureProvisioningConfig -Windows -AdminUserName $adminUserName -Password $adminPassword | Add-AzureDataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'datadisk2' -LUN 0 | Add-AzureEndpoint -Protocol tcp -LocalPort 80 -PublicPort 80 -Name 'lbweb' -LBSetName 'lbweb' -ProbePort 80 -ProbeProtocol http -ProbePath '/'

New-AzureVM -ServiceName $cloudSvcName -VMs $vm2,$vm3

For Linux:

PowerShell *(not necessary for lab steps)*

$vmname2 = 'mytestvm2'

$vmname3 = 'mytestvm3'

$vm2 = New-AzureVMConfig -Name $vmname2 -InstanceSize ExtraSmall -ImageName $image | Add-AzureProvisioningConfig -Linux -LinuxUser $linuxUser -Password $adminPassword | Add-AzureDataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'datadisk1' -LUN 0 | Add-AzureEndpoint -Protocol tcp -LocalPort 80 -PublicPort 80 -Name 'lbweb' -LBSetName 'lbweb' -ProbePort 80 -ProbeProtocol http -ProbePath '/'

$vm3 = New-AzureVMConfig -Name $vmname3 -InstanceSize ExtraSmall -ImageName $image | Add-AzureProvisioningConfig -Linux -LinuxUser $linuxUser -Password $adminPassword | Add-AzureDataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'datadisk2' -LUN 0 | Add-AzureEndpoint -Protocol tcp -LocalPort 80 -PublicPort 80 -Name 'lbweb' -LBSetName 'lbweb' -ProbePort 80 -ProbeProtocol http -ProbePath '/'

New-AzureVM -ServiceName $cloudSvcName -VMs $vm2,$vm3

NOTE**:** You will still need to log into the machine and configure/format the data disk via disk manager. In the next task you will find a walk through for these steps.

### Task 2 - Post Provisioning Configuration

Modifying an existing virtual machine requires retrieving the current settings by calling **Get-AzureVM**, modifying them and then calling the **Update-AzureVM** cmdlet to save the changes.

You can hot add and remove data disks and networking endpoints. Changing disk cache settings requires a reboot as does changing the virtual machine's instance size.

In the following task uses the **Get-AzureVM** cmdlet to retrieve the virtual machine object and send it to the PowerShell Pipeline.

**Add-AzureDataDisk** with the **CreateNew** parameter allows you to dynamically add storage to the virtual machine. In this case we are calling it twice to attach two unformatted blank VHDs to the server each 50 gigs of storage each. The -LUN parameter tells the order of the device being attached and optionally uses the -MediaLocation to specify the location in Storage to keep the newly created VHDs.

**Add-AzureDataDisk** also supports the **Import** parameter to attach a disk in the disk library and **-ImportFrom** to attach a disk that already exists in storage.

The task also adds a new endpoint for TCP port 1433 internally that is listening externally on port 2000 using the **Add-AzureEndpoint** command.

1. Use the following script to hot add data disks and endpoints to an existing virtual machine.

PowerShell

$vmname = 'mytestvm1'

Get-AzureVM -Name $vmname -ServiceName $cloudSvcName | Add-AzureDataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'datadisk1' -LUN 0 | Add-AzureDataDisk -CreateNew -DiskSizeInGB 50 -DiskLabel 'translogs1' -LUN 1 | Add-AzureEndpoint -Protocol tcp -LocalPort 1433 -PublicPort 2000 -Name 'sql' | Update-AzureVM

**NOTE:** To connect to SQL Server you would still need to enable 1433 on the Windows Server firewall to connect.

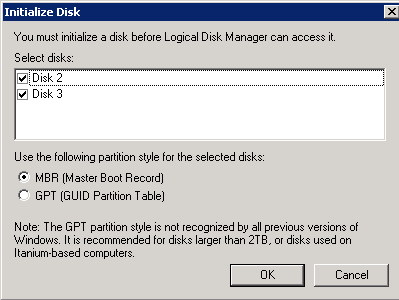
1. Once the **Update-AzureVM** call has completed you will need to log into the machine by using the following command. Use the credentials you've configured when creating the virtual machine to log in.

PowerShell

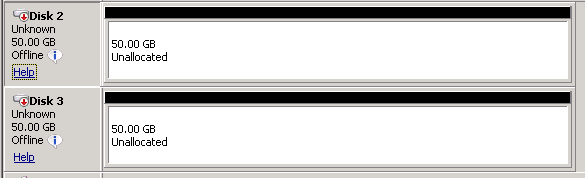
Get-AzureRemoteDesktopFile -ServiceName $cloudSvcName -Name $vmname -LocalPath 'C:\Temp\myvmconnection.rdp' -Launch

**NOTE:** Make sure the **C:\Temp** folder is created or change the path.

1. Once logged open the Disk Management tool. To do this in Windows, you can use the **WINDOWS+X** key combination and then select Disk Management from the menu.
2. The **Initialize Disk** dialog will be prompted. Click **OK** to initialize the disks.



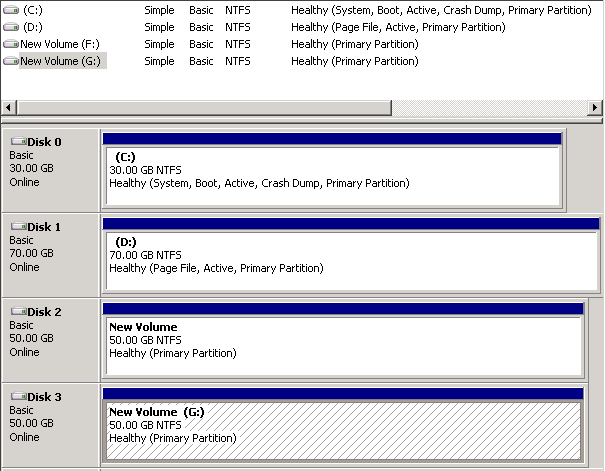
Initializing disks



Virtual Machine Disks

**NOTE:** If the disks are not online, right-click them (on the left side) and click **Online**. Once the disks are online you will need to right-click on one and click **Initialize** (on the left side).

1. Once the disks are initialized you will then need to right-click on the right side and select **New Simple Volume** (software RAID is supported so those are options are available as well). The **New Simple Volume** wizard will guide you to format the disks and mount them for use. Leave all the default values of the wizard.



Formatted disks

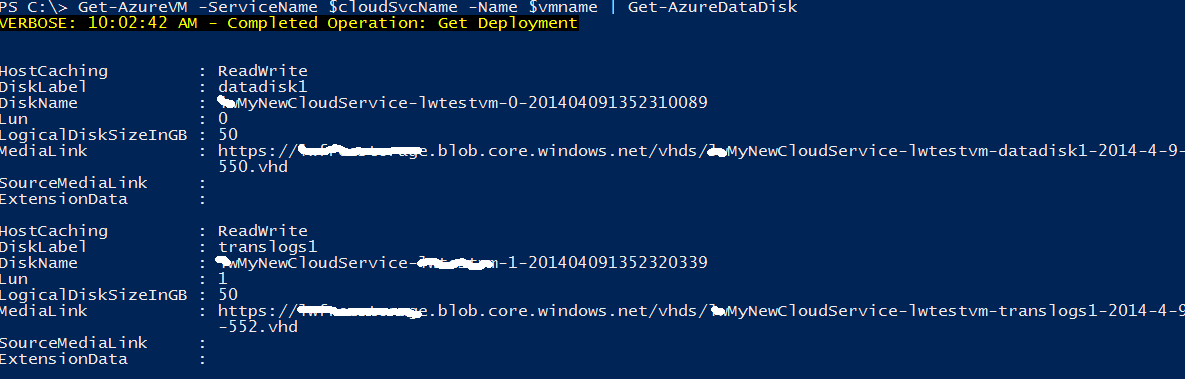
1. You can control the disk cache settings for your data disks by calling **Set-AzureDataDisk** and configuring the HostCaching parameter. Valid values for HostCaching are ReadOnly, ReadWrite and None. With the following script you are enabling Write Cache on a data disk and viewing the resulting change.

NOTE: This change is at the host level and will not be reflected in disk manager. By default write cache is disabled and read cache is enabled on data disks.

PowerShell

Get-AzureVM -Name $vmName -ServiceName $cloudSvcName | Set-AzureDataDisk -HostCaching ReadWrite -LUN 0 | Set-AzureDataDisk -HostCaching ReadWrite -LUN 1 | Update-AzureVM

Get-AzureVM -ServiceName $cloudSvcName -Name $vmname | Get-AzureDataDisk



Get-AzureVM Cmdlet Sample Output

### Task 3 - Performing Changes That Require a Reboot

Some changes require the virtual machine to be **restarted** when applied. Making changes to the underlying hardware by changing the instance size using **Set-AzureRoleSize**, modifying the OS Disk cache settings with **Set-AzureOSDisk** or moving the virtual machine between subnets using **Set-Subnet** all will result in an automatic restart of the virtual machine.

1. Run the following script to disable the write disk cache, changing the write cache setting of the OS disk from write cache enabled to write cache disabled. Once executed the virtual machine will restart with the new settings.

PowerShell

$vm = Get-AzureVM -ServiceName $cloudSvcName -Name $vmName

Set-AzureDataDisk -HostCaching ReadOnly –Lun 0 –VM $vm | Update-AzureVM

Set-AzureOSDisk -HostCaching ReadOnly –VM $vm

Get-AzureOSDisk –VM $vm

1. Run the following script to change the instance size of a Virtual Machine.

NOTE: The snippet below sets the instance size of the specified virtual machine. This does require a reboot as the new hardware is provisioned.

PowerShell

Get-AzureVM -ServiceName $cloudSvcName -Name $vmName | Set-AzureVMSize -InstanceSize Medium | Update-AzureVM

### Task 4 - Managing Disk Images

An image is a VHD that you use as a template to create a new virtual machine. An image is a template because, unlike a running virtual machine, it doesn't have specific settings such as the computer name and user account settings. When you create a virtual machine from an image, an operating system disk is automatically created for the new virtual machine.

It is simple to view all of the data disks or images in the disk and image repository with PowerShell. Running the **Get-AzureDisk** command will enumerate all of the data disks in your subscription.

1. Use the following command to retrieve all your subscriptions' disks.

PowerShell

Get-AzureDisk

1. You can use PowerShell's built in capabilities to limit the results. For instance, with this example you will be able to find a specific virtual machine's VHD image.

PowerShell

$vmname = 'mytestvm2'

Get-AzureDisk | Where {$\_.AttachedTo.RoleName -eq $vmname }

NOTE: mytestvm2 is created in Task 1 of this exercise.

1. Currently, when a virtual machine is removed the underlying VHDs are not removed as well. PowerShell allows you to clean up the underlying storage when removing a virtual machine.

The following script removes a specific Virtual Machine as well as its disks.

PowerShell

$vmname = 'mytestvm2'

$vmDisks = Get-AzureDisk | Where {$\_.AttachedTo.RoleName -eq $vmname }

Remove-AzureVM -ServiceName $cloudSvcName -Name $vmname

$vmDisks | foreach {Remove-AzureDisk -DiskName $\_.DiskName -DeleteVHD}

**NOTE:** If you get an exception saying the disk is in use when deleting the disks, wait a few minutes until the virtual machine has been completely deleted from Microsoft Azure.

1. Similar functionality exists for managing the image repository on your subscription. With this script you will identify user created images (as opposed to images provided by Microsoft Azure in the Image Gallery).

PowerShell

Get-AzureVMImage | Where { $\_.Category -eq 'User' }

### Task 5 - Imaging, Exporting and Importing Virtual Machine Configurations

Microsoft Azure IaaS provides the capability to customize a virtual machine, generalize it using a tool like sysprep, and then capture the virtual machine to the image library. This functionality allows you to create customized images that you can then re-use to generate multiple identical machines. The steps to accomplish this from PowerShell are relatively simple.

1. Execute the following script to create a virtual machine that will be the start of the image.
   1. For Windows virtual machines.

PowerShell

$vmname = 'winvmforimg'

New-AzureVMConfig -Name $vmname -InstanceSize Small -ImageName $image | Add-AzureProvisioningConfig -Windows -AdminUserName $adminUserName -Password $adminPassword | New-AzureVM -ServiceName $cloudSvcName

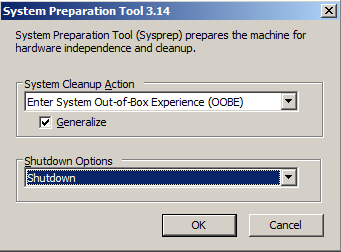
* 1. For Linux virtual machines.

PowerShell

$vmname = 'linuxvmforimg'

New-AzureVMConfig -Name $vmname -InstanceSize Small -ImageName $image | Add-AzureProvisioningConfig -Linux -LinuxUser $linuxuser -Password $adminPassword | New-AzureVM -ServiceName $cloudSvcName

1. Now you need to generalize a virtual machine for capturing an image. At this point you would customize the virtual machine with settings required for the captured image.
2. Connect to the Virtual Machine using either RDP or SSH. You can use the **Get-AzureRemoteDesktopFile** cmdlet as shown in Task 2.
3. For Windows, sysprep from within Windows. To do this, open the **Run** dialog and type **sysprep**. In the opened Windows Explorer, double-click the sysprep executable. Then select **Entire System Out-of-Box Experience (OOBE)**, check **Generalize** and select **Shutdown**.



SysPrep

* 1. For Linux virtual machines, run the following script.

PowerShell

sudo /usr/sbin/waagent -deprovision+user

1. Generate a new image using the **Save-AzureVMImage** cmdlet.

NOTE**:** The virtual machine must be completely shut down before running the Save-AzureVMImage cmdlet. You can check the status of the virtual machine by typing in **Get-AzureVM -Name $vmname** and making sure the status is **StoppedVM**.

PowerShell

Save-AzureVMImage -ServiceName $cloudSvcName -Name $vmname -NewImageName '[YOUR-NEW-VM-IMAGE-NAME]' -NewImageLabel '[YOUR-NEW-IMAGE-LABEL]'

**NOTE:** The **Save-AzureVMImage** cmdlet makes a running persistent virtual machine available as an image for reuse. For Windows virtual machines, the image should be sysprepped before capture. After performing the capture, you can delete or reprovision the virtual machine using the PostCaptureAction parameter with Delete | Reprovision value.

1. Verify the image was created by running the following script.

PowerShell

Get-AzureVMImage -ImageName '[YOUR-NEW-VM-IMAGE-NAME]'

### Task 6 - Exporting and Importing Virtual Machine Configuration

The Microsoft Azure PowerShell Cmdlets provide the capability of saving the configuration of a virtual machine. This is useful in scenarios where you need to completely remove the virtual machine but at some point (easily) put it back. It works by understanding the fact that when you remove a virtual machine by default the underlying data and OS disk in storage is not removed. The **Export-AzureVM** cmdlet saves all of the configuration of the virtual machine including the disk names, endpoint settings and so on, to an XML file. This allows you to delete the virtual machine and then later re-create it using the saved configuration.

1. Run the following script to export the virtual machine Configuration and remove the Deployment. (Make sure you create a Temp folder within C: drive before executing the command or change the path)

PowerShell

$vmname = 'mytestvm1'

Export-AzureVM -ServiceName $cloudSvcName -Name $vmname -Path 'C:\Temp\mytestvm1-config.xml'

Remove-AzureVM -ServiceName $cloudSvcName -Name $vmname

**NOTE:** This code saves the configuration of the mytestvm1 virtual machine and then removes it by removing the virtual machine. This cmdlet does NOT delete the associated .vhd.

.

1. Once the deployment has been removed you can then recreate the virtual machine from the saved state. Run the following script to import the virtual machine Configuration into a New Deployment.

PowerShell

Import-AzureVM -Path 'C:\Temp\mytestvm1-config.xml' | New-AzureVM -ServiceName $cloudSvcName

**NOTE:** In deployments with Virtual Networking this will result in a new IP address so it is not recommended for virtual machines that require a persistent IP such as a domain controller.

### Task 7 - Managing Remote Desktop (RDP) and SSH Connectivity

By default all new virtual machines created from the Microsoft Azure PowerShell cmdlets will allow RDP for Windows or SSH. However you can remove or add endpoints by using the **Remove-AzureEndpoint/Add-AzureEndpoints** cmdlets.

To discover the ports for these endpoints you can use the **Get-AzureEndpoint** to learn the public port of the Remote Desktop (RDP) or SSH input endpoint.

1. Run the following script to get the virtual machine endpoints and check which ones are opened.

PowerShell

Get-AzureVM -ServiceName $cloudSvcName -Name $vmname | Get-AzureEndpoint

1. Remove the Remote Desktop endpoint by using the following script.

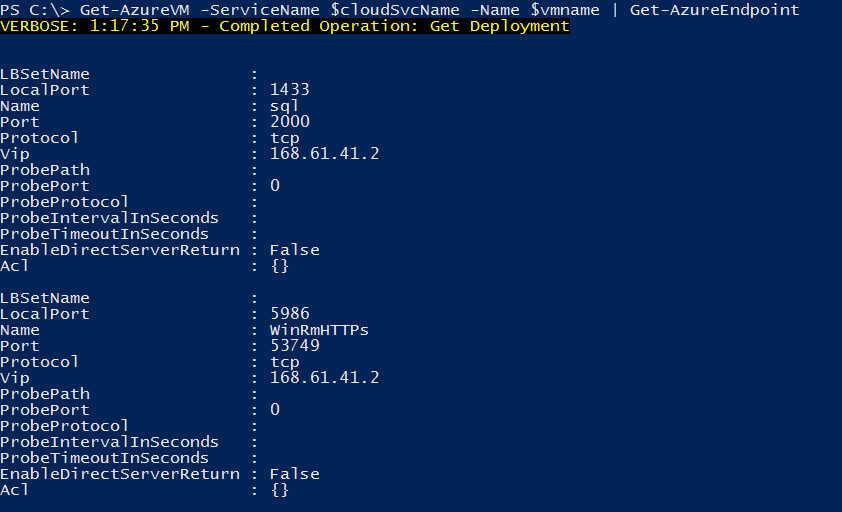
PowerShell

Get-AzureVM -ServiceName $cloudSvcName -Name $vmname | Remove-AzureEndpoint -Name "RemoteDesktop" | Update-AzureVM

1. Use the following script to verify that the endpoint was removed.

PowerShell

Get-AzureVM -ServiceName $cloudSvcName -Name $vmname | Get-AzureEndpoint



Get-AzureEndpoint Cmdlet Output

## Summary

In this hands-on lab you were shown how to configure your subscription in a PowerShell session to manage Microsoft Azure Virtual Machines. You were also shown the basics of how to provision virtual machines and modify them with hot add capabilities and changes that require reboots such as changing the instance size.

In addition you were shown how you can use the Microsoft Azure PowerShell cmdlets to manage your disk and image libraries along with the capability of exporting and import virtual machine configurations.