

Table of Contents

Windows VMs Documentation

Overview

[About Virtual Machines](#)

Quickstarts

[Create VM - Portal](#)

[Create VM - PowerShell](#)

[Create VM - Azure CLI](#)

Tutorials

[1 - Create / manage a VM](#)

[2 - Create / manage disks](#)

[3 - Automate configuration](#)

[4 - Create VM images](#)

[5 - Highly available VMs](#)

[6 - Create a VM scale set](#)

[7 - Load balance VMs](#)

[8 - Manage networking](#)

[9 - Backup virtual machines](#)

[10 - Monitor and update VMs](#)

[11 - Manage VM security](#)

[12 - Team Services CI/CD](#)

[13 - Install a SQL\IIS\.NET stack](#)

[14 - Secure web server with SSL](#)

Samples

[PowerShell](#)

[Azure CLI](#)

Concepts

[Azure Resource Manager](#)

[Regions and availability](#)

[VM types and sizes](#)

- [General purpose](#)
- [Compute optimized](#)
- [Memory optimized](#)
- [Storage optimized](#)
- [GPU optimized](#)
- [High performance compute](#)
- [Azure compute units \(ACU\)](#)
- [Maintenance and updates](#)
- [Disk storage](#)
 - [Managed Disks](#)
 - [Premium storage](#)
 - [Premium storage performance](#)
 - [Standard storage](#)
 - [Scalability targets for disks](#)
 - [Backup and disaster recovery for disks](#)
- [Networking](#)
 - [Auto-scale applications](#)
 - [Infrastructure automation](#)
 - [Security and policy](#)
 - [Monitoring](#)
 - [Backup and recovery](#)
 - [High performance computing](#)
 - [Deployment considerations](#)
 - [Infrastructure guidelines](#)
 - [vCPU quotas](#)
- [How-to guides](#)
 - [Create VMs](#)
 - [Use C#](#)
 - [Use specialized disk](#)
 - [Use a template with C#](#)
 - [Create VM with Chef](#)
 - [Use Java](#)

[Use Python](#)

[Use a template](#)

[Connect to a VM](#)

[Use Azure Hybrid Benefit license](#)

[Use Multitenant Hosting Rights for Windows 10](#)

[Secure VMs](#)

[Encrypt](#)

[Use WinRM](#)

[Use access controls](#)

[Use policies](#)

[Create a Key Vault](#)

[Protect VMs](#)

[Back up a single VM](#)

[Back up multiple VMs](#)

[Manage VMs](#)

[Prepay for VMs - Reserved Instances](#)

[VM usage](#)

[Common PowerShell tasks](#)

[Move a VM](#)

[Change VM size](#)

[Tag a VM](#)

[Run scripts on a VM](#)

[Change temp drive letter](#)

[Change availability set](#)

[Download template](#)

[Azure VM agent](#)

[Planned maintenance](#)

[Scheduled events](#)

[Monitor metadata](#)

[Enable nested virtualization](#)

[Use Images](#)

[Find and use images](#)

[Prepare VM for upload](#)

[Capture VM to image](#)

[Use generalized image](#)

[Build image with Packer](#)

[Use Windows client images](#)

[Download existing disk](#)

[Availability and scale](#)

[Virtual Machine Scale Sets](#)

[High availability](#)

[Vertically scale](#)

[Create VM in availability zone](#)

[Use automation tools](#)

[Chef](#)

[Publish Web App from Visual Studio](#)

[Run applications](#)

[SQL Server](#)

[MongoDB](#)

[SAP on Azure](#)

[MATLAB cluster](#)

[High Performance Computing \(HPC\)](#)

[Manage storage](#)

[Disks FAQs](#)

[Add a disk](#)

[Detach a disk](#)

[Resize a disk](#)

[Snapshot a disk](#)

[Back up unmanaged disks](#)

[Convert disk between Standard and Premium](#)

[Migrate to Premium storage with Azure Site Recovery](#)

[Use File storage](#)

[Deploy disks with template](#)

[Migrate to Managed Disks](#)

Manage networking

- [Create virtual network](#)
- [Open ports to a VM](#)
- [Assign public IP address](#)
- [Use multiple NICs](#)
- [Assign public DNS name](#)
- [DNS resolution](#)

Use VM extensions

- [VM extensions overview](#)
- [Custom Script extension](#)
- [OMS agent extension](#)
- [PowerShell DSC extension](#)
- [Azure Log Collector extension](#)
- [Azure diagnostics extension](#)
- [Exporting VM extensions](#)
- [Troubleshoot extensions](#)
- [Monitoring and Diagnostics Extension](#)
- [Network Watcher Agent](#)

Migrate VMs

- [Migrate AWS and on-premises VMs](#)
- [Migrate from Classic to Azure Resource Manager](#)

Troubleshoot

- [Remote Desktop connections](#)
- [Reset RDP password](#)
- [Reset local password without Azure agent](#)
- [Understand a system reboot](#)
- [Reset NIC](#)
- [Boot diagnostics](#)
- [Creating a VM](#)
- [Deployment issues](#)
- [Restarting or resizing a VM](#)
- [Application access](#)

- [Activation problems](#)
- [Allocation failures](#)
- [Redeploy a VM](#)
- [Common error messages](#)
- [VM recovery access](#)
- [Use PerfInsights](#)
- [Performance Diagnostics VM Extension](#)
- [Use nested virtualization](#)

Reference

- [Azure CLI](#)
- [Azure PowerShell](#)
- [.NET](#)
- [Java](#)
- [Node.js](#)
- [Python](#)
- [Compute REST](#)
- [Managed Disks REST](#)

Resources

- [Author templates](#)
- [Azure Roadmap](#)
- [Community templates](#)
- [Pricing](#)
- [Regional availability](#)
- [Stack Overflow](#)
- [Videos](#)
- [FAQ](#)

Overview of Windows virtual machines in Azure

9/22/2017 • 7 min to read • [Edit Online](#)

Azure Virtual Machines (VM) is one of several types of [on-demand, scalable computing resources](#) that Azure offers. Typically, you choose a VM when you need more control over the computing environment than the other choices offer. This article gives you information about what you should consider before you create a VM, how you create it, and how you manage it.

An Azure VM gives you the flexibility of virtualization without having to buy and maintain the physical hardware that runs it. However, you still need to maintain the VM by performing tasks, such as configuring, patching, and installing the software that runs on it.

Azure virtual machines can be used in various ways. Some examples are:

- **Development and test** – Azure VMs offer a quick and easy way to create a computer with specific configurations required to code and test an application.
- **Applications in the cloud** – Because demand for your application can fluctuate, it might make economic sense to run it on a VM in Azure. You pay for extra VMs when you need them and shut them down when you don't.
- **Extended datacenter** – Virtual machines in an Azure virtual network can easily be connected to your organization's network.

The number of VMs that your application uses can scale up and out to whatever is required to meet your needs.

What do I need to think about before creating a VM?

There are always a multitude of [design considerations](#) when you build out an application infrastructure in Azure. These aspects of a VM are important to think about before you start:

- The names of your application resources
- The location where the resources are stored
- The size of the VM
- The maximum number of VMs that can be created
- The operating system that the VM runs
- The configuration of the VM after it starts
- The related resources that the VM needs

Naming

A virtual machine has a [name](#) assigned to it and it has a computer name configured as part of the operating system. The name of a VM can be up to 15 characters.

If you use Azure to create the operating system disk, the computer name and the virtual machine name are the same. If you [upload and use your own image](#) that contains a previously configured operating system and use it to create a virtual machine, the names can be different. We recommend that when you upload your own image file, you make the computer name in the operating system and the virtual machine name the same.

Locations

All resources created in Azure are distributed across multiple [geographical regions](#) around the world. Usually, the region is called **location** when you create a VM. For a VM, the location specifies where the virtual hard disks are stored.

This table shows some of the ways you can get a list of available locations.

METHOD	DESCRIPTION
Azure portal	Select a location from the list when you create a VM.
Azure PowerShell	Use the Get-AzureRmLocation command.
REST API	Use the List locations operation.

VM size

The [size](#) of the VM that you use is determined by the workload that you want to run. The size that you choose then determines factors such as processing power, memory, and storage capacity. Azure offers a wide variety of sizes to support many types of uses.

Azure charges an [hourly price](#) based on the VM's size and operating system. For partial hours, Azure charges only for the minutes used. Storage is priced and charged separately.

VM Limits

Your subscription has default [quota limits](#) in place that could impact the deployment of many VMs for your project. The current limit on a per subscription basis is 20 VMs per region. Limits can be raised by filing a support ticket requesting an increase.

Operating system disks and images

Virtual machines use [virtual hard disks \(VHDs\)](#) to store their operating system (OS) and data. VHDs are also used for the images you can choose from to install an OS.

Azure provides many [marketplace images](#) to use with various versions and types of Windows Server operating systems. Marketplace images are identified by image publisher, offer, sku, and version (typically version is specified as latest).

This table shows some ways that you can find the information for an image.

METHOD	DESCRIPTION
Azure portal	The values are automatically specified for you when you select an image to use.
Azure PowerShell	Get-AzureRMVMImagePublisher -Location "location" Get-AzureRMVMImageOffer -Location "location" -Publisher "publisherName" Get-AzureRMVMImageSku -Location "location" -Publisher "publisherName" -Offer "offerName"
REST APIs	List image publishers List image offers List image skus

You can choose to [upload and use your own image](#) and when you do, the publisher name, offer, and sku aren't used.

Extensions

VM [extensions](#) give your VM additional capabilities through post deployment configuration and automated tasks.

These common tasks can be accomplished using extensions:

- **Run custom scripts** – The [Custom Script Extension](#) helps you configure workloads on the VM by running your

script when the VM is provisioned.

- **Deploy and manage configurations** – The [PowerShell Desired State Configuration \(DSC\) Extension](#) helps you set up DSC on a VM to manage configurations and environments.
- **Collect diagnostics data** – The [Azure Diagnostics Extension](#) helps you configure the VM to collect diagnostics data that can be used to monitor the health of your application.

Related resources

The resources in this table are used by the VM and need to exist or be created when the VM is created.

RESOURCE	REQUIRED	DESCRIPTION
Resource group	Yes	The VM must be contained in a resource group.
Storage account	Yes	The VM needs the storage account to store its virtual hard disks.
Virtual network	Yes	The VM must be a member of a virtual network.
Public IP address	No	The VM can have a public IP address assigned to it to remotely access it.
Network interface	Yes	The VM needs the network interface to communicate in the network.
Data disks	No	The VM can include data disks to expand storage capabilities.

How do I create my first VM?

You have several choices for creating your VM. The choice that you make depends on the environment you are in.

This table provides information to get you started creating your VM.

METHOD	ARTICLE
Azure portal	Create a virtual machine running Windows using the portal
Templates	Create a Windows virtual machine with a Resource Manager template
Azure PowerShell	Create a Windows VM using PowerShell
Client SDKs	Deploy Azure Resources using C#
REST APIs	Create or update a VM

You hope it never happens, but occasionally something goes wrong. If this situation happens to you, look at the information in [Troubleshoot Resource Manager deployment issues with creating a Windows virtual machine in Azure](#).

How do I manage the VM that I created?

VMs can be managed using a browser-based portal, command-line tools with support for scripting, or directly through APIs. Some typical management tasks that you might perform are getting information about a VM, logging on to a VM, managing availability, and making backups.

Get information about a VM

This table shows you some of the ways that you can get information about a VM.

METHOD	DESCRIPTION
Azure portal	On the hub menu, click Virtual Machines and then select the VM from the list. On the blade for the VM, you have access to overview information, setting values, and monitoring metrics.
Azure PowerShell	For information about using PowerShell to manage VMs, see Create and manage Windows VMs with the Azure PowerShell module .
REST API	Use the Get VM information operation to get information about a VM.
Client SDKs	For information about using C# to manage VMs, see Manage Azure Virtual Machines using Azure Resource Manager and C# .

Log on to the VM

You use the Connect button in the Azure portal to [start a Remote Desktop \(RDP\) session](#). Things can sometimes go wrong when trying to use a remote connection. If this situation happens to you, check out the help information in [Troubleshoot Remote Desktop connections to an Azure virtual machine running Windows](#).

Manage availability

It's important for you to understand how to [ensure high availability](#) for your application. This configuration involves creating multiple VMs to ensure that at least one is running.

In order for your deployment to qualify for our 99.95 VM Service Level Agreement, you need to deploy two or more VMs running your workload inside an [availability set](#). This configuration ensures your VMs are distributed across multiple fault domains and are deployed onto hosts with different maintenance windows. The full [Azure SLA](#) explains the guaranteed availability of Azure as a whole.

Back up the VM

A [Recovery Services vault](#) is used to protect data and assets in both Azure Backup and Azure Site Recovery services. You can use a Recovery Services vault to [deploy and manage backups for Resource Manager-deployed VMs using PowerShell](#).

Next steps

- If your intent is to work with Linux VMs, look at [Azure and Linux](#).
- Learn more about the guidelines around setting up your infrastructure in the [Example Azure infrastructure walkthrough](#).

Create a Windows virtual machine with the Azure portal

11/8/2017 • 2 min to read • [Edit Online](#)

Azure virtual machines can be created through the Azure portal. This method provides a browser-based user interface for creating and configuring virtual machines and all related resources. This quickstart steps through creating a virtual machine and installing a webserver on the VM.

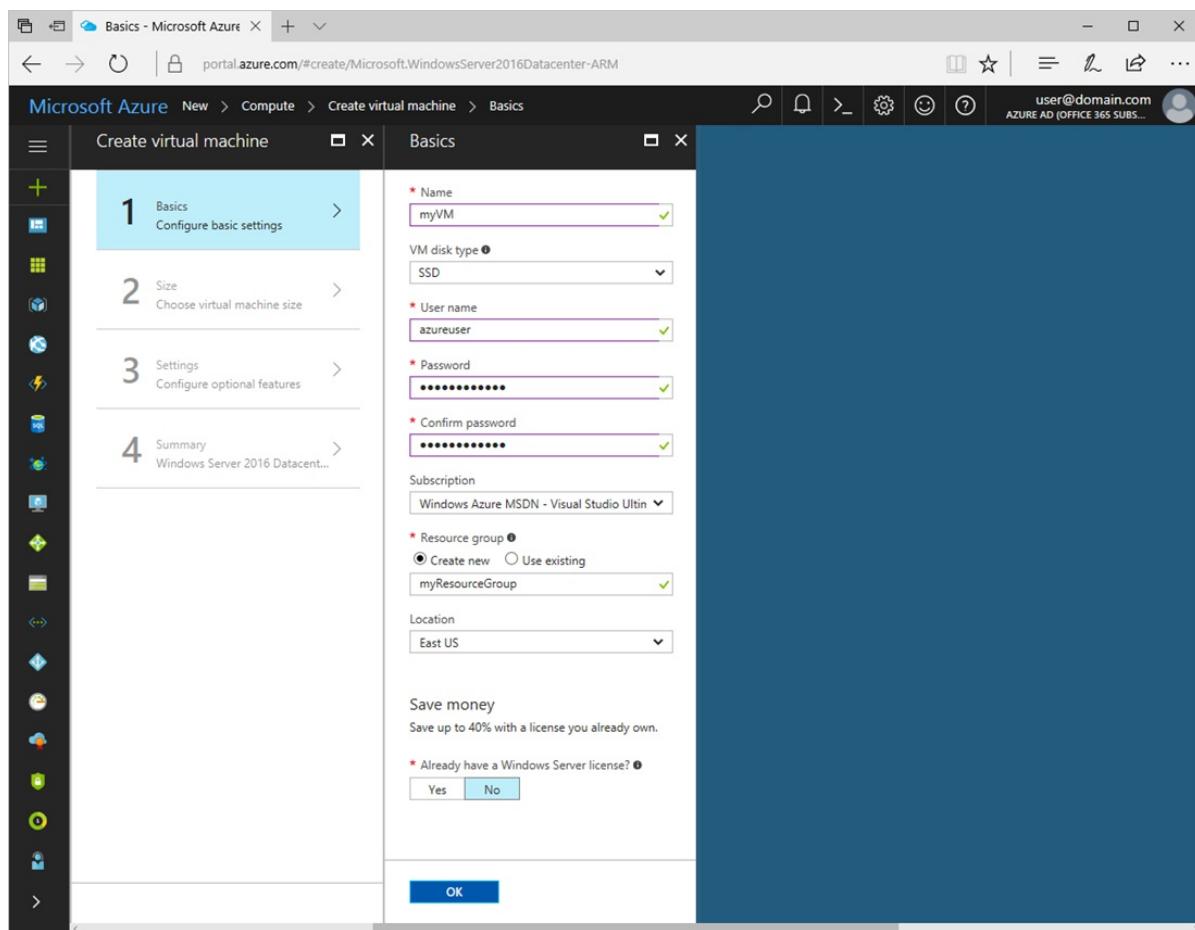
If you don't have an Azure subscription, create a [free account](#) before you begin.

Log in to Azure

Log in to the Azure portal at <http://portal.azure.com>.

Create virtual machine

1. Click the **New** button found on the upper left-hand corner of the Azure portal.
2. Select **Compute**, and then select **Windows Server 2016 Datacenter**.
3. Enter the virtual machine information. The user name and password entered here is used to log in to the virtual machine. When complete, click **OK**.



4. Select a size for the VM. To see more sizes, select **View all** or change the **Supported disk type** filter.

DS1_V2 Standard	DS2_V2 Standard	DS11_V2 Standard
1 Core	2 Cores	2 Cores
3.5 GB	7 GB	14 GB
2 Data disks	4 Data disks	4 Data disks
3200 Max IOPS	6400 Max IOPS	6400 Max IOPS
7 GB Local SSD	14 GB Local SSD	28 GB Local SSD
Load balancing	Load balancing	Load balancing
Premium disk support	Premium disk support	Premium disk support
96.72 USD/MONTH (ESTIMATED)	193.44 USD/MONTH (ESTIMATED)	223.20 USD/MONTH (ESTIMATED)

5. Under **Settings**, keep the defaults and click **OK**.
6. On the summary page, click **Ok** to start the virtual machine deployment.
7. The VM will be pinned to the Azure portal dashboard. Once the deployment has completed, the VM summary automatically opens.

Connect to virtual machine

Create a remote desktop connection to the virtual machine.

1. Click the **Connect** button on the virtual machine properties. A Remote Desktop Protocol file (.rdp file) is created and downloaded.

Microsoft Azure Resource groups > myResourceGroup > myVM

myVM Virtual machine

Search (Ctrl+ /)

Connect Start Restart Stop

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

Essentials

Resource group (change)
myResourceGroup

Status
Running

Location
West US

Subscription name (change)
Windows Azure MSDN - Visual Studio Ultimate

2. To connect to your VM, open the downloaded RDP file. If prompted, click **Connect**. On a Mac, you need an RDP client such as this [Remote Desktop Client](#) from the Mac App Store.

3. Enter the user name and password you specified when creating the virtual machine, then click **Ok**.
4. You may receive a certificate warning during the sign-in process. Click **Yes** or **Continue** to proceed with the connection.

Install IIS using PowerShell

On the virtual machine, start a PowerShell session and run the following command to install IIS.

```
Install-WindowsFeature -name Web-Server -IncludeManagementTools
```

When done, exit the RDP session and return the VM properties in the Azure portal.

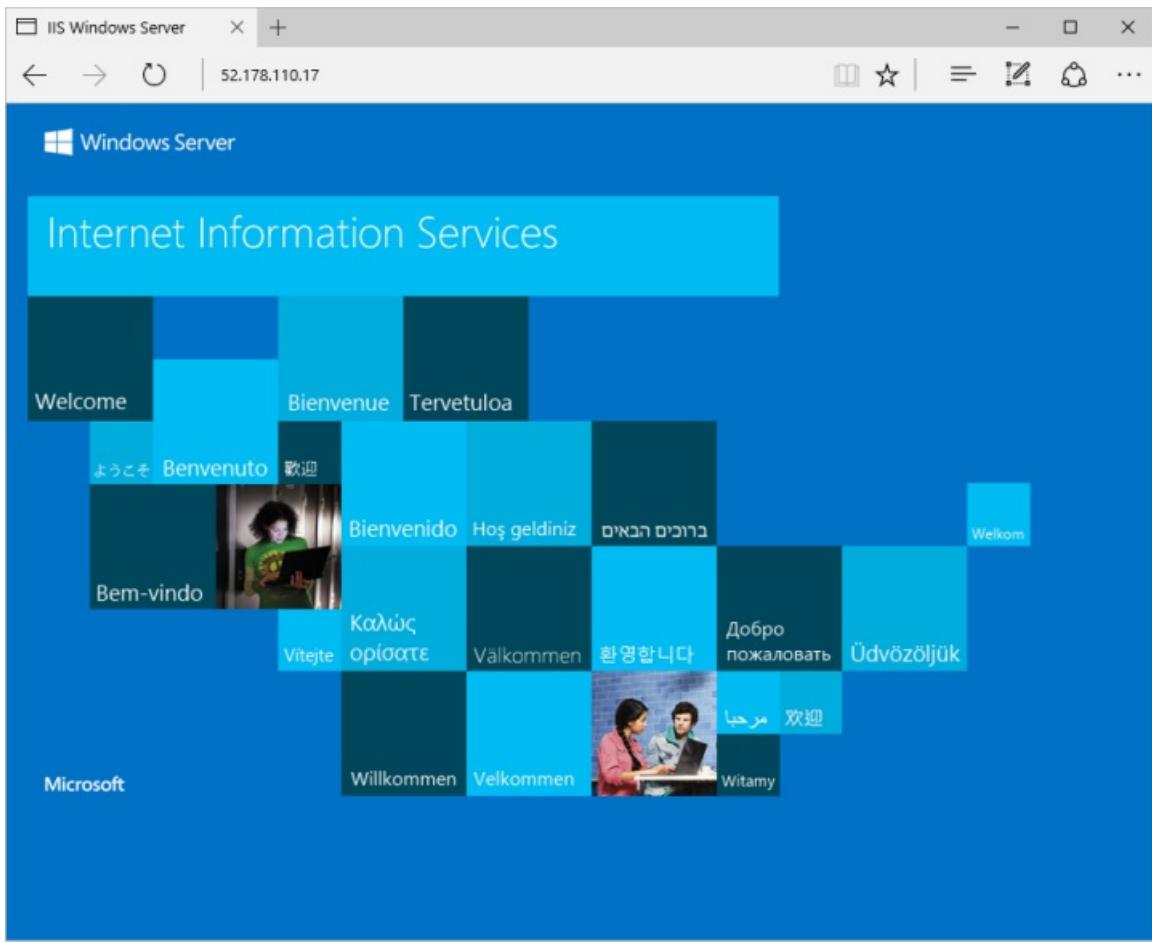
Open port 80 for web traffic

A Network security group (NSG) secures inbound and outbound traffic. When a VM is created from the Azure portal, an inbound rule is created on port 3389 for RDP connections. Because this VM hosts a webserver, an NSG rule needs to be created for port 80.

1. On the virtual machine, click the name of the **Resource group**.
2. Select the **network security group**. The NSG can be identified using the **Type** column.
3. On the left-hand menu, under settings, click **Inbound security rules**.
4. Click on **Add**.
5. In **Name**, type **http**. Make sure **Port range** is set to 80 and **Action** is set to **Allow**.
6. Click **OK**.

View the IIS welcome page

With IIS installed, and port 80 open to your VM, the webserver can now be accessed from the internet. Open a web browser, and enter the public IP address of the VM. The public IP address can be found under *Virtual Machines* in the Azure portal.



Clean up resources

When no longer needed, delete the resource group, virtual machine, and all related resources. To do so, select the resource group for the VM and click **Delete**.

Next steps

In this quick start, you've deployed a simple virtual machine, a network security group rule, and installed a web server. To learn more about Azure virtual machines, continue to the tutorial for Windows VMs.

[Azure Windows virtual machine tutorials](#)

Create a Windows virtual machine with PowerShell

12/15/2017 • 2 min to read • [Edit Online](#)

The Azure PowerShell module is used to create and manage Azure resources from the PowerShell command line or in scripts. This quickstart details using PowerShell to create and Azure virtual machine running Windows Server 2016. Once deployment is complete, we connect to the server and install IIS.

If you don't have an Azure subscription, create a [free account](#) before you begin.

Launch Azure Cloud Shell

The Azure Cloud Shell is a free interactive shell that you can use to run the steps in this article. It has common Azure tools preinstalled and configured to use with your account. Just click the **Copy** to copy the code, paste it into the Cloud Shell, and then press enter to run it. There are two ways to launch the Cloud Shell:

Click Try It in the upper right corner of a code block.	
Click the Cloud Shell button on the menu in the upper right of the Azure portal.	

If you choose to install and use the PowerShell locally, this tutorial requires the Azure PowerShell module version 5.1.1 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#). If you are running PowerShell locally, you also need to run `Login-AzureRmAccount` to create a connection with Azure.

Create resource group

Create an Azure resource group with [New-AzureRmResourceGroup](#). A resource group is a logical container into which Azure resources are deployed and managed.

```
New-AzureRmResourceGroup -Name myResourceGroup -Location EastUS
```

Create virtual machine

Create the virtual machine with [New-AzureRmVM](#). You just need to provide names for each of the resources and the New-AzureRMVM cmdlet will create them for you if they don't already exist.

When running this step, you are prompted for credentials. The values that you enter are configured as the user name and password for the virtual machine.

```
New-AzureRmVm ` 
    -ResourceGroupName "myResourceGroup" ` 
    -Name "myVM" ` 
    -Location "East US" ` 
    -VirtualNetworkName "myVnet" ` 
    -SubnetName "mySubnet" ` 
    -SecurityGroupName "myNetworkSecurityGroup" ` 
    -PublicIpAddressName "myPublicIpAddress" ` 
    -OpenPorts 80,3389
```

Connect to virtual machine

After the deployment has completed, create a remote desktop connection with the virtual machine.

Use the [Get-AzureRmPublicIpAddress](#) command to return the public IP address of the virtual machine. Take note of this IP Address so you can connect to it with your browser to test web connectivity in a future step.

```
Get-AzureRmPublicIpAddress -ResourceGroupName myResourceGroup | Select IPAddress
```

Use the following command, on your local machine, to create a remote desktop session with the virtual machine. Replace the IP address with the *publicIPAddress* of your virtual machine. When prompted, enter the credentials used when creating the virtual machine.

```
mstsc /v:<publicIpAddress>
```

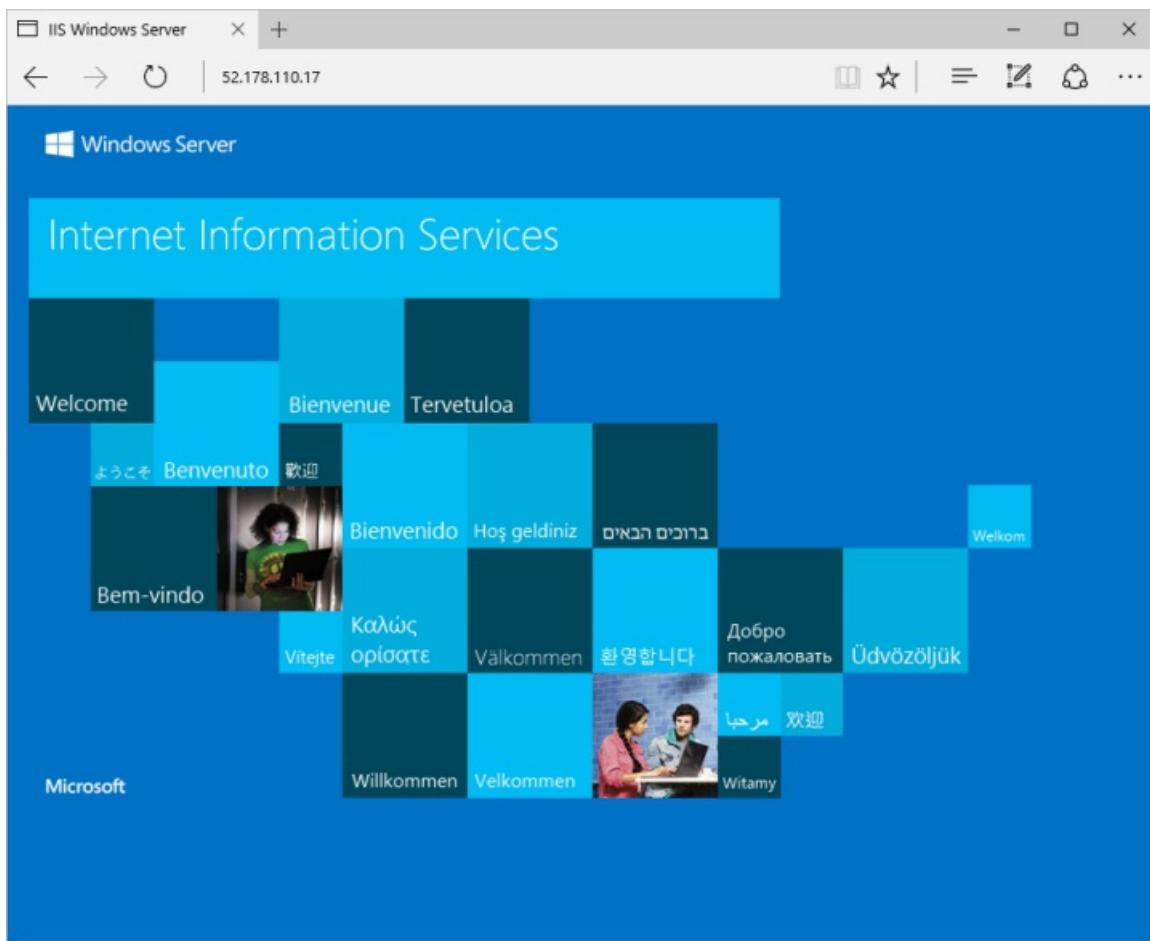
Install IIS via PowerShell

Now that you have logged in to the Azure VM, you can use a single line of PowerShell to install IIS and enable the local firewall rule to allow web traffic. Open a PowerShell prompt on the VM and run the following command:

```
Install-WindowsFeature -name Web-Server -IncludeManagementTools
```

View the IIS welcome page

With IIS installed and port 80 now open on your VM from the Internet, you can use a web browser of your choice to view the default IIS welcome page. Be sure to use the *publicIpAddress* you documented above to visit the default page.



Clean up resources

When no longer needed, you can use the [Remove-AzureRmResourceGroup](#) command to remove the resource group, VM, and all related resources.

```
Remove-AzureRmResourceGroup -Name myResourceGroup
```

Next steps

In this quick start, you've deployed a simple virtual machine, a network security group rule, and installed a web server. To learn more about Azure virtual machines, continue to the tutorial for Windows VMs.

[Azure Windows virtual machine tutorials](#)

Create a Windows virtual machine with the Azure CLI

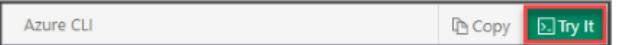
11/8/2017 • 3 min to read • [Edit Online](#)

The Azure CLI is used to create and manage Azure resources from the command line or in scripts. This quickstart details using the Azure CLI to deploy a virtual machine running Windows Server 2016. Once deployment is complete, we connect to the server and install IIS.

If you don't have an Azure subscription, create a [free account](#) before you begin.

Launch Azure Cloud Shell

The Azure Cloud Shell is a free interactive shell that you can use to run the steps in this article. It has common Azure tools preinstalled and configured to use with your account. Just click the **Copy** to copy the code, paste it into the Cloud Shell, and then press enter to run it. There are two ways to launch the Cloud Shell:

Click Try It in the upper right corner of a code block.	
Click the Cloud Shell button on the menu in the upper right of the Azure portal .	

If you choose to install and use the CLI locally, this quickstart requires that you are running the Azure CLI version 2.0.4 or later. Run `az --version` to find the version. If you need to install or upgrade, see [Install Azure CLI 2.0](#).

Create a resource group

Create a resource group with [az group create](#). An Azure resource group is a logical container into which Azure resources are deployed and managed.

The following example creates a resource group named *myResourceGroup* in the *eastus* location.

```
az group create --name myResourceGroup --location eastus
```

Create virtual machine

Create a VM with [az vm create](#).

The following example creates a VM named *myVM*. This example uses *azureuser* for an administrative user name and *myPassword12* as the password. Update these values to something appropriate to your environment. These values are needed when creating a connection with the virtual machine.

```
az vm create --resource-group myResourceGroup --name myVM --image win2016datacenter --admin-username azureuser --admin-password myPassword12
```

When the VM has been created, the Azure CLI shows information similar to the following example. Take note of the `publicIpAddress`. This address is used to access the VM.

```
{  
    "fqdns": "",  
    "id": "/subscriptions/d5b9d4b7-6fc1-0000-0000-  
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM",  
    "location": "eastus",  
    "macAddress": "00-0D-3A-23-9A-49",  
    "powerState": "VM running",  
    "privateIpAddress": "10.0.0.4",  
    "publicIpAddress": "52.174.34.95",  
    "resourceGroup": "myResourceGroup"  
}
```

Open port 80 for web traffic

By default only RDP connections are allowed in to Windows virtual machines deployed in Azure. If this VM is going to be a webserver, you need to open port 80 from the Internet. Use the `az vm open-port` command to open the desired port.

```
az vm open-port --port 80 --resource-group myResourceGroup --name myVM
```

Connect to virtual machine

Use the following command to create a remote desktop session with the virtual machine. Replace the IP address with the public IP address of your virtual machine. When prompted, enter the credentials used when creating the virtual machine.

```
mstsc /v:Public IP Address
```

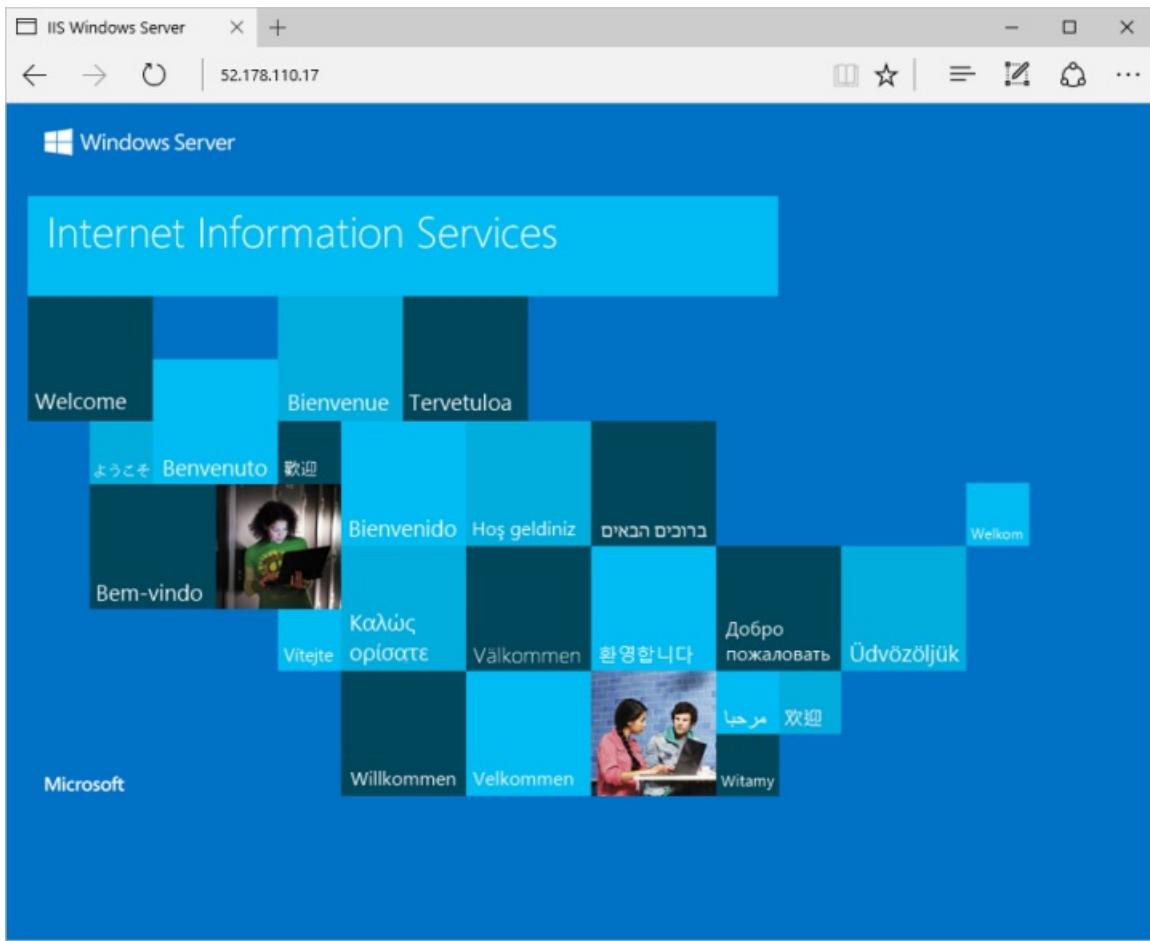
Install IIS using PowerShell

Now that you have logged in to the Azure VM, you can use a single line of PowerShell to install IIS and enable the local firewall rule to allow web traffic. Open a PowerShell prompt and run the following command:

```
Install-WindowsFeature -name Web-Server -IncludeManagementTools
```

View the IIS welcome page

With IIS installed and port 80 now open on your VM from the Internet, you can use a web browser of your choice to view the default IIS welcome page. Be sure to use the public IP address you documented above to visit the default page.



Clean up resources

When no longer needed, you can use the [az group delete](#) command to remove the resource group, VM, and all related resources.

```
az group delete --name myResourceGroup
```

Next steps

In this quick start, you've deployed a simple virtual machine, a network security group rule, and installed a web server. To learn more about Azure virtual machines, continue to the tutorial for Windows VMs.

[Azure Windows virtual machine tutorials](#)

Create and Manage Windows VMs with the Azure PowerShell module

11/16/2017 • 9 min to read • [Edit Online](#)

Azure virtual machines provide a fully configurable and flexible computing environment. This tutorial covers basic Azure virtual machine deployment items such as selecting a VM size, selecting a VM image, and deploying a VM. You learn how to:

- Create and connect to a VM
- Select and use VM images
- View and use specific VM sizes
- Resize a VM
- View and understand VM state

Launch Azure Cloud Shell

The Azure Cloud Shell is a free interactive shell that you can use to run the steps in this article. It has common Azure tools preinstalled and configured to use with your account. Just click the **Copy** to copy the code, paste it into the Cloud Shell, and then press enter to run it. There are two ways to launch the Cloud Shell:

Click Try It in the upper right corner of a code block.	
Click the Cloud Shell button on the menu in the upper right of the Azure portal.	

If you choose to install and use the PowerShell locally, this tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#). If you are running PowerShell locally, you also need to run `Login-AzureRmAccount` to create a connection with Azure.

Create resource group

Create a resource group with the [New-AzureRmResourceGroup](#) command.

An Azure resource group is a logical container into which Azure resources are deployed and managed. A resource group must be created before a virtual machine. In this example, a resource group named *myResourceGroupVM* is created in the *EastUS* region.

```
New-AzureRmResourceGroup -ResourceGroupName myResourceGroupVM -Location EastUS
```

The resource group is specified when creating or modifying a VM, which can be seen throughout this tutorial.

Create virtual machine

A virtual machine must be connected to a virtual network. You communicate with the virtual machine using a

public IP address through a network interface card.

Create virtual network

Create a subnet with [New-AzureRmVirtualNetworkSubnetConfig](#):

```
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig `  
    -Name mySubnet `  
    -AddressPrefix 192.168.1.0/24
```

Create a virtual network with [New-AzureRmVirtualNetwork](#):

```
$vnet = New-AzureRmVirtualNetwork `  
    -ResourceGroupName myResourceGroupVM `  
    -Location EastUS `  
    -Name myVnet `  
    -AddressPrefix 192.168.0.0/16 `  
    -Subnet $subnetConfig
```

Create public IP address

Create a public IP address with [New-AzureRmPublicIpAddress](#):

```
$pip = New-AzureRmPublicIpAddress `  
    -ResourceGroupName myResourceGroupVM `  
    -Location EastUS `  
    -AllocationMethod Static `  
    -Name myPublicIPAddress
```

Create network interface card

Create a network interface card with [New-AzureRmNetworkInterface](#):

```
$nic = New-AzureRmNetworkInterface `  
    -ResourceGroupName myResourceGroupVM `  
    -Location EastUS `  
    -Name myNic `  
    -SubnetId $vnet.Subnets[0].Id `  
    -PublicIpAddressId $pip.Id
```

Create network security group

An Azure [network security group](#) (NSG) controls inbound and outbound traffic for one or many virtual machines. Network security group rules allow or deny network traffic on a specific port or port range. These rules can also include a source address prefix so that only traffic originating at a predefined source can communicate with a virtual machine. To access the IIS webserver that you are installing, you must add an inbound NSG rule.

To create an inbound NSG rule, use [Add-AzureRmNetworkSecurityRuleConfig](#). The following example creates an NSG rule named *myNSGRule* that opens port 3389 for the virtual machine:

```
$nsgRule = New-AzureRmNetworkSecurityRuleConfig `  
-Name myNSGRule `  
-Protocol Tcp `  
-Direction Inbound `  
-Priority 1000 `  
-SourceAddressPrefix * `  
-SourcePortRange * `  
-DestinationAddressPrefix * `  
-DestinationPortRange 3389 `  
-Access Allow
```

Create the NSG using *myNSGRule* with [New-AzureRmNetworkSecurityGroup](#):

```
$nsg = New-AzureRmNetworkSecurityGroup `  
-ResourceGroupName myResourceGroupVM `  
-Location EastUS `  
-Name myNetworkSecurityGroup `  
-SecurityRules $nsgRule
```

Add the NSG to the subnet in the virtual network with [Set-AzureRmVirtualNetworkSubnetConfig](#):

```
Set-AzureRmVirtualNetworkSubnetConfig `  
-Name mySubnet `  
-VirtualNetwork $vnet `  
-NetworkSecurityGroup $nsg `  
-AddressPrefix 192.168.1.0/24
```

Update the virtual network with [Set-AzureRmVirtualNetwork](#):

```
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet
```

Create virtual machine

When creating a virtual machine, several options are available such as operating system image, disk sizing, and administrative credentials. In this example, a virtual machine is created with a name of *myVM* running the latest version of Windows Server 2016 Datacenter.

Set the username and password needed for the administrator account on the virtual machine with [Get-Credential](#):

```
$cred = Get-Credential
```

Create the initial configuration for the virtual machine with [New-AzureRmVMConfig](#):

```
$vm = New-AzureRmVMConfig -VMName myVM -VMSize Standard_D1
```

Add the operating system information to the virtual machine configuration with [Set-AzureRmVMOperatingSystem](#):

```
$vm = Set-AzureRmVMOperatingSystem `  
-VM $vm `  
-Windows `  
-ComputerName myVM `  
-Credential $cred `  
-ProvisionVMAgent -EnableAutoUpdate
```

Add the image information to the virtual machine configuration with [Set-AzureRmVMSourceImage](#):

```
$vm = Set-AzureRmVMSourceImage `  
    -VM $vm `  
    -PublisherName MicrosoftWindowsServer `  
    -Offer WindowsServer `  
    -Skus 2016-Datacenter `  
    -Version latest
```

Add the operating system disk settings to the virtual machine configuration with [Set-AzureRmVMOSDisk](#):

```
$vm = Set-AzureRmVMOSDisk `  
    -VM $vm `  
    -Name myOsDisk `  
    -DiskSizeInGB 128 `  
    -CreateOption FromImage `  
    -Caching ReadWrite
```

Add the network interface card that you previously created to the virtual machine configuration with [Add-AzureRmVMNetworkInterface](#):

```
$vm = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id
```

Create the virtual machine with [New-AzureRmVM](#).

```
New-AzureRmVM -ResourceGroupName myResourceGroupVM -Location EastUS -VM $vm
```

Connect to VM

After the deployment has completed, create a remote desktop connection with the virtual machine.

Run the following commands to return the public IP address of the virtual machine. Take note of this IP Address so you can connect to it with your browser to test web connectivity in a future step.

```
Get-AzureRmPublicIpAddress -ResourceGroupName myResourceGroupVM | Select IPAddress
```

Use the following command, on your local machine, to create a remote desktop session with the virtual machine. Replace the IP address with the *publicIpAddress* of your virtual machine. When prompted, enter the credentials used when creating the virtual machine.

```
mstsc /v:<publicIpAddress>
```

Understand VM images

The Azure marketplace includes many virtual machine images that can be used to create a new virtual machine. In the previous steps, a virtual machine was created using the Windows Server 2016-Datacenter image. In this step, the PowerShell module is used to search the marketplace for other Windows images, which can also act as a base for new VMs. This process consists of finding the publisher, offer, and the image name (Sku).

Use the [Get-AzureRmVMImagePublisher](#) command to return a list of image publishers.

```
Get-AzureRmVMImagePublisher -Location "EastUS"
```

Use the [Get-AzureRmVMImageOffer](#) to return a list of image offers. With this command, the returned list is filtered on the specified publisher.

```
Get-AzureRmVMImageOffer -Location "EastUS" -PublisherName "MicrosoftWindowsServer"
```

Offer	PublisherName	Location
Windows-HUB	MicrosoftWindowsServer	EastUS
WindowsServer	MicrosoftWindowsServer	EastUS
WindowsServer-HUB	MicrosoftWindowsServer	EastUS

The [Get-AzureRmVMImageSku](#) command will then filter on the publisher and offer name to return a list of image names.

```
Get-AzureRmVMImageSku -Location "EastUS" -PublisherName "MicrosoftWindowsServer" -Offer "WindowsServer"
```

Skus	Offer	PublisherName	Location
2008-R2-SP1	WindowsServer	MicrosoftWindowsServer	EastUS
2008-R2-SP1-smalldisk	WindowsServer	MicrosoftWindowsServer	EastUS
2012-Datacenter	WindowsServer	MicrosoftWindowsServer	EastUS
2012-Datacenter-smalldisk	WindowsServer	MicrosoftWindowsServer	EastUS
2012-R2-Datacenter	WindowsServer	MicrosoftWindowsServer	EastUS
2012-R2-Datacenter-smalldisk	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter-Server-Core	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter-Server-Core-smalldisk	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter-smalldisk	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter-with-Containers	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter-with-Containers-smalldisk	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Datacenter-with-RDSH	WindowsServer	MicrosoftWindowsServer	EastUS
2016-Nano-Server	WindowsServer	MicrosoftWindowsServer	EastUS

This information can be used to deploy a VM with a specific image. This example sets the image name on the VM object. Refer to the previous examples in this tutorial for complete deployment steps.

```
$vm = Set-AzureRmVMSourceImage `<br/>`  
-VM $vm `<br/>`  
-PublisherName MicrosoftWindowsServer `<br/>`  
-Offer WindowsServer `<br/>`  
-Skus 2016-Datacenter-with-Containers `<br/>`  
-Version latest
```

Understand VM sizes

A virtual machine size determines the amount of compute resources such as CPU, GPU, and memory that are made available to the virtual machine. Virtual machines need to be created with a size appropriate for the expected work load. If workload increases, an existing virtual machine can be resized.

VM Sizes

The following table categorizes sizes into use cases.

TYPE	SIZES	DESCRIPTION
General purpose	DSv2, Dv2, DS, D, Av2, A0-7	Balanced CPU-to-memory. Ideal for dev / test and small to medium applications and data solutions.
Compute optimized	Fs, F	High CPU-to-memory. Good for medium traffic applications, network appliances, and batch processes.
Memory optimized	GS, G, DSv2, DS, Dv2, D	High memory-to-CPU. Great for relational databases, medium to large caches, and in-memory analytics.
Storage optimized	Ls	High disk throughput and IO. Ideal for Big Data, SQL, and NoSQL databases.
GPU	NV, NC	Specialized VMs targeted for heavy graphic rendering and video editing.
High performance	H, A8-11	Our most powerful CPU VMs with optional high-throughput network interfaces (RDMA).

Find available VM sizes

To see a list of VM sizes available in a particular region, use the [Get-AzureRmVmSize](#) command.

```
Get-AzureRmVmSize -Location EastUS
```

Resize a VM

After a VM has been deployed, it can be resized to increase or decrease resource allocation.

Before resizing a VM, check if the desired size is available on the current VM cluster. The [Get-AzureRmVmSize](#) command returns a list of sizes.

```
Get-AzureRmVmSize -ResourceGroupName myResourceGroupVM -VMName myVM
```

If the desired size is available, the VM can be resized from a powered-on state, however it is rebooted during the operation.

```
$vm = Get-AzureRmVM -ResourceGroupName myResourceGroupVM -VMName myVM
$vm.HardwareProfile.VmSize = "Standard_D4"
Update-AzureRmVM -VM $vm -ResourceGroupName myResourceGroupVM
```

If the desired size is not on the current cluster, the VM needs to be deallocated before the resize operation can occur. Note, when the VM is powered back on, any data on the temp disk are removed, and the public IP address change unless a static IP address is being used.

```

Stop-AzureRmVM -ResourceGroupName myResourceGroupVM -Name "myVM" -Force
$vm = Get-AzureRmVM -ResourceGroupName myResourceGroupVM -VMName myVM
$vm.HardwareProfile.VmSize = "Standard_F4s"
Update-AzureRmVM -VM $vm -ResourceGroupName myResourceGroupVM
Start-AzureRmVM -ResourceGroupName myResourceGroupVM -Name $vm.name

```

VM power states

An Azure VM can have one of many power states. This state represents the current state of the VM from the standpoint of the hypervisor.

Power states

POWER STATE	DESCRIPTION
Starting	Indicates the virtual machine is being started.
Running	Indicates that the virtual machine is running.
Stopping	Indicates that the virtual machine is being stopped.
Stopped	Indicates that the virtual machine is stopped. Note that virtual machines in the stopped state still incur compute charges.
Deallocating	Indicates that the virtual machine is being deallocated.
Deallocated	Indicates that the virtual machine is completely removed from the hypervisor but still available in the control plane. Virtual machines in the Deallocated state do not incur compute charges.
-	Indicates that the power state of the virtual machine is unknown.

Find power state

To retrieve the state of a particular VM, use the [Get-AzureRmVM](#) command. Be sure to specify a valid name for a virtual machine and resource group.

```

Get-AzureRmVM ` 
-ResourceGroupName myResourceGroupVM ` 
-Name myVM ` 
-Status | Select @n="Status"; e={$_.Statuses[1].Code}

```

Output:

```

Status
-----
PowerState/running

```

Management tasks

During the lifecycle of a virtual machine, you may want to run management tasks such as starting, stopping, or deleting a virtual machine. Additionally, you may want to create scripts to automate repetitive or complex tasks. Using Azure PowerShell, many common management tasks can be run from the command line or in scripts.

Stop virtual machine

Stop and deallocate a virtual machine with [Stop-AzureRmVM](#):

```
Stop-AzureRmVM -ResourceGroupName myResourceGroupVM -Name "myVM" -Force
```

If you want to keep the virtual machine in a provisioned state, use the `-StayProvisioned` parameter.

Start virtual machine

```
Start-AzureRmVM -ResourceGroupName myResourceGroupVM -Name myVM
```

Delete resource group

Deleting a resource group also deletes all resources contained within.

```
Remove-AzureRmResourceGroup -Name myResourceGroupVM -Force
```

Next steps

In this tutorial, you learned about basic VM creation and management such as how to:

- Create and connect to a VM
- Select and use VM images
- View and use specific VM sizes
- Resize a VM
- View and understand VM state

Advance to the next tutorial to learn about VM disks.

[Create and Manage VM disks](#)

Manage Azure disks with PowerShell

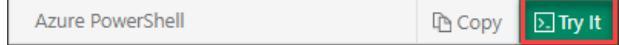
12/19/2017 • 5 min to read • [Edit Online](#)

Azure virtual machines use disks to store the VMs operating system, applications, and data. When creating a VM it is important to choose a disk size and configuration appropriate to the expected workload. This tutorial covers deploying and managing VM disks. You learn about:

- OS disks and temporary disks
- Data disks
- Standard and Premium disks
- Disk performance
- Attaching and preparing data disks

Launch Azure Cloud Shell

The Azure Cloud Shell is a free interactive shell that you can use to run the steps in this article. It has common Azure tools preinstalled and configured to use with your account. Just click the **Copy** to copy the code, paste it into the Cloud Shell, and then press enter to run it. There are two ways to launch the Cloud Shell:

Click Try It in the upper right corner of a code block.	
Click the Cloud Shell button on the menu in the upper right of the Azure portal.	

If you choose to install and use the PowerShell locally, this tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#). If you are running PowerShell locally, you also need to run `Login-AzureRmAccount` to create a connection with Azure.

Default Azure disks

When an Azure virtual machine is created, two disks are automatically attached to the virtual machine.

Operating system disk - Operating system disks can be sized up to 4 terabyte, and hosts the VMs operating system. The OS disk is assigned a drive letter of *c*: by default. The disk caching configuration of the OS disk is optimized for OS performance. The OS disk **should not** host applications or data. For applications and data, use a data disk, which is detailed later in this article.

Temporary disk - Temporary disks use a solid-state drive that is located on the same Azure host as the VM. Temp disks are highly performant and may be used for operations such as temporary data processing. However, if the VM is moved to a new host, any data stored on a temporary disk is removed. The size of the temporary disk is determined by the VM size. Temporary disks are assigned a drive letter of *d*: by default.

Temporary disk sizes

Type	VM Size	Max Temp Disk Size (GB)
General purpose	A and D series	800
Compute optimized	F series	800
Memory optimized	D and G series	6144
Storage optimized	L series	5630
GPU	N series	1440
High performance	A and H series	2000

Azure data disks

Additional data disks can be added for installing applications and storing data. Data disks should be used in any situation where durable and responsive data storage is desired. Each data disk has a maximum capacity of 1 terabyte. The size of the virtual machine determines how many data disks can be attached to a VM. For each VM vCPU, two data disks can be attached.

Max data disks per VM

Type	VM Size	Max Data Disks per VM
General purpose	A and D series	32
Compute optimized	F series	32
Memory optimized	D and G series	64
Storage optimized	L series	64
GPU	N series	48
High performance	A and H series	32

VM disk types

Azure provides two types of disk.

Standard disk

Standard Storage is backed by HDDs, and delivers cost-effective storage while still being performant. Standard disks are ideal for a cost effective dev and test workload.

Premium disk

Premium disks are backed by SSD-based high-performance, low-latency disk. Perfect for VMs running production workload. Premium Storage supports DS-series, DSv2-series, GS-series, and FS-series VMs. Premium disks come in five types (P10, P20, P30, P40, P50), the size of the disk determines the disk type. When selecting, a disk size the value is rounded up to the next type. For example, if the size is below 128 GB the disk type will be P10, between 129 and 512 P20, 512 for P30, P40 for 2TB and P50 4TB .

Premium disk performance

PREMIUM STORAGE DISK TYPE	P10	P20	P30
Disk size (round up)	128 GB	512 GB	1,024 GB (1 TB)
IOPS per disk	500	2,300	5,000
Throughput per disk	100 MB/s	150 MB/s	200 MB/s

While the above table identifies max IOPS per disk, a higher level of performance can be achieved by striping multiple data disks. For instance, 64 data disks can be attached to Standard_GS5 VM. If each of these disks are sized as a P30, a maximum of 80,000 IOPS can be achieved. For detailed information on max IOPS per VM, see [VM types and sizes](#).

Create and attach disks

To complete the example in this tutorial, you must have an existing virtual machine. If needed, this [script sample](#) can create one for you. When working through the tutorial, replace the resource group and VM names where needed.

Create the initial configuration with [New-AzureRmDiskConfig](#). The following example configures a disk that is 128 gigabytes in size.

```
$diskConfig = New-AzureRmDiskConfig -Location EastUS -CreateOption Empty -DiskSizeGB 128
```

Create the data disk with the [New-AzureRmDisk](#) command.

```
$dataDisk = New-AzureRmDisk -ResourceGroupName myResourceGroup -DiskName myDataDisk -Disk $diskConfig
```

Get the virtual machine that you want to add the data disk to with the [Get-AzureRmVM](#) command.

```
$vm = Get-AzureRmVM -ResourceGroupName myResourceGroup -Name myVM
```

Add the data disk to the virtual machine configuration with the [Add-AzureRmVMDataDisk](#) command.

```
$vm = Add-AzureRmVMDataDisk -VM $vm -Name myDataDisk -CreateOption Attach -ManagedDiskId $dataDisk.Id -Lun 1
```

Update the virtual machine with the [Update-AzureRmVM](#) command.

```
Update-AzureRmVM -ResourceGroupName myResourceGroup -VM $vm
```

Prepare data disks

Once a disk has been attached to the virtual machine, the operating system needs to be configured to use the disk. The following example shows how to manually configure the first disk added to the VM. This process can also be automated using the [custom script extension](#).

Manual configuration

Create an RDP connection with the virtual machine. Open up PowerShell and run this script.

```
Get-Disk | Where partitionstyle -eq 'raw' | `  
Initialize-Disk -PartitionStyle MBR -PassThru | `  
New-Partition -AssignDriveLetter -UseMaximumSize | `  
Format-Volume -FileSystem NTFS -NewFileSystemLabel "myDataDisk" -Confirm:$false
```

Next steps

In this tutorial, you learned about VM disks topics such as:

- OS disks and temporary disks
- Data disks
- Standard and Premium disks
- Disk performance
- Attaching and preparing data disks

Advance to the next tutorial to learn about automating VM configuration.

[Automate VM configuration](#)

How to customize a Windows virtual machine in Azure

12/14/2017 • 4 min to read • [Edit Online](#)

To configure virtual machines (VMs) in a quick and consistent manner, some form of automation is typically desired. A common approach to customize a Windows VM is to use [Custom Script Extension for Windows](#). In this tutorial you learn how to:

- Use the Custom Script Extension to install IIS
- Create a VM that uses the Custom Script Extension
- View a running IIS site after the extension is applied

Launch Azure Cloud Shell

The Azure Cloud Shell is a free interactive shell that you can use to run the steps in this article. It has common Azure tools preinstalled and configured to use with your account. Just click the **Copy** to copy the code, paste it into the Cloud Shell, and then press enter to run it. There are two ways to launch the Cloud Shell:

Click Try It in the upper right corner of a code block.	
Click the Cloud Shell button on the menu in the upper right of the Azure portal.	

If you choose to install and use the PowerShell locally, this tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#). If you are running PowerShell locally, you also need to run `Login-AzureRmAccount` to create a connection with Azure.

Custom script extension overview

The Custom Script Extension downloads and executes scripts on Azure VMs. This extension is useful for post deployment configuration, software installation, or any other configuration / management task. Scripts can be downloaded from Azure storage or GitHub, or provided to the Azure portal at extension run time.

The Custom Script extension integrates with Azure Resource Manager templates, and can also be run using the Azure CLI, PowerShell, Azure portal, or the Azure Virtual Machine REST API.

You can use the Custom Script Extension with both Windows and Linux VMs.

Create virtual machine

Before you can create a VM, create a resource group with [New-AzureRmResourceGroup](#). The following example creates a resource group named *myResourceGroupAutomate* in the *EastUS* location:

```
New-AzureRmResourceGroup -ResourceGroupName myResourceGroupAutomate -Location EastUS
```

Set an administrator username and password for the VMs with [Get-Credential](#):

```
$cred = Get-Credential
```

Now you can create the VM with [New-AzureRmVM](#). The following example creates the required virtual network components, the OS configuration, and then creates a VM named *myVM*:

```
# Create a subnet configuration
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig `

    -Name mySubnet `

    -AddressPrefix 192.168.1.0/24

# Create a virtual network
$vnet = New-AzureRmVirtualNetwork `

    -ResourceGroupName myResourceGroupAutomate `

    -Location EastUS `

    -Name myVnet `

    -AddressPrefix 192.168.0.0/16 `

    -Subnet $subnetConfig

# Create a public IP address and specify a DNS name
$publicIP = New-AzureRmPublicIpAddress `

    -ResourceGroupName myResourceGroupAutomate `

    -Location EastUS `

    -AllocationMethod Static `

    -IdleTimeoutInMinutes 4 `

    -Name "myPublicIP"

# Create an inbound network security group rule for port 3389
$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig `

    -Name myNetworkSecurityGroupRuleRDP `

    -Protocol Tcp `

    -Direction Inbound `

    -Priority 1000 `

    -SourceAddressPrefix * `

    -SourcePortRange * `

    -DestinationAddressPrefix * `

    -DestinationPortRange 3389 `

    -Access Allow

# Create an inbound network security group rule for port 80
$nsgRuleWeb = New-AzureRmNetworkSecurityRuleConfig `

    -Name myNetworkSecurityGroupRuleWWW `

    -Protocol Tcp `

    -Direction Inbound `

    -Priority 1001 `

    -SourceAddressPrefix * `

    -SourcePortRange * `

    -DestinationAddressPrefix * `

    -DestinationPortRange 80 `

    -Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup `

    -ResourceGroupName myResourceGroupAutomate `

    -Location EastUS `

    -Name myNetworkSecurityGroup `

    -SecurityRules $nsgRuleRDP,$nsgRuleWeb

# Create a virtual network card and associate with public IP address and NSG
$nic = New-AzureRmNetworkInterface `

    -Name myNic `

    -ResourceGroupName myResourceGroupAutomate `

    -Location EastUS `

    -SubnetId $vnet.Subnets[0].Id `

    -PublicIpAddressId $publicIP.Id `
```

```

# Create a network security group
$ns = New-AzureRmNetworkSecurityGroup -ResourceGroupName myResourceGroupAutomate -Location EastUS -Name myNSG
$rule = New-AzureRmNetworkSecurityRuleConfig -Name AllowRDP -Protocol Tcp -Direction Inbound -Priority 1000 -SourceAddressPrefix Internet -SourcePortRange * -DestinationAddressPrefix * -DestinationPortRange 3389 -Access Allow
Set-AzureRmNetworkSecurityGroup -NetworkSecurityGroupId $ns.Id

# Create a virtual machine configuration
$vmConfig = New-AzureRmVMConfig -VMName myVM -VMSize Standard_DS2 | ` 
Set-AzureRmVMOperatingSystem -Windows -ComputerName myVM -Credential $cred | ` 
Set-AzureRmVMSourceImage -PublisherName MicrosoftWindowsServer ` 
-Offer WindowsServer -Skus 2016-Datacenter -Version latest | ` 
Add-AzureRmVMNetworkInterface -Id $nic.Id

# Create a virtual machine using the configuration
New-AzureRmVM -ResourceGroupName myResourceGroupAutomate -Location EastUS -VM $vmConfig

```

It takes a few minutes for the resources and VM to be created.

Automate IIS install

Use [Set-AzureRmVMExtension](#) to install the Custom Script Extension. The extension runs

`powershell Add-WindowsFeature Web-Server` to install the IIS webserver and then updates the *Default.htm* page to show the hostname of the VM:

```

Set-AzureRmVMExtension -ResourceGroupName myResourceGroupAutomate ` 
-ExtensionName IIS ` 
-VMName myVM ` 
-Publisher Microsoft.Compute ` 
-ExtensionType CustomScriptExtension ` 
-TypeHandlerVersion 1.8 ` 
-SettingString '{"commandToExecute":"powershell Add-WindowsFeature Web-Server; powershell Add-Content -Path` 
\"C:\\\\inetpub\\\\wwwroot\\\\Default.htm\" -Value $($env:computername)"}' ` 
-Location EastUS

```

Test web site

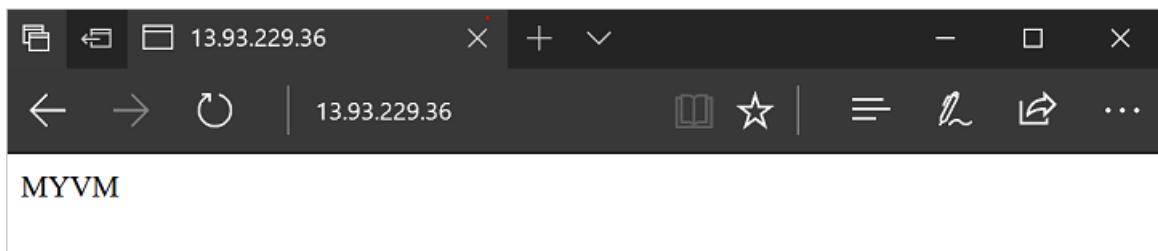
Obtain the public IP address of your load balancer with [Get-AzureRmPublicIPAddress](#). The following example obtains the IP address for *myPublicIP* created earlier:

```

Get-AzureRmPublicIPAddress ` 
-ResourceGroupName myResourceGroupAutomate ` 
-Name myPublicIP | selectIpAddress

```

You can then enter the public IP address in to a web browser. The website is displayed, including the hostname of the VM that the load balancer distributed traffic to as in the following example:



Next steps

In this tutorial, you automated the IIS install on a VM. You learned how to:

- Use the Custom Script Extension to install IIS
- Create a VM that uses the Custom Script Extension
- View a running IIS site after the extension is applied

Advance to the next tutorial to learn how to create custom VM images.

[Create custom VM images](#)

Create a custom image of an Azure VM using PowerShell

12/8/2017 • 4 min to read • [Edit Online](#)

Custom images are like marketplace images, but you create them yourself. Custom images can be used to bootstrap configurations such as preloading applications, application configurations, and other OS configurations. In this tutorial, you create your own custom image of an Azure virtual machine. You learn how to:

- Sysprep and generalize VMs
- Create a custom image
- Create a VM from a custom image
- List all the images in your subscription
- Delete an image

This tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

Before you begin

The steps below detail how to take an existing VM and turn it into a re-usable custom image that you can use to create new VM instances.

To complete the example in this tutorial, you must have an existing virtual machine. If needed, this [script sample](#) can create one for you. When working through the tutorial, replace the resource group and VM names where needed.

Prepare VM

To create an image of a virtual machine, you need to prepare the VM by generalizing the VM, deallocating, and then marking the source VM as generalized in Azure.

Generalize the Windows VM using Sysprep

Sysprep removes all your personal account information, among other things, and prepares the machine to be used as an image. For details about Sysprep, see [How to Use Sysprep: An Introduction](#).

1. Connect to the virtual machine.
2. Open the Command Prompt window as an administrator. Change the directory to `%windir%\system32\sysprep`, and then run `sysprep.exe`.
3. In the **System Preparation Tool** dialog box, select *Enter System Out-of-Box Experience (OOBE)*, and make sure that the *Generalize* check box is selected.
4. In **Shutdown Options**, select *Shutdown* and then click **OK**.
5. When Sysprep completes, it shuts down the virtual machine. **Do not restart the VM.**

Deallocate and mark the VM as generalized

To create an image, the VM needs to be deallocated and marked as generalized in Azure.

Deallocated the VM using [Stop-AzureRmVM](#).

```
Stop-AzureRmVM -ResourceGroupName myResourceGroup -Name myVM -Force
```

Set the status of the virtual machine to `-Generalized` using [Set-AzureRmVm](#).

```
Set-AzureRmVM -ResourceGroupName myResourceGroup -Name myVM -Generalized
```

Create the image

Now you can create an image of the VM by using [New-AzureRmImageConfig](#) and [New-AzureRmImage](#). The following example creates an image named *myImage* from a VM named *myVM*.

Get the virtual machine.

```
$vm = Get-AzureRmVM -Name myVM -ResourceGroupName myResourceGroup
```

Create the image configuration.

```
$image = New-AzureRmImageConfig -Location EastUS -SourceVirtualMachineId $vm.ID
```

Create the image.

```
New-AzureRmImage -Image $image -ImageName myImage -ResourceGroupName myResourceGroup
```

Create VMs from the image

Now that you have an image, you can create one or more new VMs from the image. Creating a VM from a custom image is very similar to creating a VM using a Marketplace image. When you use a Marketplace image, you have to provide the information about the image, image provider, offer, SKU and version. With a custom image, you just need to provide the ID of the custom image resource.

In the following script, we create a variable `$image` to store information about the custom image using [Get-AzureRmImage](#) and then we use [Set-AzureRmVMSourceImage](#) and specify the ID using the `$image` variable we just created.

The script creates a VM named *myVMfromImage* from our custom image in a new resource group named *myResourceGroupFromImage* in the *West US* location.

```
$cred = Get-Credential -Message "Enter a username and password for the virtual machine."  
  
New-AzureRmResourceGroup -Name myResourceGroupFromImage -Location EastUS  
  
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig `  
    -Name mySubnet `  
    -AddressPrefix 192.168.1.0/24  
  
$vnet = New-AzureRmVirtualNetwork `  
    -ResourceGroupName myResourceGroupFromImage `  
    -Location EastUS `  
    -Name MYvNET `  
    -AddressPrefix 192.168.0.0/16 `  
    -Subnet $subnetConfig  
  
$pip = New-AzureRmPublicIpAddress `  
    -ResourceGroupName myResourceGroupFromImage `  
    -Location EastUS `  
    -Name "mypublicdns$(Get-Random)" `  
    -AllocationMethod Static `  
    -IdleTimeoutInMinutes 4
```

```

$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig ` 
    -Name myNetworkSecurityGroupRuleRDP ` 
    -Protocol Tcp ` 
    -Direction Inbound ` 
    -Priority 1000 ` 
    -SourceAddressPrefix * ` 
    -SourcePortRange * ` 
    -DestinationAddressPrefix * ` 
    -DestinationPortRange 3389 ` 
    -Access Allow

$nsg = New-AzureRmNetworkSecurityGroup ` 
    -ResourceGroupName myResourceGroupFromImage ` 
    -Location EastUS ` 
    -Name myNetworkSecurityGroup ` 
    -SecurityRules $nsgRuleRDP

$nic = New-AzureRmNetworkInterface ` 
    -Name myNic ` 
    -ResourceGroupName myResourceGroupFromImage ` 
    -Location EastUS ` 
    -SubnetId $vnet.Subnets[0].Id ` 
    -PublicIpAddressId $pip.Id ` 
    -NetworkSecurityGroupId $nsg.Id

$vmConfig = New-AzureRmVMConfig ` 
    -VMName myVMfromImage ` 
    -VMSize Standard_D1 | Set-AzureRmVMOperatingSystem -Windows ` 
        -ComputerName myComputer ` 
        -Credential $cred

# Here is where we create a variable to store information about the image
$image = Get-AzureRmImage ` 
    -ImageName myImage ` 
    -ResourceGroupName myResourceGroup

# Here is where we specify that we want to create the VM from and image and provide the image ID
$vmConfig = Set-AzureRmVMSourceImage -VM $vmConfig -Id $image.Id

$vmConfig = Add-AzureRmVMNetworkInterface -VM $vmConfig -Id $nic.Id

New-AzureRmVM ` 
    -ResourceGroupName myResourceGroupFromImage ` 
    -Location EastUS ` 
    -VM $vmConfig

```

Image management

Here are some examples of common management image tasks and how to complete them using PowerShell.

List all images by name.

```
$images = Find-AzureRMResource -ResourceType Microsoft.Compute/images
$images.name
```

Delete an image. This example deletes the image named *myOldImage* from the *myResourceGroup*.

```
Remove-AzureRmImage ` 
    -ImageName myOldImage ` 
    -ResourceGroupName myResourceGroup
```

Next steps

In this tutorial, you created a custom VM image. You learned how to:

- Sysprep and generalize VMs
- Create a custom image
- Create a VM from a custom image
- List all the images in your subscription
- Delete an image

Advance to the next tutorial to learn about how highly available virtual machines.

[Create highly available VMs](#)

How to use availability sets

11/22/2017 • 5 min to read • [Edit Online](#)

In this tutorial, you learn how to increase the availability and reliability of your Virtual Machine solutions on Azure using a capability called Availability Sets. Availability sets ensure that the VMs you deploy on Azure are distributed across multiple isolated hardware nodes in a cluster. Doing this ensures that if a hardware or software failure within Azure happens, only a sub-set of your VMs are impacted and that your overall solution remains available and operational.

In this tutorial, you learn how to:

- Create an availability set
- Create a VM in an availability set
- Check available VM sizes
- Check Azure Advisor

This tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

Availability set overview

An Availability Set is a logical grouping capability that you can use in Azure to ensure that the VM resources you place within it are isolated from each other when they are deployed within an Azure datacenter. Azure ensures that the VMs you place within an Availability Set run across multiple physical servers, compute racks, storage units, and network switches. If a hardware or Azure software failure occurs, only a subset of your VMs are impacted, and your overall application stays up and continues to be available to your customers. Availability Sets are an essential capability when you want to build reliable cloud solutions.

Let's consider a typical VM-based solution where you might have 4 front-end web servers and use 2 back-end VMs that host a database. With Azure, you'd want to define two availability sets before you deploy your VMs: one availability set for the web tier and one availability set for the database tier. When you create a new VM you can then specify the availability set as a parameter to the `az vm create` command, and Azure automatically ensures that the VMs you create within the available set are isolated across multiple physical hardware resources. If the physical hardware that one of your Web Server or Database Server VMs is running on has a problem, you know that the other instances of your Web Server and Database VMs remain running because they are on different hardware.

Use Availability Sets when you want to deploy reliable VM-based solutions in Azure.

Create an availability set

You can create an availability set using [New-AzureRmAvailabilitySet](#). In this example, we set both the number of update and fault domains at 2 for the availability set named *myAvailabilitySet* in the *myResourceGroupAvailability* resource group.

Create a resource group.

```
New-AzureRmResourceGroup -Name myResourceGroupAvailability -Location EastUS
```

Create a managed availability set using [New-AzureRmAvailabilitySet](#) with the **-sku aligned** parameter.

```
New-AzureRmAvailabilitySet ` 
-Location EastUS ` 
-Name myAvailabilitySet ` 
-ResourceGroupName myResourceGroupAvailability ` 
-sku aligned ` 
-PlatformFaultDomainCount 2 ` 
-PlatformUpdateDomainCount 2
```

Create VMs inside an availability set

VMs must be created within the availability set to make sure they are correctly distributed across the hardware. You can't add an existing VM to an availability set after it is created.

The hardware in a location is divided into multiple update domains and fault domains. An **update domain** is a group of VMs and underlying physical hardware that can be rebooted at the same time. VMs in the same **fault domain** share common storage as well as a common power source and network switch.

When you create a VM configuration using [New-AzureRMVMConfig](#) you use the `-AvailabilitySetId` parameter to specify the ID of the availability set.

Create two VMs with [New-AzureRmVM](#) in the availability set.

```
$availabilitySet = Get-AzureRmAvailabilitySet ` 
-ResourceGroupName myResourceGroupAvailability ` 
-Name myAvailabilitySet

$cred = Get-Credential -Message "Enter a username and password for the virtual machine."

$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig ` 
-Name mySubnet ` 
-AddressPrefix 192.168.1.0/24
$vnet = New-AzureRmVirtualNetwork ` 
-ResourceGroupName myResourceGroupAvailability ` 
-Location EastUS ` 
-Name myVnet ` 
-AddressPrefix 192.168.0.0/16 ` 
-Subnet $subnetConfig

$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig ` 
-Name myNetworkSecurityGroupRuleRDP ` 
-Protocol Tcp ` 
-Direction Inbound ` 
-Priority 1000 ` 
-SourceAddressPrefix * ` 
-SourcePortRange * ` 
-DestinationAddressPrefix * ` 
-DestinationPortRange 3389 ` 
-Access Allow

$nsg = New-AzureRmNetworkSecurityGroup ` 
-Location eastus ` 
-Name myNetworkSecurityGroup ` 
-ResourceGroupName myResourceGroupAvailability ` 
-SecurityRules $nsgRuleRDP

# Apply the network security group to a subnet
Set-AzureRmVirtualNetworkSubnetConfig ` 
-VirtualNetwork $vnet ` 
-Name mySubnet ` 
-NetworkSecurityGroup $nsg ` 
-AddressPrefix 192.168.1.0/24

# Update the virtual network
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet
```

```
Set-AzureRmVirtualNetwork -Name $vnet
```

```
for ($i=1; $i -le 2; $i++)
{
    $pip = New-AzureRmPublicIpAddress ` 
        -ResourceGroupName myResourceGroupAvailability ` 
        -Location EastUS ` 
        -Name "mypublicdns$(Get-Random)" ` 
        -AllocationMethod Static ` 
        -IdleTimeoutInMinutes 4

    $nic = New-AzureRmNetworkInterface ` 
        -Name myNic$i ` 
        -ResourceGroupName myResourceGroupAvailability ` 
        -Location EastUS ` 
        -SubnetId $vnet.Subnets[0].Id ` 
        -PublicIpAddressId $pip.Id ` 
        -NetworkSecurityGroupId $nsg.Id

    # Here is where we specify the availability set
    $vm = New-AzureRmVMConfig ` 
        -VMName myVM$i ` 
        -VMSize Standard_D1 ` 
        -AvailabilitySetId $availabilitySet.Id

    $vm = Set-AzureRmVMOperatingSystem ` 
        -ComputerName myVM$i ` 
        -Credential $cred ` 
        -VM $vm ` 
        -Windows ` 
        -EnableAutoUpdate ` 
        -ProvisionVMAgent

    $vm = Set-AzureRmVMSourceImage ` 
        -VM $vm ` 
        -PublisherName MicrosoftWindowsServer ` 
        -Offer WindowsServer ` 
        -Skus 2016-Datacenter ` 
        -Version latest

    $vm = Set-AzureRmVMOSDisk ` 
        -VM $vm ` 
        -Name myOsDisk$i ` 
        -DiskSizeInGB 128 ` 
        -CreateOption FromImage ` 
        -Caching ReadWrite

    $vm = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id

    New-AzureRmVM ` 
        -ResourceGroupName myResourceGroupAvailability ` 
        -Location EastUS ` 
        -VM $vm
}
```

It takes a few minutes to create and configure both VMs. When finished, you'll have two virtual machines distributed across the underlying hardware.

If you look at the availability set in the portal by going to Resource Groups > myResourceGroupAvailability > myAvailabilitySet, you should see how the VMs are distributed across the 2 fault and update domains.

Check for available VM sizes

You can add more VMs to the availability set later, but you need to know what VM sizes are available on the hardware. Use [Get-AzureRMVmSize](#) to list all the available sizes on the hardware cluster for the availability set.

```
Get-AzureRmVmSize  
-AvailabilitySetName myAvailabilitySet  
-ResourceGroupName myResourceGroupAvailability
```

Check Azure Advisor

You can also use Azure Advisor to get more information on ways to improve the availability of your VMs. Azure Advisor helps you follow best practices to optimize your Azure deployments. It analyzes your resource configuration and usage telemetry and then recommends solutions that can help you improve the cost effectiveness, performance, high availability, and security of your Azure resources.

Sign in to the [Azure portal](#), select **More services**, and type **Advisor**. The Advisor dashboard displays personalized recommendations for the selected subscription. For more information, see [Get started with Azure Advisor](#).

Next steps

In this tutorial, you learned how to:

- Create an availability set
- Create a VM in an availability set
- Check available VM sizes
- Check Azure Advisor

Advance to the next tutorial to learn about virtual machine scale sets.

[Create a VM scale set](#)

Create a Virtual Machine Scale Set and deploy a highly available app on Windows

12/18/2017 • 7 min to read • [Edit Online](#)

A virtual machine scale set allows you to deploy and manage a set of identical, auto-scaling virtual machines. You can scale the number of VMs in the scale set manually, or define rules to autoscale based on resource usage such as CPU, memory demand, or network traffic. In this tutorial, you deploy a virtual machine scale set in Azure. You learn how to:

- Use the Custom Script Extension to define an IIS site to scale
- Create a load balancer for your scale set
- Create a virtual machine scale set
- Increase or decrease the number of instances in a scale set
- Create autoscale rules

This tutorial requires the Azure PowerShell module version 5.1.1 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

Scale Set overview

A virtual machine scale set allows you to deploy and manage a set of identical, auto-scaling virtual machines. VMs in a scale set are distributed across logic fault and update domains in one or more *placement groups*. These are groups of similarly configured VMs, similar to [availability sets](#).

VMs are created as needed in a scale set. You define autoscale rules to control how and when VMs are added or removed from the scale set. These rules can trigger based on metrics such as CPU load, memory usage, or network traffic.

Scale sets support up to 1,000 VMs when you use an Azure platform image. For workloads with significant installation or VM customization requirements, you may wish to [Create a custom VM image](#). You can create up to 300 VMs in a scale set when using a custom image.

Create an app to scale

Before you can create a scale set, create a resource group with [New-AzureRmResourceGroup](#). The following example creates a resource group named *myResourceGroupAutomate* in the *EastUS* location:

```
New-AzureRmResourceGroup -ResourceGroupName myResourceGroupScaleSet -Location EastUS
```

In an earlier tutorial, you learned how to [Automate VM configuration](#) using the Custom Script Extension. Create a scale set configuration then apply a Custom Script Extension to install and configure IIS:

```

# Create a config object
$vmssConfig = New-AzureRmVmssConfig ` 
    -Location EastUS ` 
    -SkuCapacity 2 ` 
    -SkuName Standard_DS2 ` 
    -UpgradePolicyMode Automatic

# Define the script for your Custom Script Extension to run
$publicSettings = @{
    "fileUris" = (,"https://raw.githubusercontent.com/Azure-Samples/compute-automation-configurations/master/automate-iis.ps1");
    "commandToExecute" = "powershell -ExecutionPolicy Unrestricted -File automate-iis.ps1"
}

# Use Custom Script Extension to install IIS and configure basic website
Add-AzureRmVmssExtension -VirtualMachineScaleSet $vmssConfig ` 
    -Name "customScript" ` 
    -Publisher "Microsoft.Compute" ` 
    -Type "CustomScriptExtension" ` 
    -TypeHandlerVersion 1.8 ` 
    -Setting $publicSettings

```

Create scale load balancer

An Azure load balancer is a Layer-4 (TCP, UDP) load balancer that provides high availability by distributing incoming traffic among healthy VMs. A load balancer health probe monitors a given port on each VM and only distributes traffic to an operational VM. For more information, see the next tutorial on [How to load balance Windows virtual machines](#).

Create a load balancer that has a public IP address and distributes web traffic on port 80:

```

# Create a public IP address
$publicIP = New-AzureRmPublicIpAddress `

-ResourceGroupName myResourceGroupScaleSet `

-Location EastUS `

-AllocationMethod Static `

-Name myPublicIP

# Create a frontend and backend IP pool
$frontendIP = New-AzureRmLoadBalancerFrontendIpConfig `

-Name myFrontEndPool `

-PublicIpAddress $publicIP

$backendPool = New-AzureRmLoadBalancerBackendAddressPoolConfig -Name myBackEndPool

# Create the load balancer
$lb = New-AzureRmLoadBalancer `

-ResourceGroupName myResourceGroupScaleSet `

-Name myLoadBalancer `

-Location EastUS `

-FrontendIpConfiguration $frontendIP `

-BackendAddressPool $backendPool

# Create a load balancer health probe on port 80
Add-AzureRmLoadBalancerProbeConfig -Name myHealthProbe `

-LoadBalancer $lb `

-Protocol tcp `

-Port 80 `

-IntervalInSeconds 15 `

-ProbeCount 2

# Create a load balancer rule to distribute traffic on port 80
Add-AzureRmLoadBalancerRuleConfig `

-Name myLoadBalancerRule `

-LoadBalancer $lb `

-FrontendIpConfiguration $lb.FrontendIpConfigurations[0] `

-BackendAddressPool $lb.BackendAddressPools[0] `

-Protocol Tcp `

-FrontendPort 80 `

-BackendPort 80

# Update the load balancer configuration
Set-AzureRmLoadBalancer -LoadBalancer $lb

```

Create a scale set

Now create a virtual machine scale set with [New-AzureRmVmss](#). The following example creates a scale set named *myScaleSet*:

```

# Reference a virtual machine image from the gallery
Set-AzureRmVmssStorageProfile $vmssConfig `
    -ImageReferencePublisher MicrosoftWindowsServer `
    -ImageReferenceOffer WindowsServer `
    -ImageReferenceSku 2016-Datacenter `
    -ImageReferenceVersion latest

# Set up information for authenticating with the virtual machine
Set-AzureRmVmssOsProfile $vmssConfig `
    -AdminUsername azureuser `
    -AdminPassword P@ssword! `
    -ComputerNamePrefix myVM

# Create the virtual network resources
$subnet = New-AzureRmVirtualNetworkSubnetConfig `
    -Name "mySubnet" `
    -AddressPrefix 10.0.0.0/24
$vnet = New-AzureRmVirtualNetwork `
    -ResourceGroupName "myResourceGroupScaleSet" `
    -Name "myVnet" `
    -Location "EastUS" `
    -AddressPrefix 10.0.0.0/16 `
    -Subnet $subnet
$ipConfig = New-AzureRmVmssIpConfig `
    -Name "myIPConfig" `
    -LoadBalancerBackendAddressPoolsId $lb.BackendAddressPools[0].Id `
    -SubnetId $vnet.Subnets[0].Id

# Attach the virtual network to the config object
Add-AzureRmVmssNetworkInterfaceConfiguration `
    -VirtualMachineScaleSet $vmssConfig `
    -Name "network-config" `
    -Primary $true `
    -IPConfiguration $ipConfig

# Create the scale set with the config object (this step might take a few minutes)
New-AzureRmVmss `
    -ResourceGroupName myResourceGroupScaleSet `
    -Name myScaleSet `
    -VirtualMachineScaleSet $vmssConfig

```

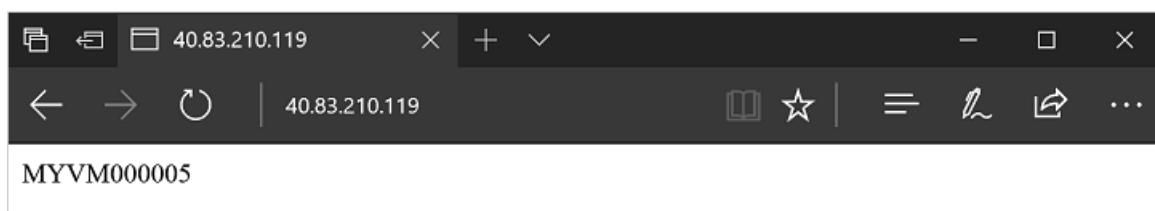
It takes a few minutes to create and configure all the scale set resources and VMs.

Test your app

To see your IIS website in action, obtain the public IP address of your load balancer with [Get-AzureRmPublicIPAddress](#). The following example obtains the IP address for *myPublicIP* created as part of the scale set:

```
Get-AzureRmPublicIPAddress -ResourceGroupName myResourceGroupScaleSet -Name myPublicIP | select IPAddress
```

Enter the public IP address in to a web browser. The app is displayed, including the hostname of the VM that the load balancer distributed traffic to:



To see the scale set in action, you can force-refresh your web browser to see the load balancer distribute traffic

across all the VMs running your app.

Management tasks

Throughout the lifecycle of the scale set, you may need to run one or more management tasks. Additionally, you may want to create scripts that automate various lifecycle-tasks. Azure PowerShell provides a quick way to do those tasks. Here are a few common tasks.

View VMs in a scale set

To view a list of VMs running in your scale set, use [Get-AzureRmVmssVM](#) as follows:

```
# Get current scale set
$scaleset = Get-AzureRmVmss ` 
    -ResourceGroupName myResourceGroupScaleSet ` 
    -VMScaleSetName myScaleSet

# Loop through the instances in your scale set
for ($i=1; $i -le ($scaleset.Sku.Capacity - 1); $i++) {
    Get-AzureRmVmssVM -ResourceGroupName myResourceGroupScaleSet ` 
        -VMScaleSetName myScaleSet ` 
        -InstanceId $i
}
```

Increase or decrease VM instances

To see the number of instances you currently have in a scale set, use [Get-AzureRmVmss](#) and query on `sku.capacity`:

```
Get-AzureRmVmss -ResourceGroupName myResourceGroupScaleSet ` 
    -VMScaleSetName myScaleSet | ` 
    Select -ExpandProperty Sku
```

You can then manually increase or decrease the number of virtual machines in the scale set with [Update-AzureRmVmss](#). The following example sets the number of VMs in your scale set to 3:

```
# Get current scale set
$scaleset = Get-AzureRmVmss ` 
    -ResourceGroupName myResourceGroupScaleSet ` 
    -VMScaleSetName myScaleSet

# Set and update the capacity of your scale set
$scaleset.sku.capacity = 3
Update-AzureRmVmss -ResourceGroupName myResourceGroupScaleSet ` 
    -Name myScaleSet ` 
    -VirtualMachineScaleSet $scaleset
```

It takes a few minutes to update the specified number of instances in your scale set.

Configure autoscale rules

Rather than manually scaling the number of instances in your scale set, you define autoscale rules. These rules monitor the instances in your scale set and respond accordingly based on metrics and thresholds you define. The following example scales out the number of instances by one when the average CPU load is greater than 60% over a 5-minute period. If the average CPU load then drops below 30% over a 5-minute period, the instances are scaled in by one instance:

```

# Define your scale set information
$mySubscriptionId = (Get-AzureRmSubscription).Id
$myResourceGroup = "myResourceGroupScaleSet"
$myScaleSet = "myScaleSet"
$myLocation = "East US"

# Create a scale up rule to increase the number instances after 60% average CPU usage exceeded for a 5-minute
period
$myRuleScaleUp = New-AzureRmAutoscaleRule `

    -MetricName "Percentage CPU" `

    -MetricResourceId
/subscriptions/$mySubscriptionId/resourceGroups/$myResourceGroup/providers/Microsoft.Compute/virtualMachineSca
leSets/$myScaleSet `

    -Operator GreaterThan `

    -MetricStatistic Average `

    -Threshold 60 `

    -TimeGrain 00:01:00 `

    -TimeWindow 00:05:00 `

    -ScaleActionCooldown 00:05:00 `

    -ScaleActionDirection Increase `

    -ScaleActionValue 1

# Create a scale down rule to decrease the number of instances after 30% average CPU usage over a 5-minute
period
$myRuleScaleDown = New-AzureRmAutoscaleRule `

    -MetricName "Percentage CPU" `

    -MetricResourceId
/subscriptions/$mySubscriptionId/resourceGroups/$myResourceGroup/providers/Microsoft.Compute/virtualMachineSca
leSets/$myScaleSet `

    -Operator LessThan `

    -MetricStatistic Average `

    -Threshold 30 `

    -TimeGrain 00:01:00 `

    -TimeWindow 00:05:00 `

    -ScaleActionCooldown 00:05:00 `

    -ScaleActionDirection Decrease `

    -ScaleActionValue 1

# Create a scale profile with your scale up and scale down rules
$myScaleProfile = New-AzureRmAutoscaleProfile `

    -DefaultCapacity 2 `

    -MaximumCapacity 10 `

    -MinimumCapacity 2 `

    -Rules $myRuleScaleUp,$myRuleScaleDown `

    -Name "autoprofile"

# Apply the autoscale rules
Add-AzureRmAutoscaleSetting `

    -Location $myLocation `

    -Name "autosetting" `

    -ResourceGroup $myResourceGroup `

    -TargetResourceId
/subscriptions/$mySubscriptionId/resourceGroups/$myResourceGroup/providers/Microsoft.Compute/virtualMachineSca
leSets/$myScaleSet `

    -AutoscaleProfiles $myScaleProfile

```

For more design information on the use of autoscale, see [autoscale best practices](#).

Next steps

In this tutorial, you created a virtual machine scale set. You learned how to:

- Use the Custom Script Extension to define an IIS site to scale
- Create a load balancer for your scale set

- Create a virtual machine scale set
- Increase or decrease the number of instances in a scale set
- Create autoscale rules

Advance to the next tutorial to learn more about load balancing concepts for virtual machines.

[Load balance virtual machines](#)

How to load balance Windows virtual machines in Azure to create a highly available application

12/15/2017 • 8 min to read • [Edit Online](#)

Load balancing provides a higher level of availability by spreading incoming requests across multiple virtual machines. In this tutorial, you learn about the different components of the Azure load balancer that distribute traffic and provide high availability. You learn how to:

- Create an Azure load balancer
- Create a load balancer health probe
- Create load balancer traffic rules
- Use the Custom Script Extension to create a basic IIS site
- Create virtual machines and attach to a load balancer
- View a load balancer in action
- Add and remove VMs from a load balancer

This tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

Azure load balancer overview

An Azure load balancer is a Layer-4 (TCP, UDP) load balancer that provides high availability by distributing incoming traffic among healthy VMs. A load balancer health probe monitors a given port on each VM and only distributes traffic to an operational VM.

You define a front-end IP configuration that contains one or more public IP addresses. This front-end IP configuration allows your load balancer and applications to be accessible over the Internet.

Virtual machines connect to a load balancer using their virtual network interface card (NIC). To distribute traffic to the VMs, a back-end address pool contains the IP addresses of the virtual (NICs) connected to the load balancer.

To control the flow of traffic, you define load balancer rules for specific ports and protocols that map to your VMs.

Create Azure load balancer

This section details how you can create and configure each component of the load balancer. Before you can create your load balancer, create a resource group with [New-AzureRmResourceGroup](#). The following example creates a resource group named *myResourceGroupLoadBalancer* in the *EastUS* location:

```
New-AzureRmResourceGroup `  
-ResourceGroupName myResourceGroupLoadBalancer `  
-Location EastUS
```

Create a public IP address

To access your app on the Internet, you need a public IP address for the load balancer. Create a public IP address with [New-AzureRmPublicIpAddress](#). The following example creates a public IP address named *myPublicIP* in the *myResourceGroupLoadBalancer* resource group:

```
$publicIP = New-AzureRmPublicIpAddress `  
-ResourceGroupName myResourceGroupLoadBalancer `  
-Location EastUS `  
-AllocationMethod Static `  
-Name myPublicIP
```

Create a load balancer

Create a frontend IP pool with [New-AzureRmLoadBalancerFrontendIpConfig](#). The following example creates a frontend IP pool named *myFrontEndPool* and attaches the *myPublicIP* address:

```
$frontendIP = New-AzureRmLoadBalancerFrontendIpConfig `  
-Name myFrontEndPool `  
-PublicIpAddress $publicIP
```

Create a backend address pool with [New-AzureRmLoadBalancerBackendAddressPoolConfig](#). The VMs attach to this backend pool in the remaining steps. The following example creates a backend address pool named *myBackEndPool*:

```
$backendPool = New-AzureRmLoadBalancerBackendAddressPoolConfig -Name myBackEndPool
```

Now, create the load balancer with [New-AzureRmLoadBalancer](#). The following example creates a load balancer named *myLoadBalancer* using the frontend and backend IP pools created in the preceding steps:

```
$lb = New-AzureRmLoadBalancer `  
-ResourceGroupName myResourceGroupLoadBalancer `  
-Name myLoadBalancer `  
-Location EastUS `  
-FrontendIpConfiguration $frontendIP `  
-BackendAddressPool $backendPool
```

Create a health probe

To allow the load balancer to monitor the status of your app, you use a health probe. The health probe dynamically adds or removes VMs from the load balancer rotation based on their response to health checks. By default, a VM is removed from the load balancer distribution after two consecutive failures at 15-second intervals. You create a health probe based on a protocol or a specific health check page for your app.

The following example creates a TCP probe. You can also create custom HTTP probes for more fine grained health checks. When using a custom HTTP probe, you must create the health check page, such as *healthcheck.aspx*. The probe must return an **HTTP 200 OK** response for the load balancer to keep the host in rotation.

To create a TCP health probe, you use [Add-AzureRmLoadBalancerProbeConfig](#). The following example creates a health probe named *myHealthProbe* that monitors each VM on *TCP port 80*:

```
Add-AzureRmLoadBalancerProbeConfig `  
-Name myHealthProbe `  
-LoadBalancer $lb `  
-Protocol tcp `  
-Port 80 `  
-IntervalInSeconds 15 `  
-ProbeCount 2
```

To apply the health probe, update the load balancer with [Set-AzureRmLoadBalancer](#):

```
Set-AzureRmLoadBalancer -LoadBalancer $lb
```

Create a load balancer rule

A load balancer rule is used to define how traffic is distributed to the VMs. You define the front-end IP configuration for the incoming traffic and the back-end IP pool to receive the traffic, along with the required source and destination port. To make sure only healthy VMs receive traffic, you also define the health probe to use.

Create a load balancer rule with [Add-AzureRmLoadBalancerRuleConfig](#). The following example creates a load balancer rule named *myLoadBalancerRule* and balances traffic on TCP port 80:

```
$probe = Get-AzureRmLoadBalancerProbeConfig -LoadBalancer $lb -Name myHealthProbe

Add-AzureRmLoadBalancerRuleConfig `-
    -Name myLoadBalancerRule `-
    -LoadBalancer $lb `-
    -FrontendIpConfiguration $lb.FrontendIpConfigurations[0] `-
    -BackendAddressPool $lb.BackendAddressPools[0] `-
    -Protocol Tcp `-
    -FrontendPort 80 `-
    -BackendPort 80 `-
    -Probe $probe
```

Update the load balancer with [Set-AzureRmLoadBalancer](#):

```
Set-AzureRmLoadBalancer -LoadBalancer $lb
```

Configure virtual network

Before you deploy some VMs and can test your balancer, create the supporting virtual network resources. For more information about virtual networks, see the [Manage Azure Virtual Networks](#) tutorial.

Create network resources

Create a virtual network with [New-AzureRmVirtualNetwork](#). The following example creates a virtual network named *myVnet* with *mySubnet*:

```
# Create subnet config
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig `-
    -Name mySubnet `-
    -AddressPrefix 192.168.1.0/24

# Create the virtual network
$vnet = New-AzureRmVirtualNetwork `-
    -ResourceGroupName myResourceGroupLoadBalancer `-
    -Location EastUS `-
    -Name myVnet `-
    -AddressPrefix 192.168.0.0/16 `-
    -Subnet $subnetConfig
```

Create a network security group rule with [New-AzureRmNetworkSecurityRuleConfig](#), then create a network security group with [New-AzureRmNetworkSecurityGroup](#). Add the network security group to the subnet with [Set-AzureRmVirtualNetworkSubnetConfig](#) and then update the virtual network with [Set-AzureRmVirtualNetwork](#).

The following example creates a network security group rule named *myNetworkSecurityGroup* and applies it to *mySubnet*:

```

# Create security rule config
$nsgRule = New-AzureRmNetworkSecurityRuleConfig ` 
    -Name myNetworkSecurityGroupRule ` 
    -Protocol Tcp ` 
    -Direction Inbound ` 
    -Priority 1001 ` 
    -SourceAddressPrefix * ` 
    -SourcePortRange * ` 
    -DestinationAddressPrefix * ` 
    -DestinationPortRange 80 ` 
    -Access Allow

# Create the network security group
$nsg = New-AzureRmNetworkSecurityGroup ` 
    -ResourceGroupName myResourceGroupLoadBalancer ` 
    -Location EastUS ` 
    -Name myNetworkSecurityGroup ` 
    -SecurityRules $nsgRule

# Apply the network security group to a subnet
Set-AzureRmVirtualNetworkSubnetConfig ` 
    -VirtualNetwork $vnet ` 
    -Name mySubnet ` 
    -NetworkSecurityGroup $nsg ` 
    -AddressPrefix 192.168.1.0/24

# Update the virtual network
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet

```

Virtual NICs are created with [New-AzureRmNetworkInterface](#). The following example creates three virtual NICs. (One virtual NIC for each VM you create for your app in the following steps). You can create additional virtual NICs and VMs at any time and add them to the load balancer:

```

for ($i=1; $i -le 3; $i++)
{
    New-AzureRmNetworkInterface ` 
        -ResourceGroupName myResourceGroupLoadBalancer ` 
        -Name myNic$i ` 
        -Location EastUS ` 
        -Subnet $vnet.Subnets[0] ` 
        -LoadBalancerBackendAddressPool $lb.BackendAddressPools[0]
}

```

Create virtual machines

To improve the high availability of your app, place your VMs in an availability set.

Create an availability set with [New-AzureRmAvailabilitySet](#). The following example creates an availability set named *myAvailabilitySet*:

```

$availabilitySet = New-AzureRmAvailabilitySet ` 
    -ResourceGroupName myResourceGroupLoadBalancer ` 
    -Name myAvailabilitySet ` 
    -Location EastUS ` 
    -Managed ` 
    -PlatformFaultDomainCount 3 ` 
    -PlatformUpdateDomainCount 2

```

Set an administrator username and password for the VMs with [Get-Credential](#):

```
$cred = Get-Credential
```

Now you can create the VMs with [New-AzureRmVM](#). The following example creates three VMs:

```
for ($i=1; $i -le 3; $i++) {
    $vm = New-AzureRmVMConfig ` 
        -VMName myVM$i ` 
        -VMSize Standard_D1 ` 
        -AvailabilitySetId $availabilitySet.Id
    $vm = Set-AzureRmVMOperatingSystem ` 
        -VM $vm ` 
        -Windows ` 
        -ComputerName myVM$i ` 
        -Credential $cred ` 
        -ProvisionVMAgent ` 
        -EnableAutoUpdate
    $vm = Set-AzureRmVMSourceImage ` 
        -VM $vm ` 
        -PublisherName MicrosoftWindowsServer ` 
        -Offer WindowsServer ` 
        -Skus 2016-Datacenter ` 
        -Version latest
    $vm = Set-AzureRmVMOSDisk ` 
        -VM $vm ` 
        -Name myOsDisk$i ` 
        -DiskSizeInGB 128 ` 
        -CreateOption FromImage ` 
        -Caching ReadWrite
    $nic = Get-AzureRmNetworkInterface ` 
        -ResourceGroupName myResourceGroupLoadBalancer ` 
        -Name myNic$i
    $vm = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id
    New-AzureRmVM ` 
        -ResourceGroupName myResourceGroupLoadBalancer ` 
        -Location EastUS ` 
        -VM $vm
}
```

It takes a few minutes to create and configure all three VMs.

Install IIS with Custom Script Extension

In a previous tutorial on [How to customize a Windows virtual machine](#), you learned how to automate VM customization with the Custom Script Extension for Windows. You can use the same approach to install and configure IIS on your VMs.

Use [Set-AzureRmVMExtension](#) to install the Custom Script Extension. The extension runs

```
powershell Add-WindowsFeature Web-Server
```

 to install the IIS webserver and then updates the *Default.htm* page to show the hostname of the VM:

```

for ($i=1; $i -le 3; $i++)
{
    Set-AzureRmVMExtension ` 
        -ResourceGroupName myResourceGroupLoadBalancer ` 
        -ExtensionName IIS ` 
        -VMName myVM$i ` 
        -Publisher Microsoft.Compute ` 
        -ExtensionType CustomScriptExtension ` 
        -TypeHandlerVersion 1.4 ` 
        -SettingString '{"commandToExecute":"powershell Add-WindowsFeature Web-Server; powershell Add-Content -Path \"C:\\inetpub\\wwwroot\\Default.htm\" -Value $($env:computername)"}' ` 
        -Location EastUS
}

```

Test load balancer

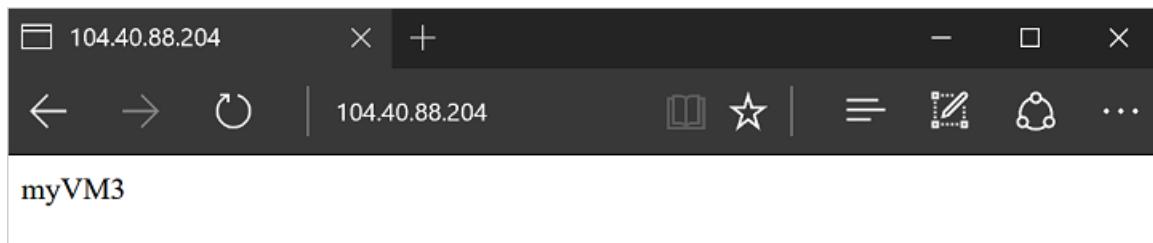
Obtain the public IP address of your load balancer with [Get-AzureRmPublicIPAddress](#). The following example obtains the IP address for *myPublicIP* created earlier:

```

Get-AzureRmPublicIPAddress ` 
    -ResourceGroupName myResourceGroupLoadBalancer ` 
    -Name myPublicIP | select IpAddress

```

You can then enter the public IP address in to a web browser. The website is displayed, including the hostname of the VM that the load balancer distributed traffic to as in the following example:



To see the load balancer distribute traffic across all three VMs running your app, you can force-refresh your web browser.

Add and remove VMs

You may need to perform maintenance on the VMs running your app, such as installing OS updates. To deal with increased traffic to your app, you may need to add additional VMs. This section shows you how to remove or add a VM from the load balancer.

Remove a VM from the load balancer

Get the network interface card with [Get-AzureRmNetworkInterface](#), then set the *LoadBalancerBackendAddressPools* property of the virtual NIC to *\$null*. Finally, update the virtual NIC.:

```

$nic = Get-AzureRmNetworkInterface ` 
    -ResourceGroupName myResourceGroupLoadBalancer ` 
    -Name myNic2
$nic.IpConfigurations[0].LoadBalancerBackendAddressPools=$null
Set-AzureRmNetworkInterface -NetworkInterface $nic

```

To see the load balancer distribute traffic across the remaining two VMs running your app you can force-refresh your web browser. You can now perform maintenance on the VM, such as installing OS updates or performing a VM reboot.

Add a VM to the load balancer

After performing VM maintenance, or if you need to expand capacity, set the *LoadBalancerBackendAddressPools* property of the virtual NIC to the *BackendAddressPool* from [Get-AzureRMLoadBalancer](#):

Get the load balancer:

```
$lb = Get-AzureRMLoadBalancer `  
    -ResourceGroupName myResourceGroupLoadBalancer `  
    -Name myLoadBalancer  
$nic.IpConfigurations[0].LoadBalancerBackendAddressPools=$lb.BackendAddressPools[0]  
Set-AzureRmNetworkInterface -NetworkInterface $nic
```

Next steps

In this tutorial, you created a load balancer and attached VMs to it. You learned how to:

- Create an Azure load balancer
- Create a load balancer health probe
- Create load balancer traffic rules
- Use the Custom Script Extension to create a basic IIS site
- Create virtual machines and attach to a load balancer
- View a load balancer in action
- Add and remove VMs from a load balancer

Advance to the next tutorial to learn how to manage VM networking.

[Manage VMs and virtual networks](#)

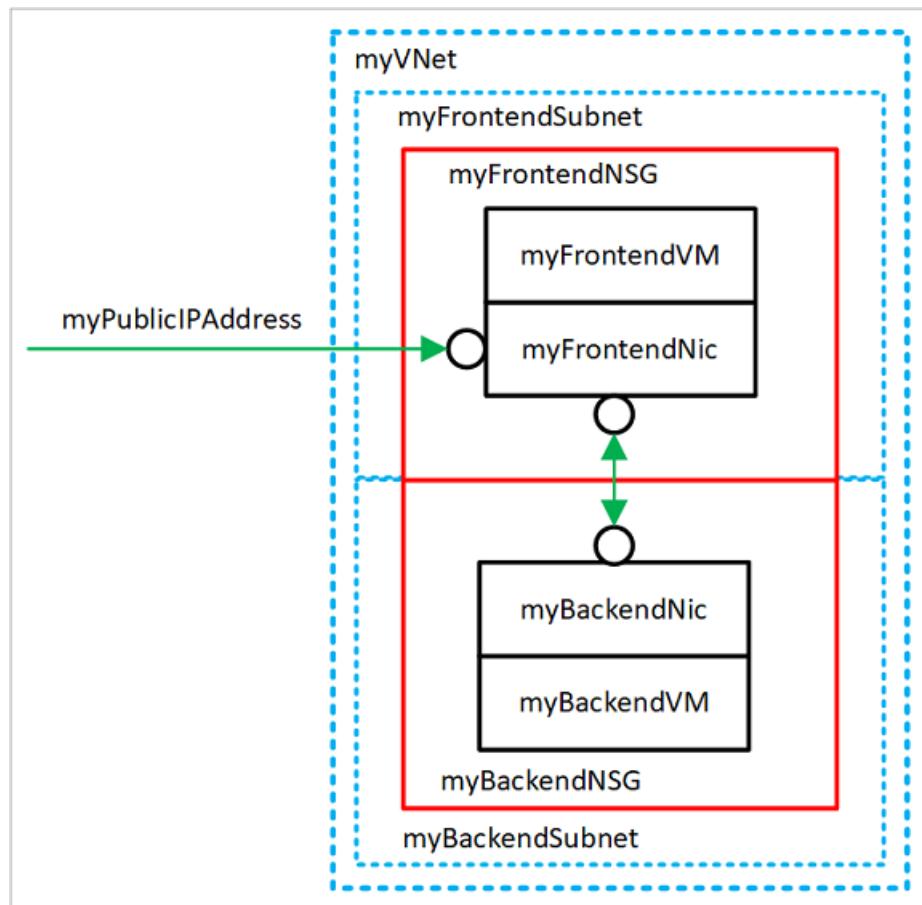
Manage Azure Virtual Networks and Windows Virtual Machines with Azure PowerShell

10/25/2017 • 7 min to read • [Edit Online](#)

Azure virtual machines use Azure networking for internal and external network communication. This tutorial walks through deploying two virtual machines and configuring Azure networking for these VMs. The examples in this tutorial assume that the VMs are hosting a web application with a database back-end, however an application is not deployed in the tutorial. In this tutorial, you learn how to:

- Create a virtual network and subnet
- Create a public IP address
- Create a front-end VM
- Secure network traffic
- Create back-end VM

While completing this tutorial, you can see these resources created:



- *myVNet* - The virtual network that the VMs use to communicate with each other and the internet.
- *myFrontendSubnet* - The subnet in *myVNet* used by the front-end resources.
- *myPublicIPAddress* - The public IP address used to access *myFrontendVM* from the internet.
- *myFrontendNic* - The network interface used by *myFrontendVM* to communicate with *myBackendVM*.
- *myFrontendVM* - The VM used to communicate between the internet and *myBackendVM*.
- *myBackendNSG* - The network security group that controls communication between the *myFrontendVM* and *myBackendVM*.

- *myBackendSubnet* - The subnet associated with *myBackendNSG* and used by the back-end resources.
- *myBackendNic* - The network interface used by *myBackendVM* to communicate with *myFrontendVM*.
- *myBackendVM* - The VM that uses port 1433 to communicate with *myFrontendVM*.

This tutorial requires the Azure PowerShell module version 3.6 or later. To find the version, run

```
Get-Module -ListAvailable AzureRM
```

VM networking overview

Azure virtual networks enable secure network connections between virtual machines, the internet, and other Azure services such as Azure SQL database. Virtual networks are broken down into logical segments called subnets. Subnets are used to control network flow, and as a security boundary. When deploying a VM, it generally includes a virtual network interface, which is attached to a subnet.

Create a virtual network and subnet

For this tutorial, a single virtual network is created with two subnets. A front-end subnet for hosting a web application, and a back-end subnet for hosting a database server.

Before you can create a virtual network, create a resource group using [New-AzureRmResourceGroup](#). The following example creates a resource group named *myRGNetwork* in the *EastUS* location:

```
New-AzureRmResourceGroup -ResourceGroupName myRGNetwork -Location EastUS
```

Create subnet configurations

Create a subnet configuration named *myFrontendSubnet* using [New-AzureRmVirtualNetworkSubnetConfig](#):

```
$frontendSubnet = New-AzureRmVirtualNetworkSubnetConfig ` 
    -Name myFrontendSubnet ` 
    -AddressPrefix 10.0.0.0/24
```

And, create a subnet configuration named *myBackendSubnet*:

```
$backendSubnet = New-AzureRmVirtualNetworkSubnetConfig ` 
    -Name myBackendSubnet ` 
    -AddressPrefix 10.0.1.0/24
```

Create virtual network

Create a VNET named *myVNet* using *myFrontendSubnet* and *myBackendSubnet* using [New-AzureRmVirtualNetwork](#):

```
$vnet = New-AzureRmVirtualNetwork ` 
    -ResourceGroupName myRGNetwork ` 
    -Location EastUS ` 
    -Name myVNet ` 
    -AddressPrefix 10.0.0.0/16 ` 
    -Subnet $frontendSubnet, $backendSubnet
```

At this point, a network has been created and segmented into two subnets, one for front-end services, and another for back-end services. In the next section, virtual machines are created and connected to these subnets.

Create a public IP address

A public IP address allows Azure resources to be accessible on the internet. The allocation method of the public IP address can be configured as dynamic or static. By default, a public IP address is dynamically allocated. Dynamic IP addresses are released when a VM is deallocated. This behavior causes the IP address to change during any operation that includes a VM deallocation.

The allocation method can be set to static, which ensures that the IP address remains assigned to a VM, even during a deallocated state. When using a statically allocated IP address, the IP address itself cannot be specified. Instead, it is allocated from a pool of available addresses.

Create a public IP address named *myPublicIPAddress* using [New-AzureRmPublicIpAddress](#):

```
$pip = New-AzureRmPublicIpAddress `  
    -ResourceGroupName myRGNetwork `  
    -Location EastUS `  
    -AllocationMethod Dynamic `  
    -Name myPublicIPAddress
```

You could change the `-AllocationMethod` parameter to `static` to assign a static public IP address.

Create a front-end VM

For a VM to communicate in a virtual network, it needs a virtual network interface (NIC). Create a NIC using [New-AzureRmNetworkInterface](#):

```
$frontendNic = New-AzureRmNetworkInterface `  
    -ResourceGroupName myRGNetwork `  
    -Location EastUS `  
    -Name myFrontendNic `  
    -SubnetId $vnet.Subnets[0].Id `  
    -PublicIpAddressId $pip.Id
```

Set the username and password needed for the administrator account on the VM using [Get-Credential](#). You use these credentials to connect to the VM in additional steps:

```
$cred = Get-Credential
```

Create the VMs using [New-AzureRmVMConfig](#), [Set-AzureRmVMOperatingSystem](#), [Set-AzureRmVMSourceImage](#), [Set-AzureRmVMOSDisk](#), [Add-AzureRmVMNetworkInterface](#), and [New-AzureRmVM](#):

```

$frontendVM = New-AzureRmVMConfig ` 
    -VMName myFrontendVM ` 
    -VMSize Standard_D1
$frontendVM = Set-AzureRmVMOperatingSystem ` 
    -VM $frontendVM ` 
    -Windows ` 
    -ComputerName myFrontendVM ` 
    -Credential $cred ` 
    -ProvisionVMAgent ` 
    -EnableAutoUpdate
$frontendVM = Set-AzureRmVMSourceImage ` 
    -VM $frontendVM ` 
    -PublisherName MicrosoftWindowsServer ` 
    -Offer WindowsServer ` 
    -Skus 2016-Datacenter ` 
    -Version latest
$frontendVM = Set-AzureRmVMOSDisk ` 
    -VM $frontendVM ` 
    -Name myFrontendOSDisk ` 
    -DiskSizeInGB 128 ` 
    -CreateOption FromImage ` 
    -Caching ReadWrite
$frontendVM = Add-AzureRmVMNetworkInterface ` 
    -VM $frontendVM ` 
    -Id $frontendNic.Id
New-AzureRmVM ` 
    -ResourceGroupName myRGNetwork ` 
    -Location EastUS ` 
    -VM $frontendVM

```

Secure network traffic

A network security group (NSG) contains a list of security rules that allow or deny network traffic to resources connected to Azure Virtual Networks (VNet). NSGs can be associated to subnets or individual network interfaces. When an NSG is associated with a network interface, it applies only the associated VM. When an NSG is associated to a subnet, the rules apply to all resources connected to the subnet.

Network security group rules

NSG rules define networking ports over which traffic is allowed or denied. The rules can include source and destination IP address ranges so that traffic is controlled between specific systems or subnets. NSG rules also include a priority (between 1—and 4096). Rules are evaluated in the order of priority. A rule with a priority of 100 is evaluated before a rule with priority 200.

All NSGs contain a set of default rules. The default rules cannot be deleted, but because they are assigned the lowest priority, they can be overridden by the rules that you create.

- **Virtual network** - Traffic originating and ending in a virtual network is allowed both in inbound and outbound directions.
- **Internet** - Outbound traffic is allowed, but inbound traffic is blocked.
- **Load balancer** - Allow Azure's load balancer to probe the health of your VMs and role instances. If you are not using a load balanced set, you can override this rule.

Create network security groups

Create an inbound rule named *myFrontendNSGRule* to allow incoming web traffic on *myFrontendVM* using [New-AzureRmNetworkSecurityRuleConfig](#):

```
$nsgFrontendRule = New-AzureRmNetworkSecurityRuleConfig ` 
-Name myFrontendNSGRule ` 
-Protocol Tcp ` 
-Direction Inbound ` 
-Priority 200 ` 
-SourceAddressPrefix * ` 
-SourcePortRange * ` 
-DestinationAddressPrefix * ` 
-DestinationPortRange 80 ` 
-Access Allow
```

You can limit internal traffic to *myBackendVM* from only *myFrontendVM* by creating an NSG for the back-end subnet. The following example creates an NSG rule named *myBackendNSGRule*:

```
$nsgBackendRule = New-AzureRmNetworkSecurityRuleConfig ` 
-Name myBackendNSGRule ` 
-Protocol Tcp ` 
-Direction Inbound ` 
-Priority 100 ` 
-SourceAddressPrefix 10.0.0.0/24 ` 
-SourcePortRange * ` 
-DestinationAddressPrefix * ` 
-DestinationPortRange 1433 ` 
-Access Allow
```

Add a network security group named *myFrontendNSG* using [New-AzureRmNetworkSecurityGroup](#):

```
$nsgFrontend = New-AzureRmNetworkSecurityGroup ` 
-ResourceGroupName myRGNetwork ` 
-Location EastUS ` 
-Name myFrontendNSG ` 
-SecurityRules $nsgFrontendRule
```

Now, add a network security group named *myBackendNSG* using [New-AzureRmNetworkSecurityGroup](#):

```
$nsgBackend = New-AzureRmNetworkSecurityGroup ` 
-ResourceGroupName myRGNetwork ` 
-Location EastUS ` 
-Name myBackendNSG ` 
-SecurityRules $nsgBackendRule
```

Add the network security groups to the subnets:

```
$vnet = Get-AzureRmVirtualNetwork ` 
-ResourceGroupName myRGNetwork ` 
-Name myVNet
$frontendSubnet = $vnet.Subnets[0]
$backendSubnet = $vnet.Subnets[1]
$frontendSubnetConfig = Set-AzureRmVirtualNetworkSubnetConfig ` 
-VirtualNetwork $vnet ` 
-Name myFrontendSubnet ` 
-AddressPrefix $frontendSubnet.AddressPrefix ` 
-NetworkSecurityGroup $nsgFrontend
$backendSubnetConfig = Set-AzureRmVirtualNetworkSubnetConfig ` 
-VirtualNetwork $vnet ` 
-Name myBackendSubnet ` 
-AddressPrefix $backendSubnet.AddressPrefix ` 
-NetworkSecurityGroup $nsgBackend
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet
```

Create a back-end VM

The easiest way to create the back-end VM for this tutorial is by using a SQL Server image. This tutorial only creates the VM with the database server, but doesn't provide information about accessing the database.

Create *myBackendNic*:

```
$backendNic = New-AzureRmNetworkInterface `  
    -ResourceGroupName myRGNetwork `  
    -Location EastUS `  
    -Name myBackendNic `  
    -SubnetId $vnet.Subnets[1].Id
```

Set the username and password needed for the administrator account on the VM with *Get-Credential*:

```
$cred = Get-Credential
```

Create *myBackendVM*:

```
$backendVM = New-AzureRmVMConfig `  
    -VMName myBackendVM `  
    -VMSize Standard_D1  
$backendVM = Set-AzureRmVMOperatingSystem `  
    -VM $backendVM `  
    -Windows `  
    -ComputerName myBackendVM `  
    -Credential $cred `  
    -ProvisionVMAgent `  
    -EnableAutoUpdate  
$backendVM = Set-AzureRmVMSourceImage `  
    -VM $backendVM `  
    -PublisherName MicrosoftSQLServer `  
    -Offer SQL2016SP1-WS2016 `  
    -Skus Enterprise `  
    -Version latest  
$backendVM = Set-AzureRmVMOSDisk `  
    -VM $backendVM `  
    -Name myBackendOSDisk `  
    -DiskSizeInGB 128 `  
    -CreateOption FromImage `  
    -Caching ReadWrite  
$backendVM = Add-AzureRmVMNetworkInterface `  
    -VM $backendVM `  
    -Id $backendNic.Id  
New-AzureRmVM `  
    -ResourceGroupName myRGNetwork `  
    -Location EastUS `  
    -VM $backendVM
```

The image that is used has SQL Server installed, but is not used in this tutorial. It is included to show you how you can configure a VM to handle web traffic and a VM to handle database management.

Next steps

In this tutorial, you created and secured Azure networks as related to virtual machines.

- Create a virtual network and subnet
- Create a public IP address
- Create a front-end VM

- Secure network traffic
- Create a back-end VM

Advance to the next tutorial to learn about monitoring securing data on virtual machines using Azure backup.

[Back up Windows virtual machines in Azure](#)

Back up Windows virtual machines in Azure

12/12/2017 • 4 min to read • [Edit Online](#)

You can protect your data by taking backups at regular intervals. Azure Backup creates recovery points that are stored in geo-redundant recovery vaults. When you restore from a recovery point, you can restore the whole VM or just specific files. This article explains how to restore a single file to a VM running Windows Server and IIS. If you don't already have a VM to use, you can create one using the [Windows quickstart](#). In this tutorial you learn how to:

- Create a backup of a VM
- Schedule a daily backup
- Restore a file from a backup

Backup overview

When the Azure Backup service initiates a backup job, it triggers the backup extension to take a point-in-time snapshot. The Azure Backup service uses the *VMSnapshot* extension. The extension is installed during the first VM backup if the VM is running. If the VM is not running, the Backup service takes a snapshot of the underlying storage (since no application writes occur while the VM is stopped).

When taking a snapshot of Windows VMs, the Backup service coordinates with the Volume Shadow Copy Service (VSS) to get a consistent snapshot of the virtual machine's disks. Once the Azure Backup service takes the snapshot, the data is transferred to the vault. To maximize efficiency, the service identifies and transfers only the blocks of data that have changed since the previous backup.

When the data transfer is complete, the snapshot is removed and a recovery point is created.

Create a backup

Create a simple scheduled daily backup to a Recovery Services Vault.

1. Sign in to the [Azure portal](#).
2. In the menu on the left, select **Virtual machines**.
3. From the list, select a VM to back up.
4. On the VM blade, in the **Settings** section, click **Backup**. The **Enable backup** blade opens.
5. In **Recovery Services vault**, click **Create new** and provide the name for the new vault. A new vault is created in the same Resource Group and location as the virtual machine.
6. Click **Backup policy**. For this example, keep the defaults and click **OK**.
7. On the **Enable backup** blade, click **Enable Backup**. This creates a daily backup based on the default schedule.
8. To create an initial recovery point, on the **Backup** blade click **Backup now**.
9. On the **Backup Now** blade, click the calendar icon, use the calendar control to select the last day this recovery point is retained, and click **Backup**.
10. In the **Backup** blade for your VM, you will see the number of recovery points that are complete.

Restore points	
Last 30 days	1
Last 7 days	1

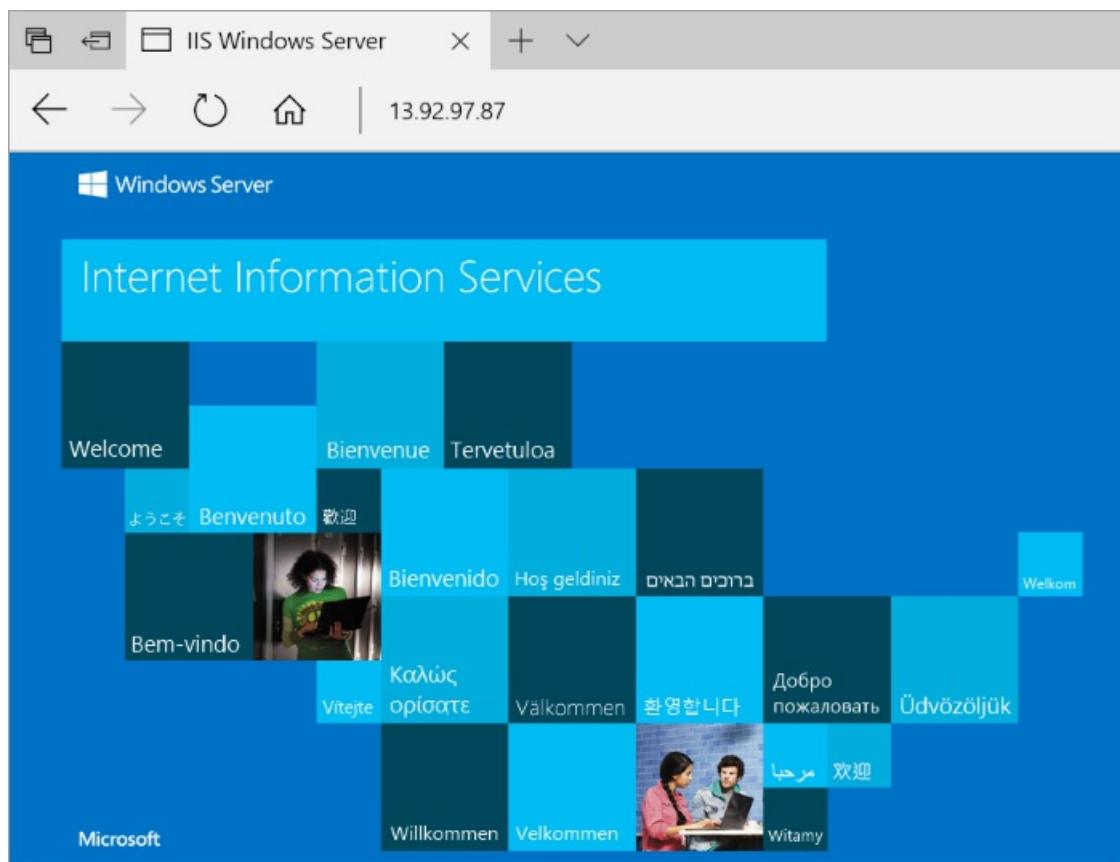
The first backup takes about 20 minutes. Proceed to the next part of this tutorial after your backup is finished.

Recover a file

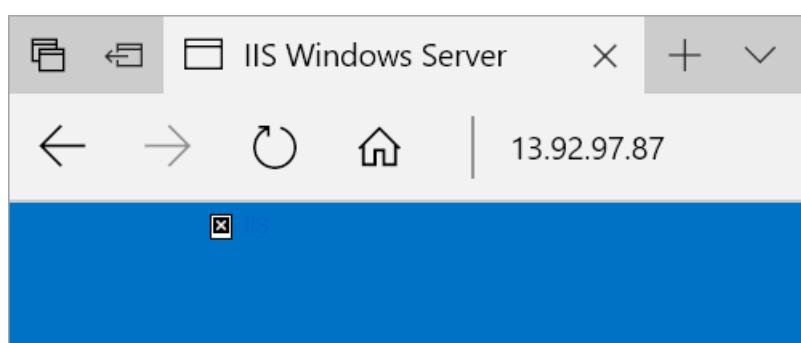
If you accidentally delete or make changes to a file, you can use File Recovery to recover the file from your backup vault. File Recovery uses a script that runs on the VM, to mount the recovery point as local drive. These drives will remain mounted for 12 hours so that you can copy files from the recovery point and restore them to the VM.

In this example, we show how to recover the image file that is used in the default web page for IIS.

1. Open a browser and connect to the IP address of the VM to show the default IIS page.



2. Connect to the VM.
3. On the VM, open **File Explorer** and navigate to **\inetpub\wwwroot** and delete the file **iisstart.png**.
4. On your local computer, refresh the browser to see that the image on the default IIS page is gone.



5. On your local computer, open a new tab and go to the [Azure portal](#).
6. In the menu on the left, select **Virtual machines** and select the VM from the list.
7. On the VM blade, in the **Settings** section, click **Backup**. The **Backup** blade opens.
8. In the menu at the top of the blade, select **File Recovery**. The **File Recovery** blade opens.
9. In **Step 1: Select recovery point**, select a recovery point from the drop-down.

10. In **Step 2: Download script to browse and recover files**, click the **Download Executable** button. Save the file to your **Downloads** folder.
11. On your local computer, open **File Explorer** and navigate to your **Downloads** folder and copy the downloaded .exe file. The filename will be prefixed by your VM name.
12. On your VM (over the RDP connection) paste the .exe file to the Desktop of your VM.
13. Navigate to the desktop of your VM and double-click on the .exe. This will launch a command prompt and then mount the recovery point as a file share that you can access. When it is finished creating the share, type **q** to close the command prompt.
14. On your VM, open **File Explorer** and navigate to the drive letter that was used for the file share.
15. Navigate to \inetpub\wwwroot and copy **iisstart.png** from the file share and paste it into \inetpub\wwwroot. For example, copy F:\inetpub\wwwroot\iisstart.png and paste it into c:\inetpub\wwwroot to recover the file.
16. On your local computer, open the browser tab where you are connected to the IP address of the VM showing the IIS default page. Press CTRL + F5 to refresh the browser page. You should now see that the image has been restored.
17. On your local computer, go back to the browser tab for the Azure portal and in **Step 3: Unmount the disks after recovery** click the **Unmount Disks** button. If you forget to do this step, the connection to the mountpoint is automatically close after 12 hours. After those 12 hours, you need to download a new script to create a new mountpoint.

Next steps

In this tutorial, you learned how to:

- Create a backup of a VM
- Schedule a daily backup
- Restore a file from a backup

Advance to the next tutorial to learn about monitoring virtual machines.

[Monitor virtual machines](#)

Monitor and update a Windows Virtual Machine with Azure PowerShell

9/25/2017 • 8 min to read • [Edit Online](#)

Azure monitoring uses agents to collect boot and performance data from Azure VMs, store this data in Azure storage, and make it accessible through portal, the Azure PowerShell module, and the Azure CLI. Update management allows you to manage updates and patches for your Azure Windows VMs.

In this tutorial, you learn how to:

- Enable boot diagnostics on a VM
- View boot diagnostics
- View VM host metrics
- Install the diagnostics extension
- View VM metrics
- Create an alert
- Manage Windows updates
- Set up advanced monitoring

This tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

To complete the example in this tutorial, you must have an existing virtual machine. If needed, this [script sample](#) can create one for you. When working through the tutorial, replace the resource group, VM name, and location where needed.

View boot diagnostics

As Windows virtual machines boot up, the boot diagnostic agent captures screen output that can be used for troubleshooting purpose. This capability is enabled by default. The captured screen shots are stored in an Azure storage account, which is also created by default.

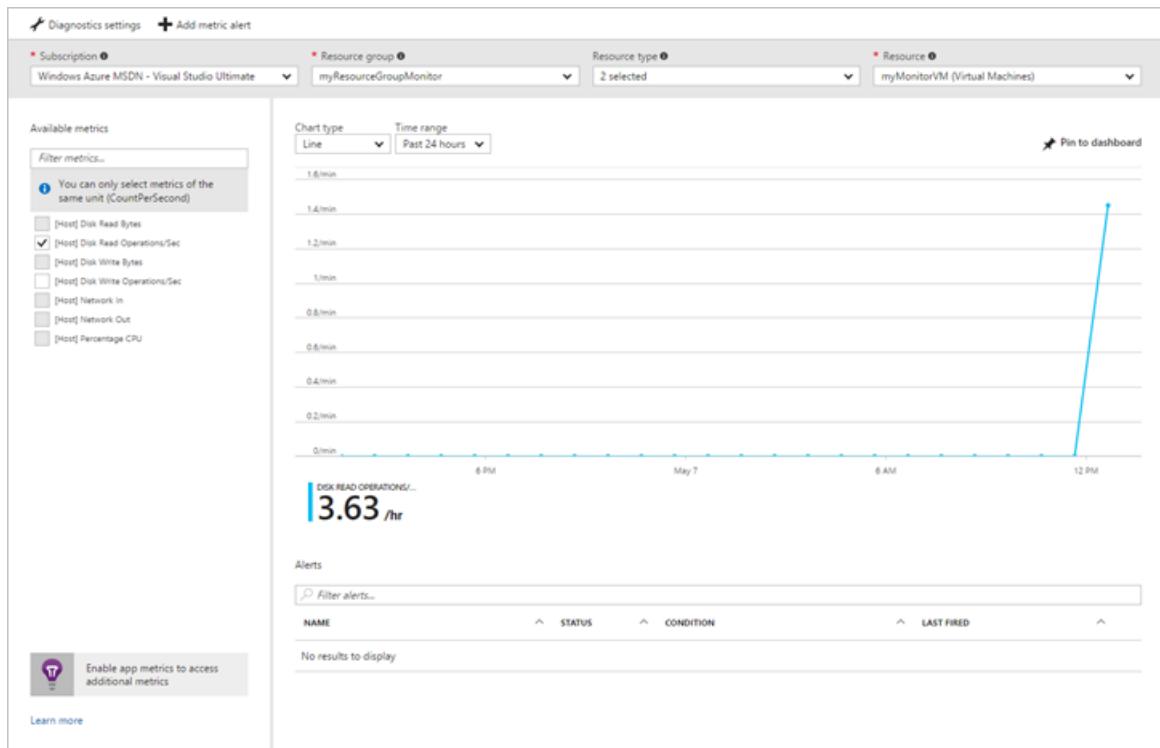
You can get the boot diagnostic data with the `Get-AzureRmVMBootDiagnosticsData` command. In the following example, boot diagnostics are downloaded to the root of the *c:* drive.

```
Get-AzureRmVMBootDiagnosticsData -ResourceGroupName myResourceGroup -Name myVM -Windows -LocalPath "c:\\"
```

View host metrics

A Windows VM has a dedicated Host VM in Azure that it interacts with. Metrics are automatically collected for the Host and can be viewed in the Azure portal.

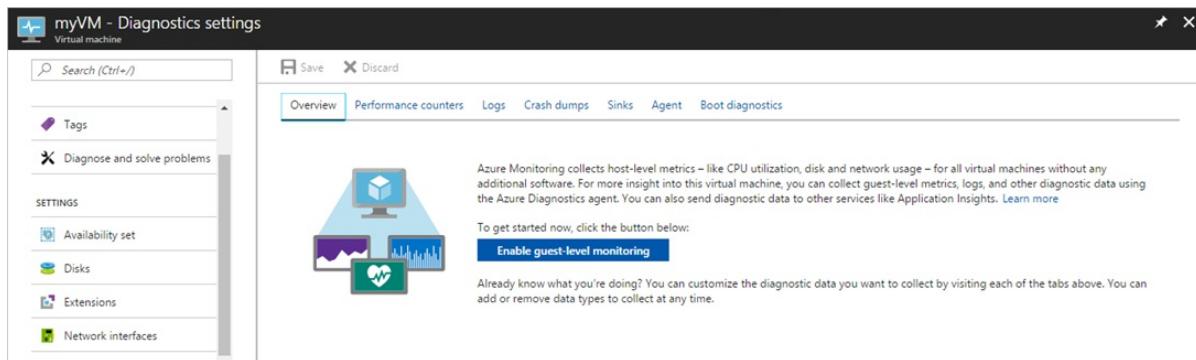
1. In the Azure portal, click **Resource Groups**, select **myResourceGroup**, and then select **myVM** in the resource list.
2. Click **Metrics** on the VM blade, and then select any of the Host metrics under **Available metrics** to see how the Host VM is performing.



Install diagnostics extension

The basic host metrics are available, but to see more granular and VM-specific metrics, you need to install the Azure diagnostics extension on the VM. The Azure diagnostics extension allows additional monitoring and diagnostics data to be retrieved from the VM. You can view these performance metrics and create alerts based on how the VM performs. The diagnostic extension is installed through the Azure portal as follows:

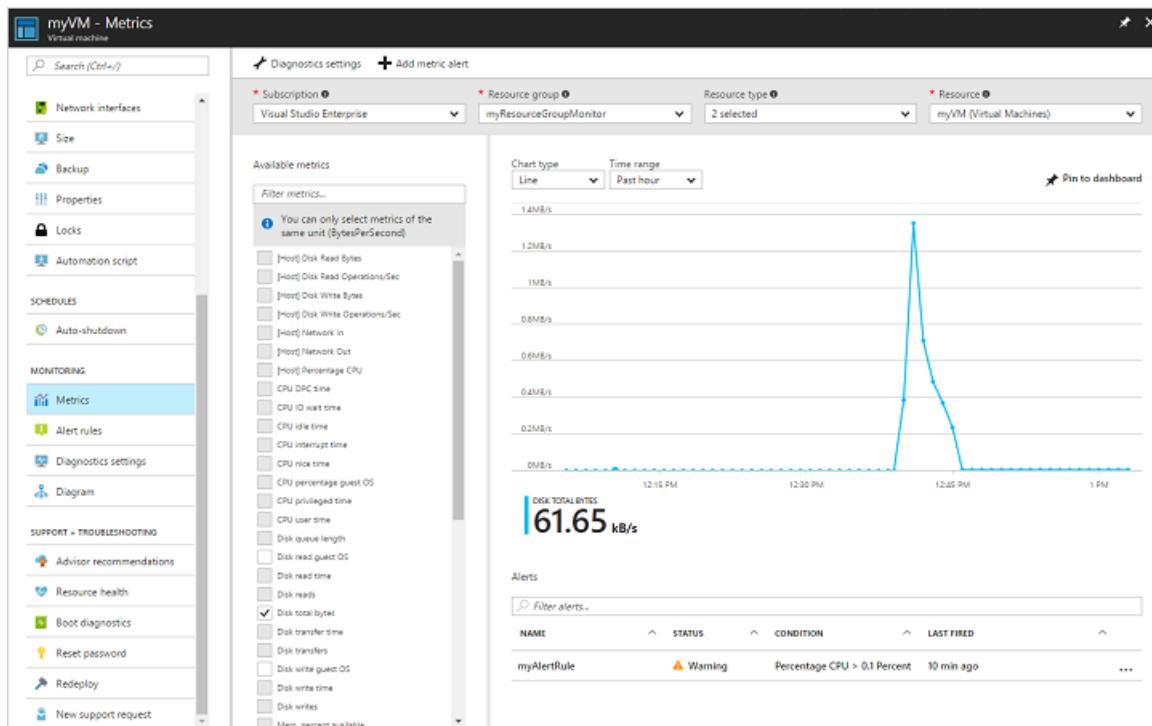
1. In the Azure portal, click **Resource Groups**, select **myResourceGroup**, and then select **myVM** in the resource list.
2. Click **Diagnosis settings**. The list shows that *Boot diagnostics* are already enabled from the previous section. Click the check box for *Basic metrics*.
3. Click the **Enable guest-level monitoring** button.



View VM metrics

You can view the VM metrics in the same way that you viewed the host VM metrics:

1. In the Azure portal, click **Resource Groups**, select **myResourceGroup**, and then select **myVM** in the resource list.
2. To see how the VM is performing, click **Metrics** on the VM blade, and then select any of the diagnostics metrics under **Available metrics**.



Create alerts

You can create alerts based on specific performance metrics. Alerts can be used to notify you when average CPU usage exceeds a certain threshold or available free disk space drops below a certain amount, for example. Alerts are displayed in the Azure portal or can be sent via email. You can also trigger Azure Automation runbooks or Azure Logic Apps in response to alerts being generated.

The following example creates an alert for average CPU usage.

1. In the Azure portal, click **Resource Groups**, select **myResourceGroup**, and then select **myVM** in the resource list.
2. Click **Alert rules** on the VM blade, then click **Add metric alert** across the top of the alerts blade.
3. Provide a **Name** for your alert, such as *myAlertRule*
4. To trigger an alert when CPU percentage exceeds 1.0 for five minutes, leave all the other defaults selected.
5. Optionally, check the box for *Email owners, contributors, and readers* to send email notification. The default action is to present a notification in the portal.
6. Click the **OK** button.

Manage Windows updates

Update management allows you to manage updates and patches for your Azure Windows VMs. Directly from your VM, you can quickly assess the status of available updates, schedule installation of required updates, and review deployment results to verify updates were applied successfully to the VM.

For pricing information, see [Automation pricing for Update management](#)

Enable Update management

Enable Update management for your VM:

1. On the left-hand side of the screen, select **Virtual machines**.
2. From the list, select a VM.
3. On the VM screen, in the **Operations** section, click **Update management**. The **Enable Update Management** screen opens.

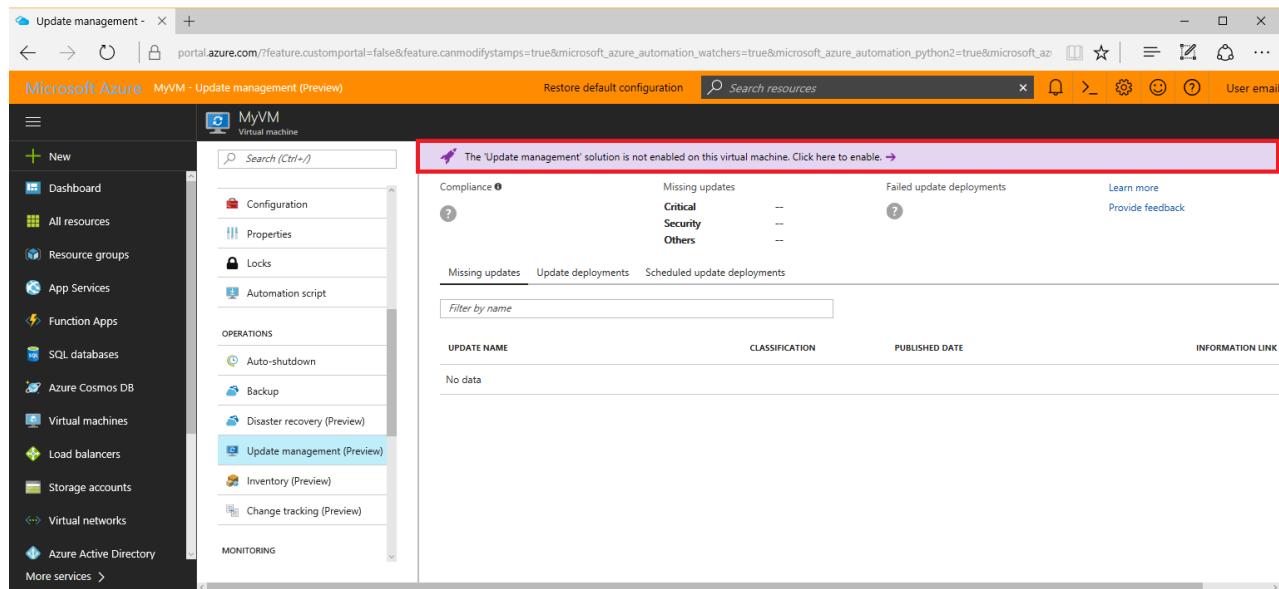
Validation is performed to determine if Update management is enabled for this VM. The validation includes checks

for a Log Analytics workspace and linked Automation account, and if the solution is in the workspace.

A Log Analytics workspace is used to collect data that is generated by features and services such as Update management. The workspace provides a single location to review and analyze data from multiple sources. To perform additional action on VMs that require updates, Azure Automation allows you to run scripts against VMs, such as to download and apply updates.

The validation process also checks to see if the VM is provisioned with the Microsoft Monitoring Agent (MMA) and hybrid worker. This agent is used to communicate with the VM and obtain information about the update status.

If these prerequisites are not met, a banner appears that gives you the option to enable the solution.

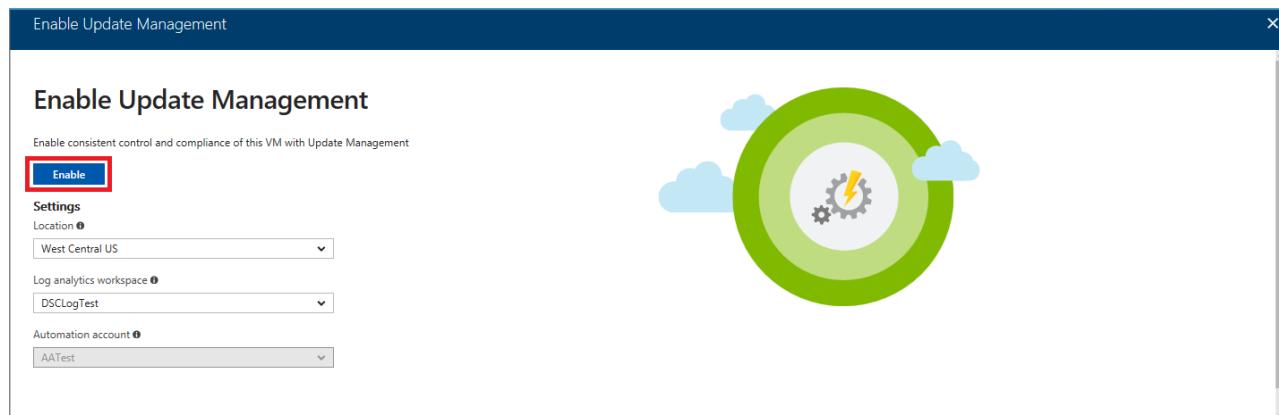


The screenshot shows the Azure portal interface for a virtual machine named 'MyVM'. In the left sidebar, under 'OPERATIONS', the 'Update management (Preview)' option is selected. A red banner at the top right of the main content area states: 'The 'Update management' solution is not enabled on this virtual machine. Click here to enable.' Below the banner, there are sections for 'Compliance', 'Missing updates', and 'Failed update deployments'. The 'Missing updates' section shows categories: Critical (0), Security (0), and Others (0). At the bottom, there is a table with columns for 'UPDATE NAME', 'CLASSIFICATION', 'PUBLISHED DATE', and 'INFORMATION LINK'. The table currently displays 'No data'.

Click the banner to enable the solution. If any of the following prerequisites were found to be missing after the validation, they will be automatically added:

- [Log Analytics](#) workspace
- [Automation](#)
- A [Hybrid runbook worker](#) is enabled on the VM

The **Enable Update Management** screen opens. Configure the settings, and click **Enable**.



The screenshot shows the 'Enable Update Management' dialog. At the top, it says 'Enable consistent control and compliance of this VM with Update Management'. A large blue button labeled 'Enable' is highlighted with a red box. Below the button, there are three dropdown menus for 'Location' (West Central US), 'Log analytics workspace' (DSCLogTest), and 'Automation account' (AAATest). To the right of the form is a decorative graphic of a green circle with a gear and lightning bolt icon, surrounded by clouds.

Enabling the solution can take up to 15 minutes, and during this time you should not close the browser window. After the solution is enabled, information about missing updates on the VM flows to Log Analytics. It can take between 30 minutes and 6 hours for the data to be available for analysis.

View update assessment

After **Update management** is enabled, the **Update management** screen appears. You can see a list of missing updates on the **Missing updates** tab.

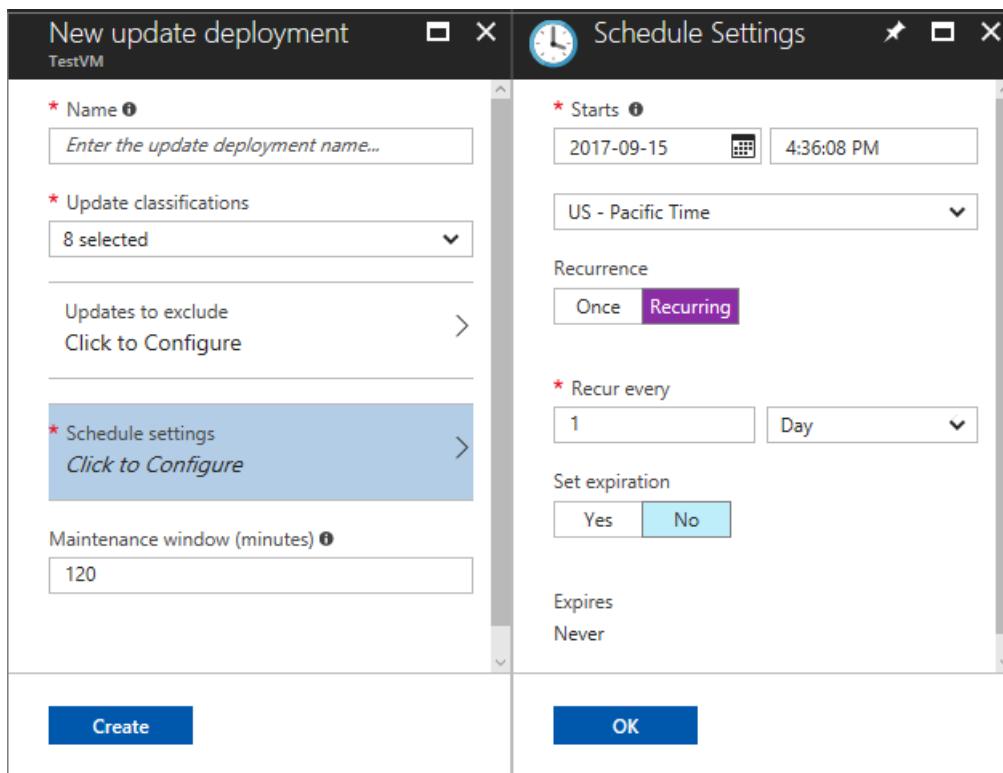
The screenshot shows the Azure portal interface for 'MyVM - Update management (Preview)'. On the left, there's a sidebar with navigation links like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Networking, Disks, and Size. The main content area has a title 'Schedule update deployment' and a 'Compliance' section with a red exclamation mark icon. It shows 'Missing updates' (0 Critical, 1 Security, 2 Others) and 'Failed update deployments' (0 out of 0). Below this are tabs for 'Missing updates', 'Update deployments', and 'Scheduled update deployments', with a 'Filter by name' input field. A table lists three updates: '2017-09 Cumulative Update for Windows Server 2016 for x64-based systems' (Security updates, 9/11/2017, ID 4038782), 'Definition Update for Windows Defender - KB2267602 (Definition updates, 9/11/2017, ID 2267602), and 'Windows Malicious Software Removal Tool for Windows 8, 8.1, 1...' (Update rollups, 9/11/2017, ID 890830).

Schedule an update deployment

To install updates, schedule a deployment that follows your release schedule and service window. You can choose which update types to include in the deployment. For example, you can include critical or security updates and exclude update rollups.

Schedule a new Update Deployment for the VM by clicking **Schedule update deployment** at the top of the **Update management** screen. In the **New update deployment** screen, specify the following information:

- **Name** - Provide a unique name to identify the update deployment.
- **Update classification** - Select the types of software the update deployment included in the deployment. The classification types are:
 - Critical updates
 - Security updates
 - Update rollups
 - Feature packs
 - Service packs
 - Definition updates
 - Tools
 - Updates
- **Schedule settings** - You can either accept the default date and time, which is 30 minutes after current time, or specify a different time. You can also specify whether the deployment occurs once or set up a recurring schedule. Click the Recurring option under Recurrence to set up a recurring schedule.



- **Maintenance window (minutes)** - Specify the period of time you want the update deployment to occur within. This helps ensure changes are performed within your defined service windows.

After you have completed configuring the schedule, click **Create** button and you return to the status dashboard. Notice that the **Scheduled** table shows the deployment schedule you created.

WARNING

For updates that require a reboot, the VM is restarted automatically.

View results of an update deployment

After the scheduled deployment is started, you can see the status for that deployment on the **Update deployments** tab on the **Update management** screen. If it is currently running, its status shows as **In progress**.

After it completes, if successful, it changes to **Succeeded**. If there is a failure with one or more updates in the deployment, the status is **Partially failed**. Click the completed update deployment to see the dashboard for that update deployment.

Name	Classification	Status
2017-09 Cumulative Update for Windows Server 2016 for x64-based Systems (KB4038782)	Security updates	Not attempted
Definition Update for Windows Defender - KB2267602 (Definition 1.251.852.0)	Definition updates	Not attempted
Definition Update for Windows Defender - KB2267602 (Definition 1.251.852.0)	Definition updates	Not attempted
Windows Malicious Software Removal Tool for Windows 8, 8.1, 10 and Windows Server 2012, 2012 R2, 2016 x64 Edition ...	Update rollups	Succeeded
Windows Malicious Software Removal Tool for Windows 8, 8.1, 10 and Windows Server 2012, 2012 R2, 2016 x64 Edition ...	Update rollups	Succeeded

In **Update results** tile is a summary of the total number of updates and deployment results on the VM. In the table to the right is a detailed breakdown of each update and the installation results, which could be one of the following

values:

- **Not attempted** - the update was not installed because there was insufficient time available based on the maintenance window duration defined.
- **Succeeded** - the update succeeded
- **Failed** - the update failed

Click **All logs** to see all log entries that the deployment created.

Click the **Output** tile to see job stream of the runbook responsible for managing the update deployment on the target VM.

Click **Errors** to see detailed information about any errors from the deployment.

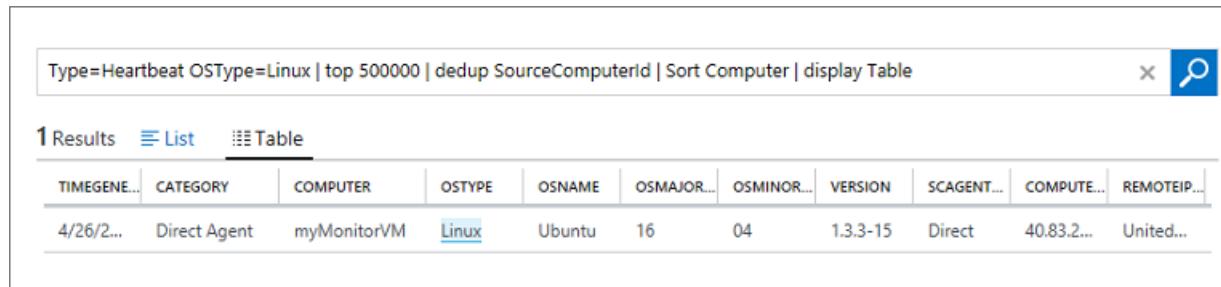
Advanced monitoring

You can do more advanced monitoring of your VM by using [Operations Management Suite](#). If you haven't already done so, you can sign up for a [free trial](#) of Operations Management Suite.

When you have access to the OMS portal, you can find the workspace key and workspace identifier on the Settings blade. Use the [Set-AzureRmVMExtension](#) command to add the OMS extension to the VM. Update the variable values in the below sample to reflect your OMS workspace key and workspace Id.

```
$omsId = "<Replace with your OMS Id>"  
$omsKey = "<Replace with your OMS key>"  
  
Set-AzureRmVMExtension -ResourceGroupName myResourceGroup `  
    -ExtensionName "Microsoft.EnterpriseCloud.Monitoring" `  
    -VMName myVM `  
    -Publisher "Microsoft.EnterpriseCloud.Monitoring" `  
    -ExtensionType "MicrosoftMonitoringAgent" `  
    -TypeHandlerVersion 1.0 `  
    -Settings @{"workspaceId" = $omsId} `  
    -ProtectedSettings @{"workspaceKey" = $omsKey} `  
    -Location eastus
```

After a few minutes, you should see the new VM in the OMS workspace.



The screenshot shows the Azure Monitor Metrics Explorer interface. At the top, there is a search bar with the query: Type=Heartbeat OSType=Linux | top 500000 | dedup SourceComputerId | Sort Computer | display Table. Below the search bar, there is a results summary: 1 Results. There are three tabs: List (selected), Table, and List. A table is displayed with the following columns: TIMEGENE..., CATEGORY, COMPUTER, OSTYPE, OSNAME, OSMajor..., OSMINOR..., VERSION, SCAGENT..., COMPUTE..., and REMOTEIP... . The single result row is: 4/26/2018 Direct Agent myMonitorVM Linux Ubuntu 16 04 1.3.3-15 Direct 40.83.2... United...

Next steps

In this tutorial, you configured and reviewed VMs with Azure Security Center. You learned how to:

- Create a virtual network
- Create a resource group and VM
- Enable boot diagnostics on the VM
- View boot diagnostics
- View host metrics
- Install the diagnostics extension

- View VM metrics
- Create an alert
- Manage Windows updates
- Set up advanced monitoring

Advance to the next tutorial to learn about Azure security center.

[Manage VM security](#)

Monitor virtual machine security by using Azure Security Center

6/27/2017 • 5 min to read • [Edit Online](#)

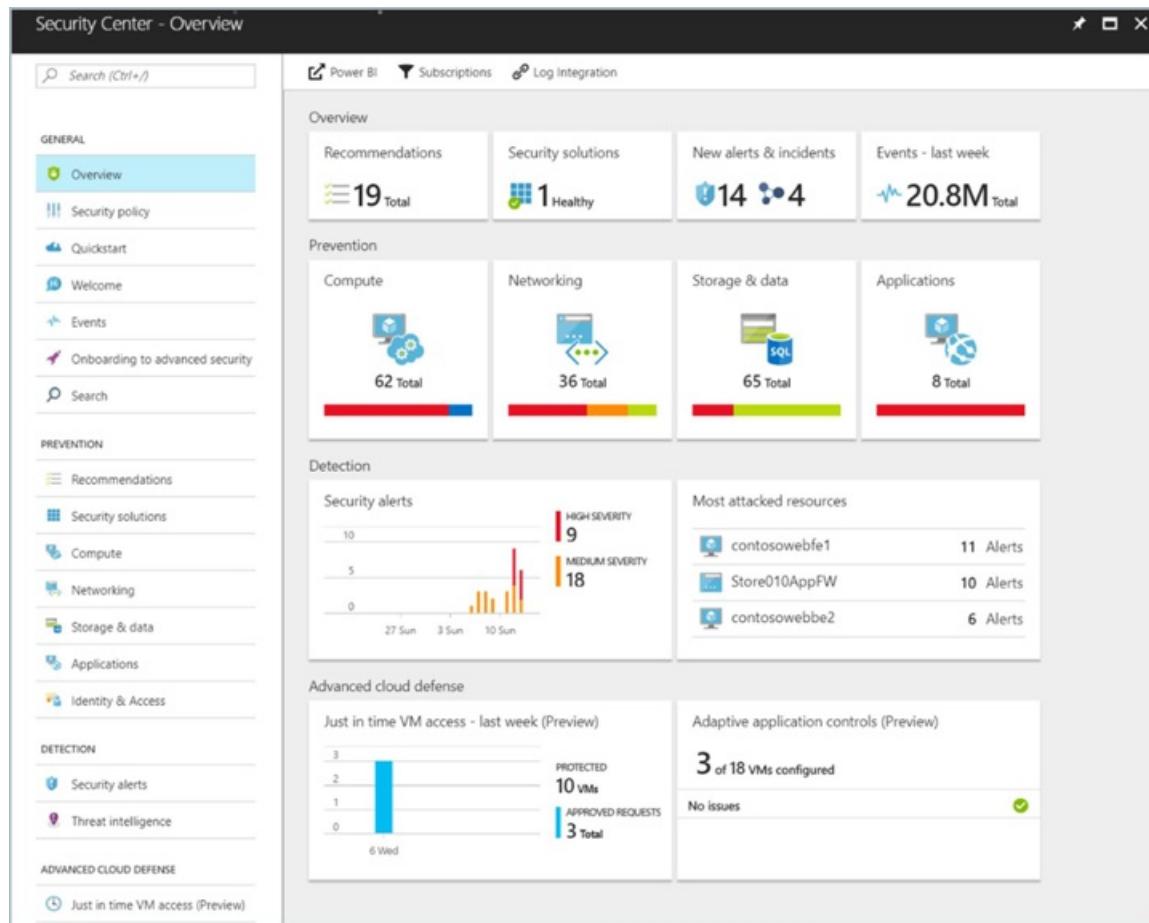
Azure Security Center can help you gain visibility into your Azure resource security practices. Security Center offers integrated security monitoring. It can detect threats that otherwise might go unnoticed. In this tutorial, you learn about Azure Security Center, and how to:

- Set up data collection
- Set up security policies
- View and fix configuration health issues
- Review detected threats

Security Center overview

Security Center identifies potential virtual machine (VM) configuration issues and targeted security threats. These might include VMs that are missing network security groups, unencrypted disks, and brute-force Remote Desktop Protocol (RDP) attacks. The information is shown on the Security Center dashboard in easy-to-read graphs.

To access the Security Center dashboard, in the Azure portal, on the menu, select **Security Center**. On the dashboard, you can see the security health of your Azure environment, find a count of current recommendations, and view the current state of threat alerts. You can expand each high-level chart to see more detail.



Security Center goes beyond data discovery to provide recommendations for issues that it detects. For example, if a VM was deployed without an attached network security group, Security Center displays a recommendation, with

remediation steps you can take. You get automated remediation without leaving the context of Security Center.

The screenshot shows the 'Recommendations' blade in the Azure Security Center. At the top, there's a 'Filter' button. Below it is a table with columns: DESCRIPTION, RESOURCE, STATE, and SEVERITY. The table lists several remediation tasks:

DESCRIPTION	RESOURCE	STATE	SEVERITY	...
Install Endpoint Protection	2016OMSAA	Open	! High	...
Add a Next Generation Firewall	2 endpoints	Open	! High	...
Enable Network Security Groups on sub...	2 subnets	Open	! High	...
Apply disk encryption	3 virtual mac...	Open	! High	...
Enable encryption for Azure Storage Acc...	9 storage acc...	Open	! High	...
Restrict access through Internet facing e...	2 virtual mac...	Open	⚠ Medium	...
Add a vulnerability assessment solution	2016OMSAA	Open	⚠ Medium	...
Remediate OS vulnerabilities (by Micros...	2016OMSLin...	Open	ℹ Low	...

Set up data collection

Before you can get visibility into VM security configurations, you need to set up Security Center data collection. This involves turning on data collection and creating an Azure storage account to hold collected data.

1. On the Security Center dashboard, click **Security policy**, and then select your subscription.
2. For **Data collection**, select **On**.
3. To create a storage account, select **Choose a storage account**. Then, select **OK**.
4. On the **Security Policy** blade, select **Save**.

The Security Center data collection agent is then installed on all VMs, and data collection begins.

Set up a security policy

Security policies are used to define the items for which Security Center collects data and makes recommendations. You can apply different security policies to different sets of Azure resources. Although by default Azure resources are evaluated against all policy items, you can turn off individual policy items for all Azure resources or for a resource group. For in-depth information about Security Center security policies, see [Set security policies in Azure Security Center](#).

To set up a security policy for all Azure resources:

1. On the Security Center dashboard, select **Security policy**, and then select your subscription.
2. Select **Prevention policy**.
3. Turn on or turn off policy items that you want to apply to all Azure resources.
4. When you're finished selecting your settings, select **OK**.
5. On the **Security policy** blade, select **Save**.

To set up a policy for a specific resource group:

1. On the Security Center dashboard, select **Security policy**, and then select a resource group.
2. Select **Prevention policy**.
3. Turn on or turn off policy items that you want to apply to the resource group.
4. Under **INHERITANCE**, select **Unique**.

5. When you're finished selecting your settings, select **OK**.

6. On the **Security policy** blade, select **Save**.

You also can turn off data collection for a specific resource group on this page.

In the following example, a unique policy has been created for a resource group named *myResourceGroup*. In this policy, disk encryption and web application firewall recommendations are turned off.

The screenshot displays three overlapping windows:

- Security policy**: A table showing policy settings for four resources: Free Trial (Inheritance: ..., Data Collection: On), myResourceGroup (Inheritance: Unique, Data Collection: On), RDPBruteForce (Inheritance: Inherited, Data Collection: On), and securitydata (Inheritance: Inherited, Data Collection: On).
- Security policy (myResourceGroup)**: Shows inheritance settings (Inherited, Unique) and a data collection section where "On" is selected for "Collect data from virtual machines".
- Prevention policy**: A list of recommendations with "On" status for most, except Disk encryption and Web application firewall which are set to "Off".

View VM configuration health

After you've turned on data collection and set a security policy, Security Center begins to provide alerts and recommendations. As VMs are deployed, the data collection agent is installed. Security Center is then populated with data for the new VMs. For in-depth information about VM configuration health, see [Protect your VMs in Security Center](#).

As data is collected, the resource health for each VM and related Azure resource is aggregated. The information is shown in an easy-to-read chart.

To view resource health:

1. On the Security Center dashboard, under **Resource security health**, select **Compute**.
2. On the **Compute** blade, select **Virtual machines**. This view provides a summary of the configuration status for all your VMs.

NAME	MONITORED	SYSTEM UPDATES	ENDPOINT PROTE...	VULNERABILITIES	DISK ENCRYPTION
2016OMSAA	✓	!	!	⚠	!
asctest	✓	!	●	!	!
2016OMSLinux	✓	✓	●	!	!

To see all recommendations for a VM, select the VM. Recommendations and remediation are covered in more detail in the next section of this tutorial.

Remediate configuration issues

After Security Center begins to populate with configuration data, recommendations are made based on the security policy you set up. For instance, if a VM was set up without an associated network security group, a recommendation is made to create one.

To see a list of all recommendations:

1. On the Security Center dashboard, select **Recommendations**.
2. Select a specific recommendation. A list of all resources for which the recommendation applies appears.
3. To apply a recommendation, select a specific resource.
4. Follow the instructions for remediation steps.

In many cases, Security Center provides actionable steps you can take to address a recommendation without leaving Security Center. In the following example, Security Center detects a network security group that has an unrestricted inbound rule. On the recommendation page, you can select the **Edit inbound rules** button. The UI that is needed to modify the rule appears.

asctest-nsg

Edit inbound rules

Network security group info

NETWORK SECURITY GROUP asctest-nsg

LOCATION westus

DESCRIPTION Your NSG has inbound rules that open access to 'Any' which might enable attackers to access your resources. We recommend that you edit the below inbound rules, to restrict access to sources, who actually access it.

Related inbound rules

PRIORITY	NAME	SOURCE	SERVICE	ACTIONS
1000	default-allow-ssh	*	TCP	Allow

Associated with

NAME	VIRTUAL MACHINE
 asctest929	asctest

As recommendations are remediated, they are marked as resolved.

View detected threats

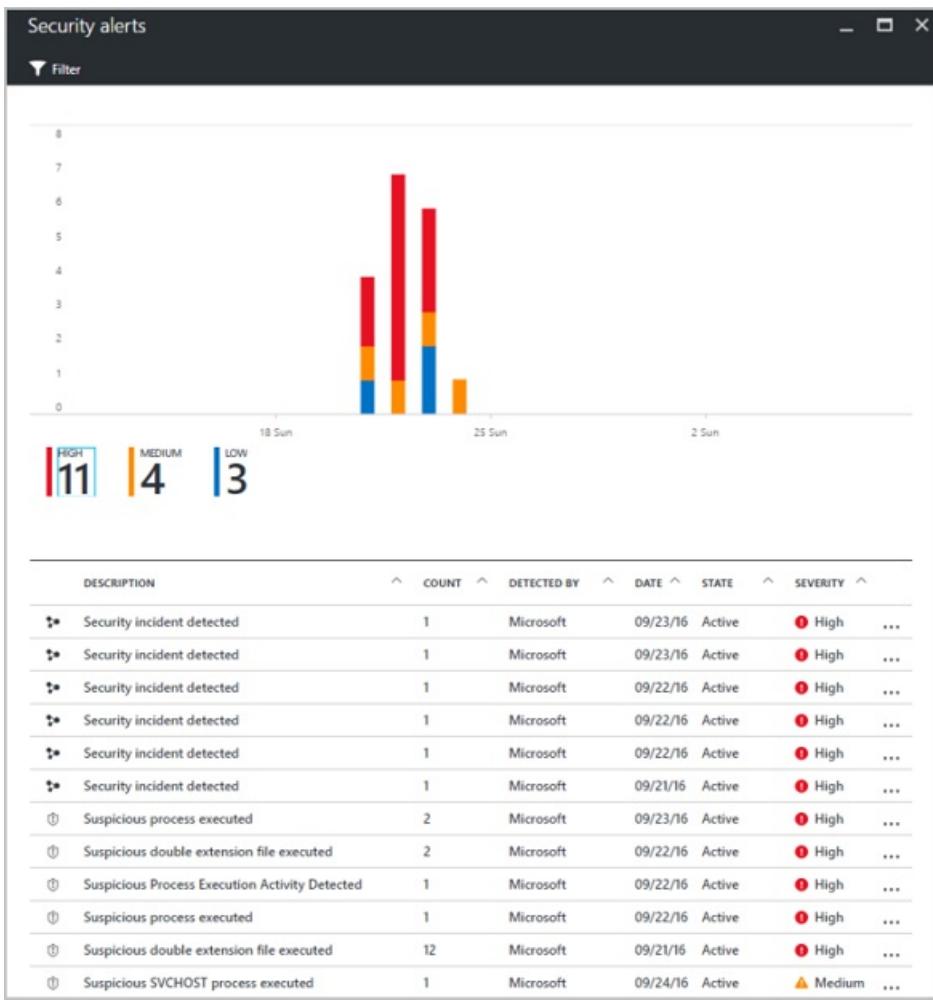
In addition to resource configuration recommendations, Security Center displays threat detection alerts. The security alerts feature aggregates data collected from each VM, Azure networking logs, and connected partner solutions to detect security threats against Azure resources. For in-depth information about Security Center threat detection capabilities, see [Azure Security Center detection capabilities](#).

The security alerts feature requires the Security Center pricing tier to be increased from *Free* to *Standard*. A 30-day **free trial** is available when you move to this higher pricing tier.

To change the pricing tier:

1. On the Security Center dashboard, click **Security policy**, and then select your subscription.
2. Select **Pricing tier**.
3. Select the new tier, and then select **Select**.
4. On the **Security policy** blade, select **Save**.

After you've changed the pricing tier, the security alerts graph begins to populate as security threats are detected.



Select an alert to view information. For example, you can see a description of the threat, the detection time, all threat attempts, and the recommended remediation. In the following example, an RDP brute-force attack was detected, with 294 failed RDP attempts. A recommended resolution is provided.

Failed RDP Brute Force Attack	
myVMWindows	
DESCRIPTION	Several Remote Desktop login attempts were detected from 5.9.57.202, none of them succeeded. Event logs analysis shows that in the last 59 minutes there were 198 failed attempts. Some of the failed login attempts aimed at 2 existing user(s).
DETECTION TIME	Saturday, April 29, 2017, 10:59:56 PM
SEVERITY	 Medium
STATE	Active
ATTACKED RESOURCE	myVMWindows
SUBSCRIPTION	Free Trial (248352d0-5fc9-4c2e-8db3-d8b3462a0020)
DETECTED BY	 Microsoft
ACTION TAKEN	Detected
ALERT START TIME (UTC)	04/30/2017 05:00:06
NON-EXISTENT USERS	97
SUCCESSFUL LOGINS	0
FAILED USER LOGONS	server, administrator
REPORTS	Report: RDP Brute Forcing
REMEDIATION STEPS	<ol style="list-style-type: none"> 1. If available, add the source IP to NSG block list for 24 hours (see https://azure.microsoft.com/en-us/documentation/articles/virtual-networks-nsg/) 2. Enforce the use of strong passwords and do not reuse them across multiple VMs and services (see http://windows.microsoft.com/en-us/Windows7/Tips-for-creating-strong-passwords-and-passphrases) 3. Create an allow list for RDP access in NSG (see https://azure.microsoft.com/en-us/documentation/articles/virtual-networks-nsg/)

Next steps

In this tutorial, you set up Azure Security Center, and then reviewed VMs in Security Center. You learned how to:

- Set up data collection
- Set up security policies
- View and fix configuration health issues
- Review detected threats

Advance to the next tutorial to learn how to create a CI/CD pipeline with Visual Studio Team Services and a Windows VM running IIS.

[Visual Studio Team Services CI/CD pipeline](#)

Create a continuous integration pipeline with Visual Studio Team Services and IIS

10/30/2017 • 8 min to read • [Edit Online](#)

To automate the build, test, and deployment phases of application development, you can use a continuous integration and deployment (CI/CD) pipeline. In this tutorial, you create a CI/CD pipeline using Visual Studio Team Services and a Windows virtual machine (VM) in Azure that runs IIS. You learn how to:

- Publish an ASP.NET web application to a Team Services project
- Create a build definition that is triggered by code commits
- Install and configure IIS on a virtual machine in Azure
- Add the IIS instance to a deployment group in Team Services
- Create a release definition to publish new web deploy packages to IIS
- Test the CI/CD pipeline

This tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable Azurerm` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

Create project in Team Services

Visual Studio Team Services allows for easy collaboration and development without maintaining an on-premises code management solution. Team Services provides cloud code testing, build, and application insights. You can choose a version control repo and IDE that best fits your code development. For this tutorial, you can use a free account to create a basic ASP.NET web app and CI/CD pipeline. If you do not already have a Team Services account, [create one](#).

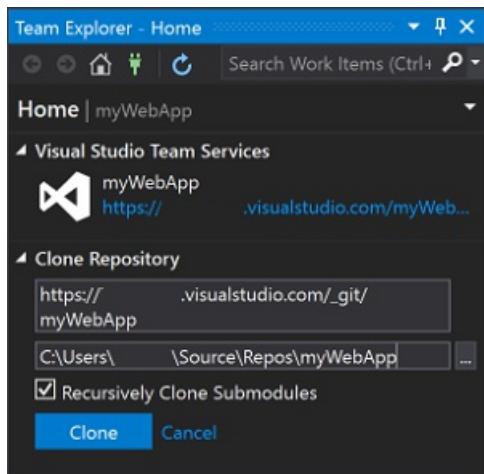
To manage the code commit process, build definitions, and release definitions, create a project in Team Services as follows:

1. Open your Team Services dashboard in a web browser and choose **New project**.
2. Enter *myWebApp* for the **Project name**. Leave all other default values to use *Git* version control and *Agile* work item process.
3. Choose the option to **Share with Team Members**, then select **Create**.
4. Once your project has been created, choose the option to **Initialize with a README or gitignore**, then **Initialize**.
5. Inside your new project, choose **Dashboards** across the top, then select **Open in Visual Studio**.

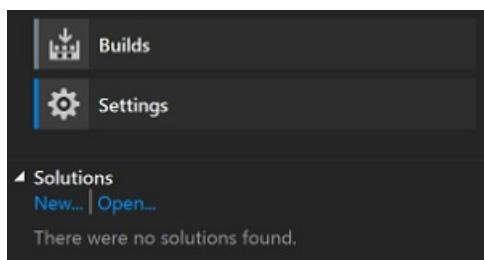
Create ASP.NET web application

In the previous step, you created a project in Team Services. The final step opens your new project in Visual Studio. You manage your code commits in the **Team Explorer** window. Create a local copy of your new project, then create an ASP.NET web application from a template as follows:

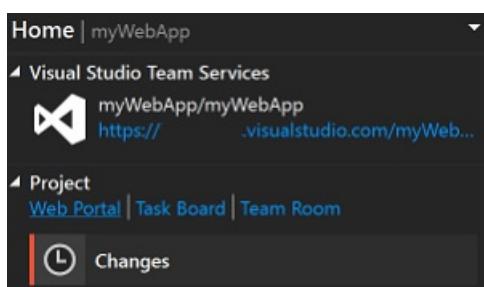
1. Select **Clone** to create a local git repo of your Team Services project.



2. Under **Solutions**, select **New**.



3. Select **Web** templates, and then choose the **ASP.NET Web Application** template.
 - a. Enter a name for your application, such as *myWebApp*, and uncheck the box for **Create directory for solution**.
 - b. If the option is available, uncheck the box to **Add Application Insights to project**. Application Insights requires you to authorize your web application with Azure Application Insights. To keep it simple in this tutorial, skip this process.
 - c. Select **OK**.
4. Choose **MVC** from the template list.
 - a. Select **Change Authentication**, choose **No Authentication**, then select **OK**.
 - b. Select **OK** to create your solution.
5. In the **Team Explorer** window, choose **Changes**.



6. In the commit text box, enter a message such as *Initial commit*. Choose **Commit All and Sync** from the drop-down menu.

Create build definition

In Team Services, you use a build definition to outline how your application should be built. In this tutorial, we create a basic definition that takes our source code, builds the solution, then creates web deploy package we can use to run the web app on an IIS server.

1. In your Team Services project, choose **Build & Release** across the top, then select **Builds**.
2. Select **+ New definition**.

3. Choose the **ASP.NET (PREVIEW)** template and select **Apply**.
4. Leave all the default task values. Under **Get sources**, ensure that the *myWebApp* repository and *master* branch are selected.

The screenshot shows the 'myWebApp-ASP.NET (PREVIEW)-CI' build definition in Visual Studio. The 'Tasks' tab is selected. On the left, the 'Process' section lists tasks: Get sources (selected), NuGet restore, Build solution, Test Assemblies, Publish symbols path, Publish Artifact, and Add Task. On the right, the 'Get sources' configuration pane shows 'From' set to 'This project', 'Repository' set to 'myWebApp', and 'Branch' set to 'master'. There is also an 'Advanced settings' toggle switch.

5. On the **Triggers** tab, move the slider for **Enable this trigger** to *Enabled*.
6. Save the build definition and queue a new build by selecting **Save & queue**, then **Save & queue** again. Leave the defaults and select **Queue**.

Watch as the build is scheduled on a hosted agent, then begins to build. The output is similar to the following example:

The screenshot shows the build log for 'Build 20170509.1'. The build status is 'Build succeeded'. The log details the build steps: Initialize Agent, Initialize Job, Get Sources, NuGet restore, Build solution, Test Assemblies, Publish symbols path, Publish Artifact, Post Job Cleanup, Finalize build, and Report build status. The 'Logs' tab is selected, showing the command-line output of the build process, which includes publishing artifacts to a file share and cleaning up credentials.

Create virtual machine

To provide a platform to run your ASP.NET web app, you need a Windows virtual machine that runs IIS. Team

Services uses an agent to interact with the IIS instance as you commit code and builds are triggered.

Create a Windows Server 2016 VM using [this script sample](#). It takes a few minutes for the script to run and create the VM. Once the VM has been created, open port 80 for web traffic with [Add-AzureRmNetworkSecurityRuleConfig](#) as follows:

```
Get-AzureRmNetworkSecurityGroup ` 
-ResourceGroupName $resourceGroup ` 
-Name "myNetworkSecurityGroup" | ` 
Add-AzureRmNetworkSecurityRuleConfig ` 
-Name "myNetworkSecurityGroupRuleWeb" ` 
-Protocol "Tcp" ` 
-Direction "Inbound" ` 
-Priority "1001" ` 
-SourceAddressPrefix "*" ` 
-SourcePortRange "*" ` 
-DestinationAddressPrefix "*" ` 
-DestinationPortRange "80" ` 
-Access "Allow" | ` 
Set-AzureRmNetworkSecurityGroup
```

To connect to your VM, obtain the public IP address with [Get-AzureRmPublicIpAddress](#) as follows:

```
Get-AzureRmPublicIpAddress -ResourceGroupName $resourceGroup | Select IpAddress
```

Create a remote desktop session to your VM:

```
mstsc /v:<publicIpAddress>
```

On the VM, open an **Administrator PowerShell** command prompt. Install IIS and required .NET features as follows:

```
Install-WindowsFeature Web-Server,Web-Asp-Net45,.NET-Framework-Features
```

Create deployment group

To push out the web deploy package to the IIS server, you define a deployment group in Team Services. This group allows you to specify which servers are the target of new builds as you commit code to Team Services and builds are completed.

1. In Team Services, choose **Build & Release** and then select **Deployment groups**.
2. Choose **Add Deployment group**.
3. Enter a name for the group, such as *myIIS*, then select **Create**.
4. In the **Register machines** section, ensure *Windows* is selected, then check the box to **Use a personal access token in the script for authentication**.
5. Select **Copy script to clipboard**.

Add IIS VM to the deployment group

With the deployment group created, add each IIS instance to the group. Team Services generates a script that downloads and configures an agent on the VM that receives new web deploy packages then applies it to IIS.

1. Back in the **Administrator PowerShell** session on your VM, paste and run the script copied from Team Services.
2. When prompted to configure tags for the agent, choose *Y* and enter *web*.

3. When prompted for the user account, press *Return* to accept the defaults.
4. Wait for the script to finish with a message *Service vstsagent.account.computername started successfully*.
5. In the **Deployment groups** page of the **Build & Release** menu, open the *myIIS* deployment group. On the **Machines** tab, verify that your VM is listed.

Machine Name	Tags	Latest Deployment
myVM	web	Never deployed

Create release definition

To publish your builds, you create a release definition in Team Services. This definition is triggered automatically by a successful build of your application. You choose the deployment group to push your web deploy package to, and define the appropriate IIS settings.

1. Choose **Build & Release**, then select **Builds**. Choose the build definition created in a previous step.
2. Under **Recently completed**, choose the most recent build, then select **Release**.
3. Choose **Yes** to create a release definition.
4. Choose the **Empty** template, then select **Next**.
5. Verify the project and source build definition are populated with your project.
6. Select the **Continuous deployment** check box, then select **Create**.
7. Select the drop-down box next to **+ Add tasks** and choose **Add a deployment group phase**.

8. Choose **Add** next to **IIS Web App Deploy(Preview)**, then select **Close**.
9. Select the **Run on deployment group** parent task.
 - a. For **Deployment Group**, select the deployment group you created earlier, such as *myIIS*.
 - b. In the **Machine tags** box, select **Add** and choose the *web* tag.

10. Select the **Deploy: IIS Web App Deploy** task to configure your IIS instance settings as follows:

- For **Website Name**, enter *Default Web Site*.
- Leave all the other default settings.

11. Choose **Save**, then select **OK** twice.

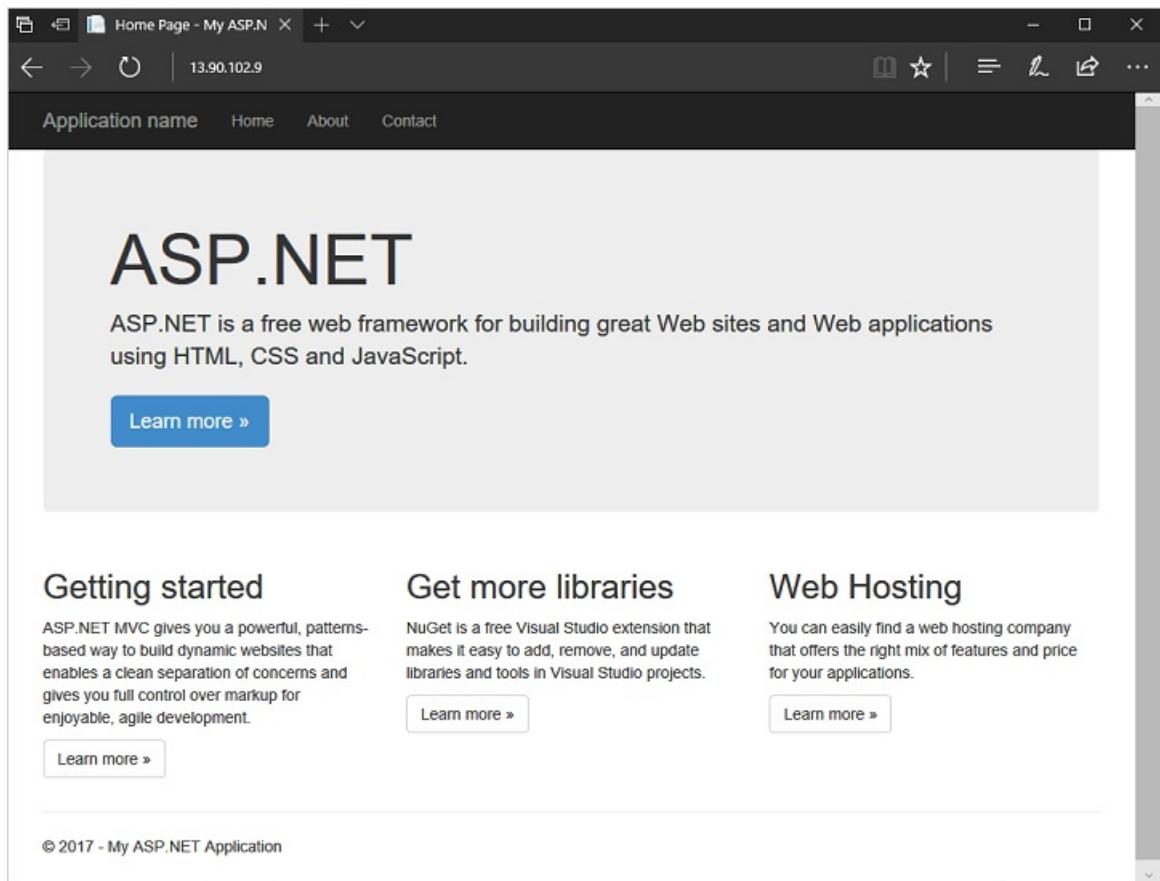
Create a release and publish

You can now push your web deploy package as a new release. This step communicates with the agent on each instance that is part of the deployment group, pushes the web deploy package, then configures IIS to run the updated web application.

- In your release definition, select **+ Release**, then choose **Create Release**.
- Verify that the latest build is selected in the drop-down list, along with **Automated deployment: After release creation**. Select **Create**.
- A small banner appears across the top of your release definition, such as *Release 'Release-1' has been created*. Select the release link.
- Open the **Logs** tab to watch the release progress.

Deployment summary: Environment 1	
Deployment Status	SUCCEEDED
Sub status	Succeeded
Trigger	Automated - After release creation
Requested by	
Requested for	
Queued time	6 minutes ago
Completed time	5 minutes ago

- After the release is complete, open a web browser and enter the public IP address of your VM. Your ASP.NET web application is running.



Test the whole CI/CD pipeline

With your web application running on IIS, now try the whole CI/CD pipeline. After you make a change in Visual Studio and commit your code, a build is triggered which then triggers a release of your updated web deploy package to IIS:

1. In Visual Studio, open the **Solution Explorer** window.
2. Navigate to and open *myWebApp | Views | Home | Index.cshtml*
3. Edit line 6 to read:

```
<h1>ASP.NET with VSTS and CI/CD!</h1>
```
4. Save the file.
5. Open the **Team Explorer** window, select the *myWebApp* project, then choose **Changes**.
6. Enter a commit message, such as *Testing CI/CD pipeline*, then choose **Commit All and Sync** from the drop-down menu.
7. In Team Services workspace, a new build is triggered from the code commit.
 - Choose **Build & Release**, then select **Builds**.
 - Choose your build definition, then select the **Queued & running** build to watch as the build progresses.
8. Once the build is successful, a new release is triggered.
 - Choose **Build & Release**, then **Releases** to see the web deploy package pushed to your IIS VM.
 - Select the **Refresh** icon to update the status. When the *Environments* column shows a green check mark, the release has successfully deployed to IIS.
9. To see your changes applied, refresh your IIS website in a browser.

Home Page - My ASP.NET

Application name Home About Contact

ASP.NET with VSTS and CI/CD!

ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS and JavaScript.

[Learn more »](#)

Getting started

ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and gives you full control over markup for enjoyable, agile development.

[Learn more »](#)

Get more libraries

NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in Visual Studio projects.

[Learn more »](#)

Web Hosting

You can easily find a web hosting company that offers the right mix of features and price for your applications.

[Learn more »](#)

© 2017 - My ASP.NET Application

Next steps

In this tutorial, you created an ASP.NET web application in Team Services and configured build and release definitions to deploy new web deploy packages to IIS on each code commit. You learned how to:

- Publish an ASP.NET web application to a Team Services project
- Create a build definition that is triggered by code commits
- Install and configure IIS on a virtual machine in Azure
- Add the IIS instance to a deployment group in Team Services
- Create a release definition to publish new web deploy packages to IIS
- Test the CI/CD pipeline

Advance to the next tutorial to learn how to install a SQL\IIS\.NET stack on a pair of Windows VMs.

[SQL\IIS\.NET stack](#)

Install a SQL\IIS\.NET stack in Azure

10/30/2017 • 2 min to read • [Edit Online](#)

In this tutorial, we install a SQL\IIS\.NET stack using Azure PowerShell. This stack consists of two VMs running Windows Server 2016, one with IIS and .NET and the other with SQL Server.

- Create a VM using New-AzVM
- Install IIS and the .NET Core SDK on the VM
- Create a VM running SQL Server
- Install the SQL Server extension

Create a IIS VM

In this example, we use the [New-AzVM](#) cmdlet in the PowerShell Cloud Shell to quickly create a Windows Server 2016 VM and then install IIS and the .NET Framework. The IIS and SQL VMs share a resource group and virtual network, so we create variables for those names.

Click on the **Try It** button to the upper right of the code block to launch Cloud Shell in this window. You will be asked to provide credentials for the virtual machine at the cmd prompt.

```
$vNetName = "myIISQLvNet"
$resourceGroup = "myIISQLGroup"
New-AzVm -Name myIISVM -ResourceGroupName $resourceGroup -VirtualNetworkName $vNetName
```

Install IIS and the .NET framework using the custom script extension.

```
Set-AzureRmVMExtension -ResourceGroupName $resourceGroup ` 
    -ExtensionName IIS ` 
    -VMName myIISVM ` 
    -Publisher Microsoft.Compute ` 
    -ExtensionType CustomScriptExtension ` 
    -TypeHandlerVersion 1.4 ` 
    -SettingString '{"commandToExecute":"powershell Add-WindowsFeature Web-Server,Web-Asp-Net45,NET-Framework-Features"}' ` 
    -Location EastUS
```

Azure SQL VM

We use a pre-configured Azure marketplace image of a SQL server to create the SQL VM. We first create the VM, then we install the SQL Server Extension on the VM.

```

# Create user object. You get a pop-up prompting you to enter the credentials for the VM.
$cred = Get-Credential -Message "Enter a username and password for the virtual machine."

# Create a subnet configuration
$vNet = Get-AzureRmVirtualNetwork -Name $vNetName -ResourceGroupName $resourceGroup
Add-AzureRmVirtualNetworkSubnetConfig -Name mySQLSubnet -VirtualNetwork $vNet -AddressPrefix "192.168.2.0/24"
Set-AzureRmVirtualNetwork -VirtualNetwork $vNet

# Create a public IP address and specify a DNS name
$pip = New-AzureRmPublicIpAddress -ResourceGroupName $resourceGroup -Location eastus ` 
    -Name "mypublicdns$(Get-Random)" -AllocationMethod Static -IdleTimeoutInMinutes 4

# Create an inbound network security group rule for port 3389
$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig -Name myNetworkSecurityGroupRuleRDP -Protocol Tcp ` 
    -Direction Inbound -Priority 1000 -SourceAddressPrefix * -SourcePortRange * -DestinationAddressPrefix * ` 
    -DestinationPortRange 3389 -Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $resourceGroup -Location eastus ` 
    -Name myNetworkSecurityGroup -SecurityRules $nsgRuleRDP

# Create a virtual network card and associate with public IP address and NSG
$nic = New-AzureRmNetworkInterface -Name mySQLNic -ResourceGroupName $resourceGroup -Location eastus ` 
    -SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $pip.Id -NetworkSecurityGroupId $nsg.Id

# Create a virtual machine configuration
$vmConfig = New-AzureRmVMConfig -VMName mySQLVM -VMSize Standard_D1 | ` 
    Set-AzureRmVMOperatingSystem -Windows -ComputerName mySQLVM -Credential $cred | ` 
    Set-AzureRmVMSourceImage -PublisherName MicrosoftSQLServer -Offer SQL2014SP2-WS2012R2 -Skus Enterprise -Version latest | ` 
    Add-AzureRmVMNetworkInterface -Id $nic.Id

# Create the VM
New-AzureRmVM -ResourceGroupName $resourceGroup -Location eastus -VM $vmConfig

```

Use [Set-AzureRmVMSqlServerExtension](#) to add the [SQL Server extension](#) to the SQL VM.

```
Set-AzureRmVMSqlServerExtension -ResourceGroupName $resourceGroup -VMName mySQLVM -name "SQLExtension"
```

Next steps

In this tutorial, you installed a SQL\IIS\.NET stack using Azure PowerShell. You learned how to:

- Create a VM using New-AzVM
- Install IIS and the .NET Core SDK on the VM
- Create a VM running SQL Server
- Install the SQL Server extension

Advance to the next tutorial to learn how to secure IIS web server with SSL certificates.

[Secure IIS web server with SSL certificates](#)

Secure IIS web server with SSL certificates on a Windows virtual machine in Azure

8/21/2017 • 5 min to read • [Edit Online](#)

To secure web servers, a Secure Sockets Layer (SSL) certificate can be used to encrypt web traffic. These SSL certificates can be stored in Azure Key Vault, and allow secure deployments of certificates to Windows virtual machines (VMs) in Azure. In this tutorial you learn how to:

- Create an Azure Key Vault
- Generate or upload a certificate to the Key Vault
- Create a VM and install the IIS web server
- Inject the certificate into the VM and configure IIS with an SSL binding

This tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable Azurerm` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#).

Overview

Azure Key Vault safeguards cryptographic keys and secrets, such as certificates or passwords. Key Vault helps streamline the certificate management process and enables you to maintain control of keys that access those certificates. You can create a self-signed certificate inside Key Vault, or upload an existing, trusted certificate that you already own.

Rather than using a custom VM image that includes certificates baked-in, you inject certificates into a running VM. This process ensures that the most up-to-date certificates are installed on a web server during deployment. If you renew or replace a certificate, you don't also have to create a new custom VM image. The latest certificates are automatically injected as you create additional VMs. During the whole process, the certificates never leave the Azure platform or are exposed in a script, command-line history, or template.

Create an Azure Key Vault

Before you can create a Key Vault and certificates, create a resource group with [New-AzureRmResourceGroup](#). The following example creates a resource group named *myResourceGroupSecureWeb* in the *East US* location:

```
$resourceGroup = "myResourceGroupSecureWeb"
.setLocation = "East US"
New-AzureRmResourceGroup -ResourceGroupName $resourceGroup -Location $location
```

Next, create a Key Vault with [New-AzureRmKeyVault](#). Each Key Vault requires a unique name, and should be all lower case. Replace `<mykeyvault>` in the following example with your own unique Key Vault name:

```
$keyvaultName="<mykeyvault>"
New-AzureRmKeyVault -VaultName $keyvaultName ` 
    -ResourceGroup $resourceGroup ` 
    -Location $location ` 
    -EnabledForDeployment
```

Generate a certificate and store in Key Vault

For production use, you should import a valid certificate signed by trusted provider with [Import-AzureKeyVaultCertificate](#). For this tutorial, the following example shows how you can generate a self-signed certificate with [Add-AzureKeyVaultCertificate](#) that uses the default certificate policy from [New-AzureKeyVaultCertificatePolicy](#).

```
$policy = New-AzureKeyVaultCertificatePolicy ` 
    -SubjectName "CN=www.contoso.com" ` 
    -SecretContentType "application/x-pkcs12" ` 
    -IssuerName Self ` 
    -ValidityInMonths 12

Add-AzureKeyVaultCertificate ` 
    -VaultName $keyvaultName ` 
    -Name "mycert" ` 
    -CertificatePolicy $policy
```

Create a virtual machine

Set an administrator username and password for the VM with [Get-Credential](#):

```
$cred = Get-Credential
```

Now you can create the VM with [New-AzureRmVM](#). The following example creates the required virtual network components, the OS configuration, and then creates a VM named *myVM*:

```
# Create a subnet configuration
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig ` 
    -Name mySubnet ` 
    -AddressPrefix 192.168.1.0/24

# Create a virtual network
$vnet = New-AzureRmVirtualNetwork ` 
    -ResourceGroupName $resourceGroup ` 
    -Location $location ` 
    -Name "myVnet" ` 
    -AddressPrefix 192.168.0.0/16 ` 
    -Subnet $subnetConfig

# Create a public IP address and specify a DNS name
$publicIP = New-AzureRmPublicIpAddress ` 
    -ResourceGroupName $resourceGroup ` 
    -Location $location ` 
    -AllocationMethod Static ` 
    -IdleTimeoutInMinutes 4 ` 
    -Name "myPublicIP"

# Create an inbound network security group rule for port 3389
$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig ` 
    -Name "myNetworkSecurityGroupRuleRDP" ` 
    -Protocol "Tcp" ` 
    -Direction "Inbound" ` 
    -Priority 1000 ` 
    -SourceAddressPrefix * ` 
    -SourcePortRange * ` 
    -DestinationAddressPrefix * ` 
    -DestinationPortRange 3389 ` 
    -Access Allow

# Create an inbound network security group rule for port 443
$nsgRuleWeb = New-AzureRmNetworkSecurityRuleConfig ` 
    -Name "myNetworkSecurityGroupRuleWWW" ` 
    -Protocol "Tcp" ` 
    -Direction "Inbound" `
```

```

-Priority 1001 ` 
-SourceAddressPrefix * ` 
-SourcePortRange * ` 
-DestinationAddressPrefix * ` 
-DestinationPortRange 443 ` 
-Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup ` 
-ResourceGroupName $resourceGroup ` 
-Location $location ` 
-Name "myNetworkSecurityGroup" ` 
-SecurityRules $nsgRuleRDP,$nsgRuleWeb

# Create a virtual network card and associate with public IP address and NSG
$nic = New-AzureRmNetworkInterface ` 
-Name "myNic" ` 
-ResourceGroupName $resourceGroup ` 
-Location $location ` 
-SubnetId $vnet.Subnets[0].Id ` 
-PublicIpAddressId $publicIP.Id ` 
-NetworkSecurityGroupId $nsg.Id

# Create a virtual machine configuration
$vmConfig = New-AzureRmVMConfig -VMName "myVM" -VMSize "Standard_DS2" | ` 
Set-AzureRmVMOperatingSystem -Windows -ComputerName "myVM" -Credential $cred | ` 
Set-AzureRmVMSourceImage -PublisherName "MicrosoftWindowsServer" ` 
-Offer "WindowsServer" -Skus "2016-Datacenter" -Version "latest" | ` 
Add-AzureRmVMNetworkInterface -Id $nic.Id

# Create virtual machine
New-AzureRmVM -ResourceGroupName $resourceGroup -Location $location -VM $vmConfig

# Use the Custom Script Extension to install IIS
Set-AzureRmVMExtension -ResourceGroupName $resourceGroup ` 
-ExtensionName "IIS" ` 
-VMName "myVM" ` 
-Location $location ` 
-Publisher "Microsoft.Compute" ` 
-ExtensionType "CustomScriptExtension" ` 
-TypeHandlerVersion 1.8 ` 
-SettingString '{"commandToExecute":"powershell Add-WindowsFeature Web-Server -IncludeManagementTools"}'

```

It takes a few minutes for the VM to be created. The last step uses the Azure Custom Script Extension to install the IIS web server with [Set-AzureRmVmExtension](#).

Add a certificate to VM from Key Vault

To add the certificate from Key Vault to a VM, obtain the ID of your certificate with [Get-AzureKeyVaultSecret](#). Add the certificate to the VM with [Add-AzureRmVMSecret](#):

```

$certURL=(Get-AzureKeyVaultSecret -VaultName $keyvaultName -Name "mycert").id

$vm=Get-AzureRmVM -ResourceGroupName $resourceGroup -Name "myVM"
$vaultId=(Get-AzureRmKeyVault -ResourceGroupName $resourceGroup -VaultName $keyVaultName).ResourceId
$vm = Add-AzureRmVMSecret -VM $vm -SourceVaultId $vaultId -CertificateStore "My" -CertificateUrl $certURL

Update-AzureRmVM -ResourceGroupName $resourceGroup -VM $vm

```

Configure IIS to use the certificate

Use the Custom Script Extension again with [Set-AzureRmVMExtension](#) to update the IIS configuration. This update applies the certificate injected from Key Vault to IIS and configures the web binding:

```

$PublicSettings = '{
    "fileUris":["https://raw.githubusercontent.com/iainfoulds/azure-samples/master/secure-iis.ps1"],
    "commandToExecute":"powershell -ExecutionPolicy Unrestricted -File secure-iis.ps1"
}'

Set-AzureRmVMExtension -ResourceGroupName $resourceGroup ` 
    -ExtensionName "IIS" ` 
    -VMName "myVM" ` 
    -Location $location ` 
    -Publisher "Microsoft.Compute" ` 
    -ExtensionType "CustomScriptExtension" ` 
    -TypeHandlerVersion 1.8 ` 
    -SettingString $publicSettings

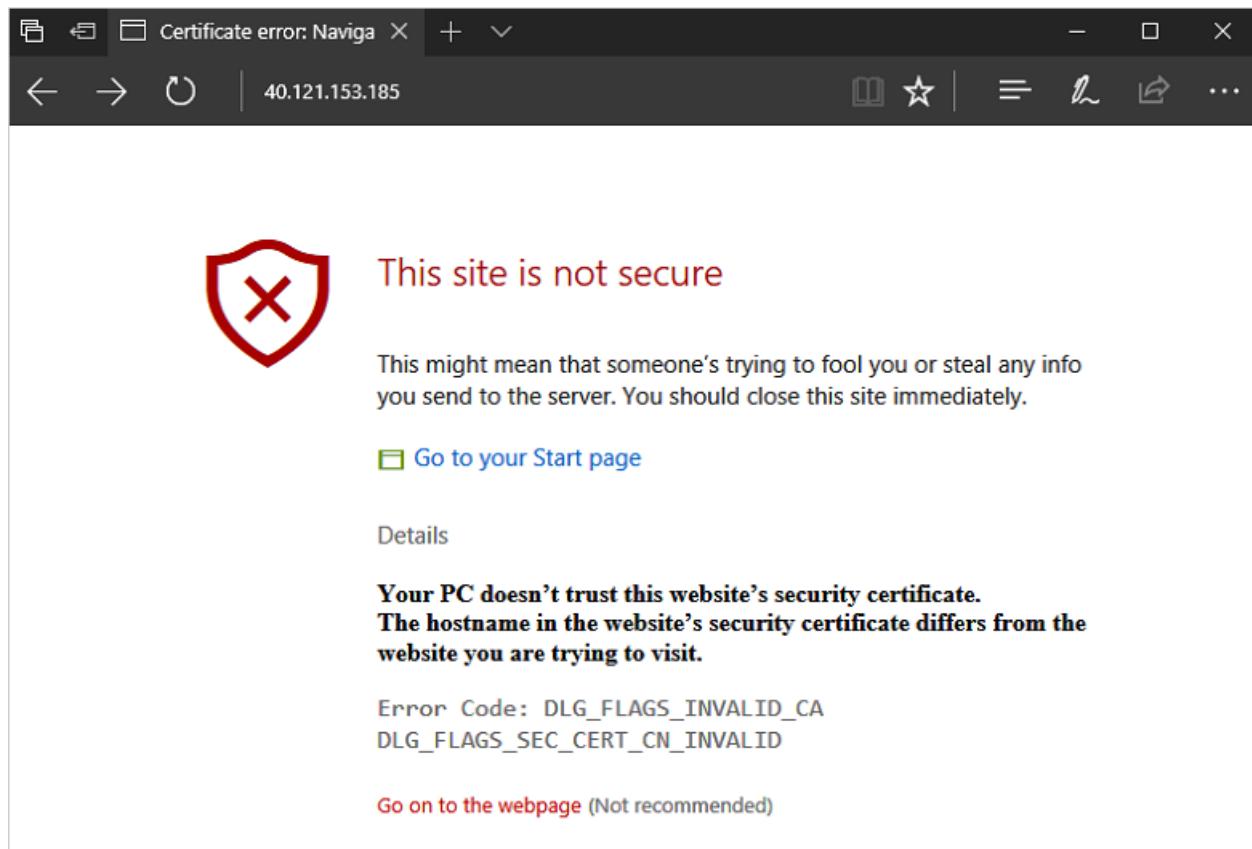
```

Test the secure web app

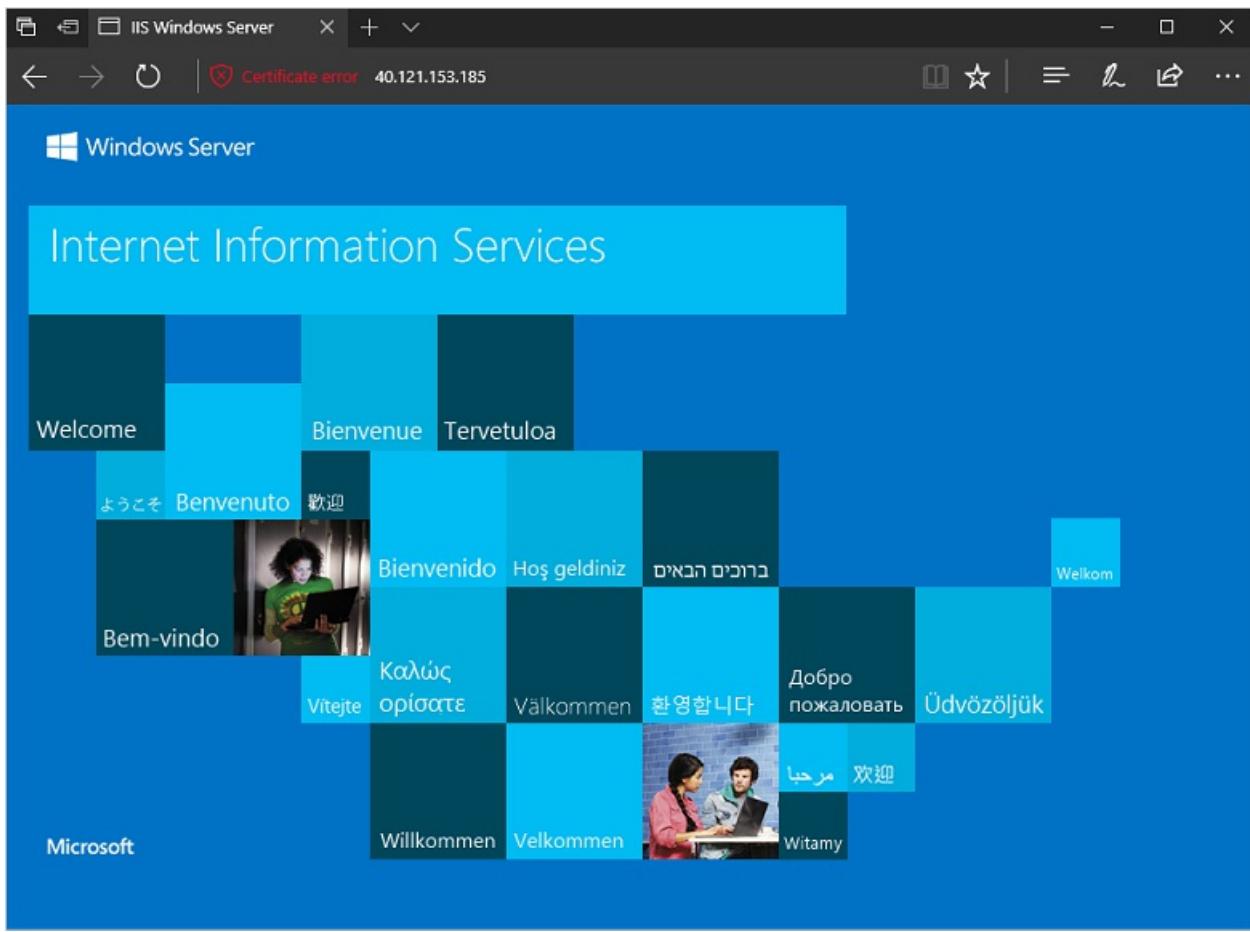
Obtain the public IP address of your VM with [Get-AzureRmPublicIPAddress](#). The following example obtains the IP address for `myPublicIP` created earlier:

```
Get-AzureRmPublicIPAddress -ResourceGroupName $resourceGroup -Name "myPublicIP" | select "IpAddress"
```

Now you can open a web browser and enter `https://<myPublicIP>` in the address bar. To accept the security warning if you used a self-signed certificate, select **Details** and then **Go on to the webpage**:



Your secured IIS website is then displayed as in the following example:



Next steps

In this tutorial, you secured an IIS web server with an SSL certificate stored in Azure Key Vault. You learned how to:

- Create an Azure Key Vault
- Generate or upload a certificate to the Key Vault
- Create a VM and install the IIS web server
- Inject the certificate into the VM and configure IIS with an SSL binding

Follow this link to see pre-built virtual machine script samples.

[Windows virtual machine script samples](#)

Azure Virtual Machine PowerShell samples

12/15/2017 • 1 min to read • [Edit Online](#)

The following table includes links to PowerShell scripts samples that create and manage Windows virtual machines.

Create virtual machines	
Quickly create a virtual machine	Creates a resource group, virtual machine, and all related resources, with the minimum of prompts.
Create a fully configured virtual machine	Creates a resource group, virtual machine, and all related resources .
Create highly available virtual machines	Creates several virtual machines in a highly available and load balanced configuration.
Create a VM and run configuration script	Creates a virtual machine and uses the Azure Custom Script extension to install IIS.
Create a VM and run DSC configuration	Creates a virtual machine and uses the Azure Desired State Configuration (DSC) extension to install IIS.
Upload a VHD and create VMs	Uploads a local VHD file to Azure, creates and image from the VHD and then creates a VM from that image.
Create a VM from a managed OS disk	Creates a virtual machine by attaching an existing Managed Disk as OS disk.
Create a VM from a snapshot	Creates a virtual machine from a snapshot by first creating a managed disk from snapshot and then attaching the new managed disk as OS disk.
Manage storage	
Create managed disk from a VHD in same or different subscription	Creates a managed disk from a specialized VHD as a OS disk or from a data VHD as data disk in same or different subscription.
Create a managed disk from a snapshot	Creates a managed disk from a snapshot.
Copy managed disk to same or different subscription	Copies managed disk to same or different subscription but in the same region as the parent managed disk.
Export a snapshot as VHD to a storage account	Exports a managed snapshot as VHD to a storage account in different region.
Create a snapshot from a VHD	Creates snapshot from a VHD to create multiple identical managed disks from snapshot in short amount of time.

Copy snapshot to same or different subscription	Copies snapshot to same or different subscription but in the same region as the parent snapshot.
Secure virtual machines	
Encrypt a VM and data disks	Creates an Azure Key Vault, encryption key, and service principal, then encrypts a VM.
Monitor virtual machines	
Monitor a VM with Operations Management Suite	Creates a virtual machine, installs the Operations Management Suite agent, and enrolls the VM in an OMS Workspace.

Azure CLI Samples for Windows virtual machines

6/27/2017 • 1 min to read • [Edit Online](#)

The following table includes links to bash scripts built using the Azure CLI that deploy Windows virtual machines.

Create virtual machines	
Create a virtual machine	Creates a Windows virtual machine with minimal configuration.
Create a fully configured virtual machine	Creates a resource group, virtual machine, and all related resources.
Create highly available virtual machines	Creates several virtual machines in a highly available and load balanced configuration.
Create a VM and run configuration script	Creates a virtual machine and uses the Azure Custom Script extension to install IIS.
Create a VM and run DSC configuration	Creates a virtual machine and uses the Azure Desired State Configuration (DSC) extension to install IIS.
Network virtual machines	
Secure network traffic between virtual machines	Creates two virtual machines, all related resources, and an internal and external network security groups (NSG).
Secure virtual machines	
Encrypt a VM and data disks	Creates an Azure Key Vault, encryption key, and service principal, then encrypts a VM.
Monitor virtual machines	
Monitor a VM with Operations Management Suite	Creates a virtual machine, installs the Operations Management Suite agent, and enrolls the VM in an OMS Workspace.

Azure Resource Manager overview

11/15/2017 • 15 min to read • [Edit Online](#)

The infrastructure for your application is typically made up of many components – maybe a virtual machine, storage account, and virtual network, or a web app, database, database server, and 3rd party services. You do not see these components as separate entities, instead you see them as related and interdependent parts of a single entity. You want to deploy, manage, and monitor them as a group. Azure Resource Manager enables you to work with the resources in your solution as a group. You can deploy, update, or delete all the resources for your solution in a single, coordinated operation. You use a template for deployment and that template can work for different environments such as testing, staging, and production. Resource Manager provides security, auditing, and tagging features to help you manage your resources after deployment.

Terminology

If you are new to Azure Resource Manager, there are some terms you might not be familiar with.

- **resource** - A manageable item that is available through Azure. Some common resources are a virtual machine, storage account, web app, database, and virtual network, but there are many more.
- **resource group** - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. See [Resource groups](#).
- **resource provider** - A service that supplies the resources you can deploy and manage through Resource Manager. Each resource provider offers operations for working with the resources that are deployed. Some common resource providers are Microsoft.Compute, which supplies the virtual machine resource, Microsoft.Storage, which supplies the storage account resource, and Microsoft.Web, which supplies resources related to web apps. See [Resource providers](#).
- **Resource Manager template** - A JavaScript Object Notation (JSON) file that defines one or more resources to deploy to a resource group. It also defines the dependencies between the deployed resources. The template can be used to deploy the resources consistently and repeatedly. See [Template deployment](#).
- **declarative syntax** - Syntax that lets you state "Here is what I intend to create" without having to write the sequence of programming commands to create it. The Resource Manager template is an example of declarative syntax. In the file, you define the properties for the infrastructure to deploy to Azure.

The benefits of using Resource Manager

Resource Manager provides several benefits:

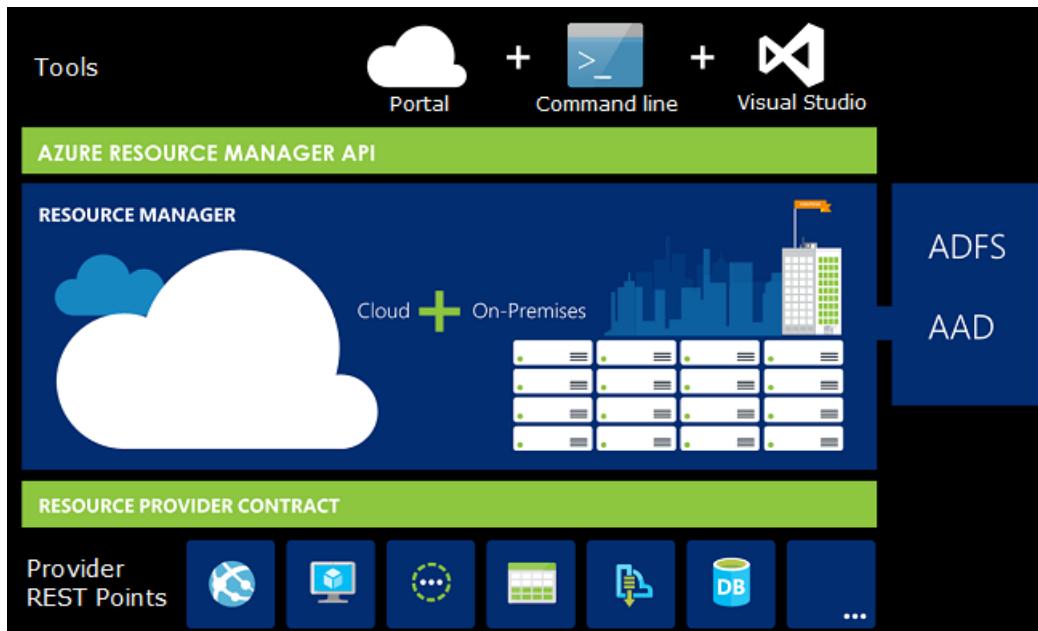
- You can deploy, manage, and monitor all the resources for your solution as a group, rather than handling these resources individually.
- You can repeatedly deploy your solution throughout the development lifecycle and have confidence your resources are deployed in a consistent state.
- You can manage your infrastructure through declarative templates rather than scripts.
- You can define the dependencies between resources so they are deployed in the correct order.
- You can apply access control to all services in your resource group because Role-Based Access Control (RBAC) is natively integrated into the management platform.
- You can apply tags to resources to logically organize all the resources in your subscription.
- You can clarify your organization's billing by viewing costs for a group of resources sharing the same tag.

Resource Manager provides a new way to deploy and manage your solutions. If you used the earlier deployment model and want to learn about the changes, see [Understanding Resource Manager deployment and classic deployment](#).

Consistent management layer

Resource Manager provides a consistent management layer for the tasks you perform through Azure PowerShell, Azure CLI, Azure portal, REST API, and development tools. All the tools use a common set of operations. You use the tools that work best for you, and can use them interchangeably without confusion.

The following image shows how all the tools interact with the same Azure Resource Manager API. The API passes requests to the Resource Manager service, which authenticates and authorizes the requests. Resource Manager then routes the requests to the appropriate resource providers.



Guidance

The following suggestions help you take full advantage of Resource Manager when working with your solutions.

1. Define and deploy your infrastructure through the declarative syntax in Resource Manager templates, rather than through imperative commands.
2. Define all deployment and configuration steps in the template. You should have no manual steps for setting up your solution.
3. Run imperative commands to manage your resources, such as to start or stop an app or machine.
4. Arrange resources with the same lifecycle in a resource group. Use tags for all other organizing of resources.

For recommendations about templates, see [Best practices for creating Azure Resource Manager templates](#).

For guidance on how enterprises can use Resource Manager to effectively manage subscriptions, see [Azure enterprise scaffold - prescriptive subscription governance](#).

Resource groups

There are some important factors to consider when defining your resource group:

1. All the resources in your group should share the same lifecycle. You deploy, update, and delete them together. If one resource, such as a database server, needs to exist on a different deployment cycle it should be in another resource group.
2. Each resource can only exist in one resource group.

3. You can add or remove a resource to a resource group at any time.
4. You can move a resource from one resource group to another group. For more information, see [Move resources to new resource group or subscription](#).
5. A resource group can contain resources that reside in different regions.
6. A resource group can be used to scope access control for administrative actions.
7. A resource can interact with resources in other resource groups. This interaction is common when the two resources are related but do not share the same lifecycle (for example, web apps connecting to a database).

When creating a resource group, you need to provide a location for that resource group. You may be wondering, "Why does a resource group need a location? And, if the resources can have different locations than the resource group, why does the resource group location matter at all?" The resource group stores metadata about the resources. Therefore, when you specify a location for the resource group, you are specifying where that metadata is stored. For compliance reasons, you may need to ensure that your data is stored in a particular region.

Resource providers

Each resource provider offers a set of resources and operations for working with an Azure service. For example, if you want to store keys and secrets, you work with the **Microsoft.KeyVault** resource provider. This resource provider offers a resource type called **vaults** for creating the key vault.

The name of a resource type is in the format: **{resource-provider}/{resource-type}**. For example, the key vault type is **Microsoft.KeyVault/vaults**.

Before getting started with deploying your resources, you should gain an understanding of the available resource providers. Knowing the names of resource providers and resources helps you define resources you want to deploy to Azure. Also, you need to know the valid locations and API versions for each resource type. For more information, see [Resource providers and types](#).

Template deployment

With Resource Manager, you can create a template (in JSON format) that defines the infrastructure and configuration of your Azure solution. By using a template, you can repeatedly deploy your solution throughout its lifecycle and have confidence your resources are deployed in a consistent state. When you create a solution from the portal, the solution automatically includes a deployment template. You do not have to create your template from scratch because you can start with the template for your solution and customize it to meet your specific needs. You can retrieve a template for an existing resource group by either exporting the current state of the resource group, or viewing the template used for a particular deployment. Viewing the [exported template](#) is a helpful way to learn about the template syntax.

To learn about the format of the template and how you construct it, see [Create your first Azure Resource Manager template](#). To view the JSON syntax for resources types, see [Define resources in Azure Resource Manager templates](#).

Resource Manager processes the template like any other request (see the image for [Consistent management layer](#)). It parses the template and converts its syntax into REST API operations for the appropriate resource providers. For example, when Resource Manager receives a template with the following resource definition:

```

"resources": [
  {
    "apiVersion": "2016-01-01",
    "type": "Microsoft.Storage/storageAccounts",
    "name": "mystorageaccount",
    "location": "westus",
    "sku": {
      "name": "Standard_LRS"
    },
    "kind": "Storage",
    "properties": {}
  }
]

```

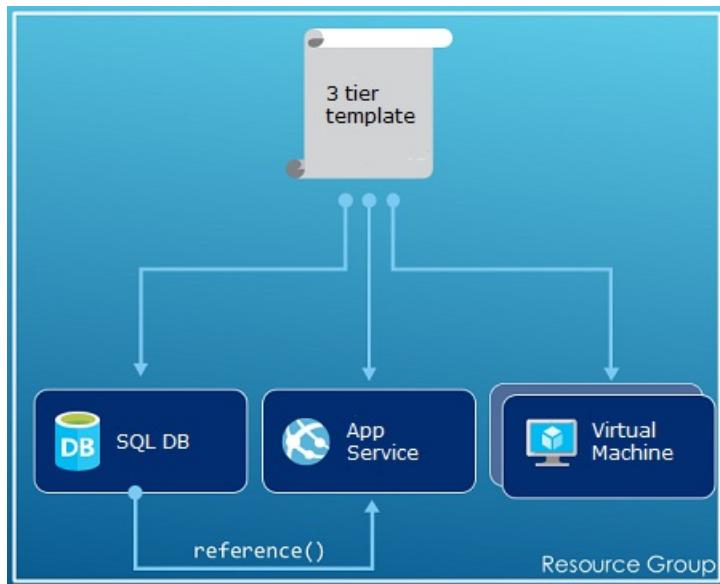
It converts the definition to the following REST API operation, which is sent to the Microsoft.Storage resource provider:

```

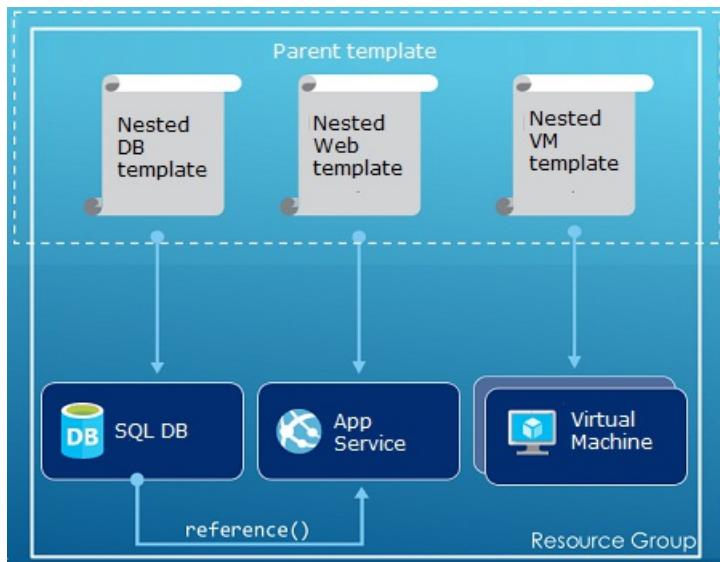
PUT
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Storage/storageAccounts/mystorageaccount?api-version=2016-01-01
REQUEST BODY
{
  "location": "westus",
  "properties": {},
  "sku": {
    "name": "Standard_LRS"
  },
  "kind": "Storage"
}

```

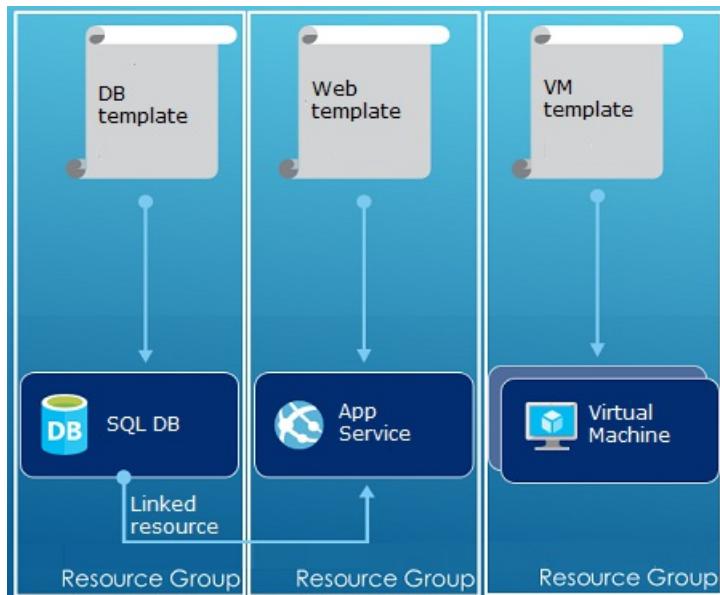
How you define templates and resource groups is entirely up to you and how you want to manage your solution. For example, you can deploy your three tier application through a single template to a single resource group.



But, you do not have to define your entire infrastructure in a single template. Often, it makes sense to divide your deployment requirements into a set of targeted, purpose-specific templates. You can easily reuse these templates for different solutions. To deploy a particular solution, you create a master template that links all the required templates. The following image shows how to deploy a three tier solution through a parent template that includes three nested templates.



If you envision your tiers having separate lifecycles, you can deploy your three tiers to separate resource groups. Notice the resources can still be linked to resources in other resource groups.



For more suggestions about designing your templates, see [Patterns for designing Azure Resource Manager templates](#). For information about nested templates, see [Using linked templates with Azure Resource Manager](#).

Azure Resource Manager analyzes dependencies to ensure resources are created in the correct order. If one resource relies on a value from another resource (such as a virtual machine needing a storage account for disks), you set a dependency. For more information, see [Defining dependencies in Azure Resource Manager templates](#).

You can also use the template for updates to the infrastructure. For example, you can add a resource to your solution and add configuration rules for the resources that are already deployed. If the template specifies creating a resource but that resource already exists, Azure Resource Manager performs an update instead of creating a new asset. Azure Resource Manager updates the existing asset to the same state as it would be as new.

Resource Manager provides extensions for scenarios when you need additional operations such as installing particular software that is not included in the setup. If you are already using a configuration management service, like DSC, Chef or Puppet, you can continue working with that service by using extensions. For information about virtual machine extensions, see [About virtual machine extensions and features](#).

Finally, the template becomes part of the source code for your app. You can check it in to your source code repository and update it as your app evolves. You can edit the template through Visual Studio.

After defining your template, you are ready to deploy the resources to Azure. For the commands to deploy the

resources, see:

- [Deploy resources with Resource Manager templates and Azure PowerShell](#)
- [Deploy resources with Resource Manager templates and Azure CLI](#)
- [Deploy resources with Resource Manager templates and Azure portal](#)
- [Deploy resources with Resource Manager templates and Resource Manager REST API](#)

Tags

Resource Manager provides a tagging feature that enables you to categorize resources according to your requirements for managing or billing. Use tags when you have a complex collection of resource groups and resources, and need to visualize those assets in the way that makes the most sense to you. For example, you could tag resources that serve a similar role in your organization or belong to the same department. Without tags, users in your organization can create multiple resources that may be difficult to later identify and manage. For example, you may wish to delete all the resources for a particular project. If those resources are not tagged for the project, you have to manually find them. Tagging can be an important way for you to reduce unnecessary costs in your subscription.

Resources do not need to reside in the same resource group to share a tag. You can create your own tag taxonomy to ensure that all users in your organization use common tags rather than users inadvertently applying slightly different tags (such as "dept" instead of "department").

The following example shows a tag applied to a virtual machine.

```
"resources": [  
  {  
    "type": "Microsoft.Compute/virtualMachines",  
    "apiVersion": "2015-06-15",  
    "name": "SimpleWindowsVM",  
    "location": "[resourceGroup().location]",  
    "tags": {  
      "costCenter": "Finance"  
    },  
    ...  
  }  
]
```

To retrieve all the resources with a tag value, use the following PowerShell cmdlet:

```
Find-AzureRmResource -TagName costCenter -TagValue Finance
```

Or, the following Azure CLI 2.0 command:

```
az resource list --tag costCenter=Finance
```

You can also view tagged resources through the Azure portal.

The [usage report](#) for your subscription includes tag names and values, which enables you to break out costs by tags. For more information about tags, see [Using tags to organize your Azure resources](#).

Access control

Resource Manager enables you to control who has access to specific actions for your organization. It natively integrates role-based access control (RBAC) into the management platform and applies that access control to all services in your resource group.

There are two main concepts to understand when working with role-based access control:

- Role definitions - describe a set of permissions and can be used in many assignments.
- Role assignments - associate a definition with an identity (user or group) for a particular scope (subscription, resource group, or resource). The assignment is inherited by lower scopes.

You can add users to pre-defined platform and resource-specific roles. For example, you can take advantage of the pre-defined role called Reader that permits users to view resources but not change them. You add users in your organization that need this type of access to the Reader role and apply the role to the subscription, resource group, or resource.

Azure provides the following four platform roles:

1. Owner - can manage everything, including access
2. Contributor - can manage everything except access
3. Reader - can view everything, but can't make changes
4. User Access Administrator - can manage user access to Azure resources

Azure also provides several resource-specific roles. Some common ones are:

1. Virtual Machine Contributor - can manage virtual machines but not grant access to them, and cannot manage the virtual network or storage account to which they are connected
2. Network Contributor - can manage all network resources, but not grant access to them
3. Storage Account Contributor - Can manage storage accounts, but not grant access to them
4. SQL Server Contributor - Can manage SQL servers and databases, but not their security-related policies
5. Website Contributor - Can manage websites, but not the web plans to which they are connected

For the full list of roles and permitted actions, see [RBAC: Built in Roles](#). For more information about role-based access control, see [Azure Role-based Access Control](#).

In some cases, you want to run code or script that accesses resources, but you do not want to run it under a user's credentials. Instead, you want to create an identity called a service principal for the application and assign the appropriate role for the service principal. Resource Manager enables you to create credentials for the application and programmatically authenticate the application. To learn about creating service principals, see one of following topics:

- [Use Azure PowerShell to create a service principal to access resources](#)
- [Use Azure CLI to create a service principal to access resources](#)
- [Use portal to create Azure Active Directory application and service principal that can access resources](#)

You can also explicitly lock critical resources to prevent users from deleting or modifying them. For more information, see [Lock resources with Azure Resource Manager](#).

Activity logs

Resource Manager logs all operations that create, modify, or delete a resource. You can use the activity logs to find an error when troubleshooting or to monitor how a user in your organization modified a resource. To see the logs, select **Activity logs** in the **Settings** blade for a resource group. You can filter the logs by many different values including which user initiated the operation. For information about working with the activity logs, see [View activity logs to manage Azure resources](#).

Customized policies

Resource Manager enables you to create customized policies for managing your resources. The types of policies you create can include diverse scenarios. You can enforce a naming convention on resources, limit which types

and instances of resources can be deployed, or limit which regions can host a type of resource. You can require a tag value on resources to organize billing by departments. You create policies to help reduce costs and maintain consistency in your subscription.

You define policies with JSON and then apply those policies either across your subscription or within a resource group. Policies are different than role-based access control because they are applied to resource types.

The following example shows a policy that ensures tag consistency by specifying that all resources include a costCenter tag.

```
{  
  "if": {  
    "not" : {  
      "field" : "tags",  
      "containsKey" : "costCenter"  
    }  
  },  
  "then" : {  
    "effect" : "deny"  
  }  
}
```

There are many more types of policies you can create. For more information, see [What is Azure Policy?](#).

SDKs

Azure SDKs are available for multiple languages and platforms. Each of these language implementations is available through its ecosystem package manager and GitHub.

Here are our Open Source SDK repositories. We welcome feedback, issues, and pull requests.

- [Azure SDK for .NET](#)
- [Azure Management Libraries for Java](#)
- [Azure SDK for Node.js](#)
- [Azure SDK for PHP](#)
- [Azure SDK for Python](#)
- [Azure SDK for Ruby](#)

For information about using these languages with your resources, see:

- [Azure for .NET developers](#)
- [Azure for Java developers](#)
- [Azure for Node.js developers](#)
- [Azure for Python developers](#)

NOTE

If the SDK doesn't provide the required functionality, you can also call to the [Azure REST API](#) directly.

Next steps

- For a simple introduction to working with templates, see [Export an Azure Resource Manager template from existing resources](#).
- For a more thorough walkthrough of creating a template, see [Create your first Azure Resource Manager template](#).

- To understand the functions you can use in a template, see [Template functions](#)
- For information about using Visual Studio with Resource Manager, see [Creating and deploying Azure resource groups through Visual Studio](#).

Here's a video demonstration of this overview:

Regions and availability for virtual machines in Azure

12/14/2017 • 7 min to read • [Edit Online](#)

Azure operates in multiple datacenters around the world. These datacenters are grouped in to geographic regions, giving you flexibility in choosing where to build your applications. It is important to understand how and where your virtual machines (VMs) operate in Azure, along with your options to maximize performance, availability, and redundancy. This article provides you with an overview of the availability and redundancy features of Azure.

What are Azure regions?

You create Azure resources in defined geographic regions like 'West US', 'North Europe', or 'Southeast Asia'. You can review the [list of regions and their locations](#). Within each region, multiple datacenters exist to provide for redundancy and availability. This approach gives you flexibility as you design applications to create VMs closest to your users and to meet any legal, compliance, or tax purposes.

Special Azure regions

Azure has some special regions that you may wish to use when building out your applications for compliance or legal purposes. These special regions include:

- **US Gov Virginia and US Gov Iowa**
 - A physical and logical network-isolated instance of Azure for US government agencies and partners, operated by screened US persons. Includes additional compliance certifications such as [FedRAMP](#) and [DISA](#). Read more about [Azure Government](#).
- **China East and China North**
 - These regions are available through a unique partnership between Microsoft and 21Vianet, whereby Microsoft does not directly maintain the datacenters. See more about [Microsoft Azure in China](#).
- **Germany Central and Germany Northeast**
 - These regions are available via a data trustee model whereby customer data remains in Germany under control of T-Systems, a Deutsche Telekom company, acting as the German data trustee.

Region pairs

Each Azure region is paired with another region within the same geography (such as US, Europe, or Asia). This approach allows for the replication of resources, such as VM storage, across a geography that should reduce the likelihood of natural disasters, civil unrest, power outages, or physical network outages affecting both regions at once. Additional advantages of region pairs include:

- In the event of a wider Azure outage, one region is prioritized out of every pair to help reduce the time to restore for applications.
- Planned Azure updates are rolled out to paired regions one at a time to minimize downtime and risk of application outage.
- Data continues to reside within the same geography as its pair (except for Brazil South) for tax and law enforcement jurisdiction purposes.

Examples of region pairs include:

PRIMARY	SECONDARY
West US	East US
North Europe	West Europe
Southeast Asia	East Asia

You can see the full [list of regional pairs here](#).

Feature availability

Some services or VM features are only available in certain regions, such as specific VM sizes or storage types. There are also some global Azure services that do not require you to select a particular region, such as [Azure Active Directory](#), [Traffic Manager](#), or [Azure DNS](#). To assist you in designing your application environment, you can check the [availability of Azure services across each region](#). You can also [programmatically query the supported VM sizes and restrictions in each region](#).

Storage availability

Understanding Azure regions and geographies becomes important when you consider the available storage replication options. Depending on the storage type, you have different replication options.

Azure Managed Disks

- Locally redundant storage (LRS)
 - Replicates your data three times within the region in which you created your storage account.

Storage account-based disks

- Locally redundant storage (LRS)
 - Replicates your data three times within the region in which you created your storage account.
- Zone redundant storage (ZRS)
 - Replicates your data three times across two to three facilities, either within a single region or across two regions.
- Geo-redundant storage (GRS)
 - Replicates your data to a secondary region that is hundreds of miles away from the primary region.
- Read-access geo-redundant storage (RA-GRS)
 - Replicates your data to a secondary region, as with GRS, but also then provides read-only access to the data in the secondary location.

The following table provides a quick overview of the differences between the storage replication types:

REPLICATION STRATEGY	LRS	ZRS	GRS	RA-GRS
Data is replicated across multiple facilities.	No	Yes	Yes	Yes
Data can be read from the secondary location and from the primary location.	No	No	No	Yes

REPLICATION STRATEGY	LRS	ZRS	GRS	RA-GRS
Number of copies of data maintained on separate nodes.	3	3	6	6

You can read more about [Azure Storage replication options here](#). For more information about managed disks, see [Azure Managed Disks overview](#).

Storage costs

Prices vary depending on the storage type and availability that you select.

Azure Managed Disks

- Premium Managed Disks are backed by Solid-State Drives (SSDs) and Standard Managed Disks are backed by regular spinning disks. Both Premium and Standard Managed Disks are charged based on the provisioned capacity for the disk.

Unmanaged disks

- Premium storage is backed by Solid-State Drives (SSDs) and is charged based on the capacity of the disk.
- Standard storage is backed by regular spinning disks and is charged based on the in-use capacity and desired storage availability.
 - For RA-GRS, there is an additional Geo-Replication Data Transfer charge for the bandwidth of replicating that data to another Azure region.

See [Azure Storage Pricing](#) for pricing information on the different storage types and availability options.

Availability sets

An availability set is a logical grouping of VMs within a datacenter that allows Azure to understand how your application is built to provide for redundancy and availability. We recommend that two or more VMs are created within an availability set to provide for a highly available application and to meet the [99.95% Azure SLA](#). When a single VM is using [Azure Premium Storage](#), the Azure SLA applies for unplanned maintenance events.

An availability set is composed of two additional groupings that protect against hardware failures and allow updates to safely be applied - fault domains (FDs) and update domains (UDs). You can read more about how to manage the availability of [Linux VMs](#) or [Windows VMs](#).

Fault domains

A fault domain is a logical group of underlying hardware that share a common power source and network switch, similar to a rack within an on-premises datacenter. As you create VMs within an availability set, the Azure platform automatically distributes your VMs across these fault domains. This approach limits the impact of potential physical hardware failures, network outages, or power interruptions.

Update domains

An update domain is a logical group of underlying hardware that can undergo maintenance or be rebooted at the same time. As you create VMs within an availability set, the Azure platform automatically distributes your VMs across these update domains. This approach ensures that at least one instance of your application always remains running as the Azure platform undergoes periodic maintenance. The order of update domains being rebooted may not proceed sequentially during planned maintenance, but only one update domain is rebooted at a time.

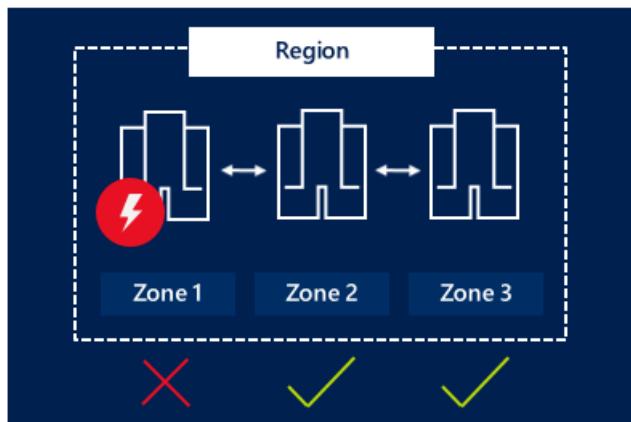
Managed Disk fault domains

For VMs using [Azure Managed Disks](#), VMs are aligned with managed disk fault domains when using a managed availability set. This alignment ensures that all the managed disks attached to a VM are within the same managed

disk fault domain. Only VMs with managed disks can be created in a managed availability set. The number of managed disk fault domains varies by region - either two or three managed disk fault domains per region. You can read more about these managed disk fault domains for [Linux VMs](#) or [Windows VMs](#).

Availability zones

[Availability zones \(preview\)](#), an alternative to availability sets, expand the level of control you have to maintain the availability of the applications and data on your VMs. An Availability Zone is a physically separate zone within an Azure region. There are three Availability Zones per supported Azure region. Each Availability Zone has a distinct power source, network, and cooling, and is logically separate from the other Availability Zones within the Azure region. By architecting your solutions to use replicated VMs in zones, you can protect your apps and data from the loss of a datacenter. If one zone is compromised, then replicated apps and data are instantly available in another zone.



NOTE

Azure Availability Zones is in preview and is ready for your development and test scenarios. Support is available for select Azure resources, regions, and size families. For more information on how to get started, and which Azure resources, regions, and size families you can try with Availability Zones, see [Overview of Availability Zones](#). You can [provide feedback](#) on the Azure website. For support, contact [StackOverflow](#) or [open an Azure support ticket](#).

Learn more about deploying a [Windows](#) or [Linux](#) VM in an Availability Zone.

Next steps

You can now start to use these availability and redundancy features to build your Azure environment. For best practices information, see [Azure availability best practices](#).

Sizes for Windows virtual machines in Azure

12/15/2017 • 1 min to read • [Edit Online](#)

This article describes the available sizes and options for the Azure virtual machines you can use to run your Windows apps and workloads. It also provides deployment considerations to be aware of when you're planning to use these resources. This article is also available for [Linux virtual machines](#).

Type	Sizes	Description
General purpose	B, Dsv3, Dv3, DSv2, Dv2, DS, D, Av2, A0-7	Balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers.
Compute optimized	Fsv2, Fs, F	High CPU-to-memory ratio. Good for medium traffic web servers, network appliances, batch processes, and application servers.
Memory optimized	Esv3, Ev3, M, GS, G, DSv2, DS, Dv2, D	High memory-to-CPU ratio. Great for relational database servers, medium to large caches, and in-memory analytics.
Storage optimized	Ls	High disk throughput and IO. Ideal for Big Data, SQL, and NoSQL databases.
GPU	NV, NC, NCv2, ND	Specialized virtual machines targeted for heavy graphic rendering and video editing. Available with single or multiple GPUs.
High performance compute	H, A8-11	Our fastest and most powerful CPU virtual machines with optional high-throughput network interfaces (RDMA).

- For information about pricing of the various sizes, see [Virtual Machines Pricing](#).
- To see general limits on Azure VMs, see [Azure subscription and service limits, quotas, and constraints](#).
- Storage costs are calculated separately based on used pages in the storage account. For details, [Azure Storage Pricing](#).
- Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Rest API

For information on using the REST API to query for VM sizes, see the following:

- [List available virtual machine sizes for resizing](#)
- [List available virtual machine sizes for a subscription](#)
- [List available virtual machine sizes in an availability set](#)

ACU

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Next steps

Learn more about the different VM sizes that are available:

- [General purpose](#)
- [Compute optimized](#)
- [Memory optimized](#)
- [Storage optimized](#)
- [GPU optimized](#)
- [High performance compute](#)

General purpose virtual machine sizes

11/16/2017 • 9 min to read • [Edit Online](#)

General purpose VM sizes provide balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers. This article provides information about the number of vCPUs, data disks and NICs as well as storage throughput and network bandwidth for each size in this grouping.

- The A-series and Av2-series VMs can be deployed on a variety of hardware types and processors. The size is throttled, based upon the hardware, to offer consistent processor performance for the running instance, regardless of the hardware it is deployed on. To determine the physical hardware on which this size is deployed, query the virtual hardware from within the Virtual Machine.
- D-series VMs are designed to run applications that demand higher compute power and temporary disk performance. D-series VMs provide faster processors, a higher memory-to-vCPU ratio, and a solid-state drive (SSD) for the temporary disk. For details, see the announcement on the Azure blog, [New D-Series Virtual Machine Sizes](#).
- Dv3-series, Dv2-series, a follow-on to the original D-series, features a more powerful CPU. The Dv2-series CPU is about 35% faster than the D-series CPU. It is based on the latest generation 2.4 GHz Intel Xeon® E5-2673 v3 (Haswell) processor, and with the Intel Turbo Boost Technology 2.0, can go up to 3.1 GHz. The Dv2-series has the same memory and disk configurations as the D-series.
- The basic tier sizes are primarily for development workloads and other applications that don't require load balancing, auto-scaling, or memory-intensive virtual machines. For information about VM sizes that are more appropriate for production applications, see [\(Sizes for virtual machines\)\[virtual-machines-size-specs.md\]](#) and for VM pricing information, see [Virtual Machines Pricing](#).

B-series

The B-series burstable VMs are ideal for workloads that do not need the full performance of the CPU continuously, like web servers, small databases and development and test environments. These workloads typically have burstable performance requirements. The B-Series provides these customers the ability to purchase a VM size with a price conscience baseline performance that allows the VM instance to build up credits when the VM is utilizing less than its base performance. When the VM has accumulated credit, the VM can burst above the VM's baseline using up to 100% of the CPU when your application requires the higher CPU performance.

SIZE	VCPU	MEMORY: GIB	LOCAL SSD: GIB	BASE PERF OF A CORE	CREDIT S BANKE D / HOUR	MAX BANKE D CREDIT S	MAX DATA DISKS	MAX LOCAL DISK PERF: IOPS / MBPS	MAX UNCAC HED DISK PERF: IOPS / MBPS	MAX NICS
Standard_B1s	1	1	4	10%	6	144	2	400 / 10	320 / 10	2
Standard_B1ms	1	2	4	20%	12	288	2	800 / 10	640 / 10	2
Standard_B2s	2	4	8	40%	24	576	4	1600 / 15	1280 / 15	3

SIZE	VCPU	MEMORY: GIB	LOCAL SSD: GIB	BASE PERF OF A CORE	CREDIT S BANKE D / HOUR	MAX BANKED CREDIT S	MAX DATA DISKS	MAX LOCAL DISK PERF: IOPS / MBPS	MAX UNCAC HED DISK PERF: IOPS / MBPS	MAX NICS
Standard_B2_ms	2	8	16	60%	36	864	4	2400 / 22.5	1920 / 22.5	3
Standard_B4_ms	4	16	32	90%	54	1296	8	3600 / 35	2880 / 35	4
Standard_B8_ms	8	32	64	135%	81	1944	16	4320 / 50	4320 / 50	4

Dsv3-series ¹

ACU: 160-190

Dsv3-series sizes are based on the 2.4 GHz Intel Xeon® E5-2673 v3 (Haswell) processor or the latest 2.3 GHz Intel XEON ® E5-2673 v4 (Broadwell) processor that can achieve 3.5GHz with Intel Turbo Boost Technology 2.0 and use premium storage. The Dsv3-series sizes offer a combination of vCPU, memory, and temporary storage for most production workloads.

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICS / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D 2s_v3	2	8	16	4	4,000 / 32 (50)	3,200 / 48	2 / moderate
Standard_D 4s_v3	4	16	32	8	8,000 / 64 (100)	6,400 / 96	2 / moderate
Standard_D 8s_v3	8	32	64	16	16,000 / 128 (200)	12,800 / 192	4 / high
Standard_D 16s_v3	16	64	128	32	32,000 / 256 (400)	25,600 / 384	8 / high
Standard_D 32s_v3	32	128	256	32	64,000 / 512 (800)	51,200 / 768	8 / Extremely high
Standard_D 64s_v3	64	256	512	32	128,000 / 1024 (1600)	80,000 / 1200	8 / Extremely high

¹ Dsv3-series VM's feature Intel® Hyper-Threading Technology

Dv3-series¹

ACU: 160-190

Dv3-series sizes are based on the 2.4 GHz Intel Xeon® E5-2673 v3 (Haswell) processor or 2.3 GHz Intel XEON ® E5-2673 v4 (Broadwell) processor that can achieve 3.5GHz with Intel Turbo Boost Technology 2.0. The Dv3-series sizes offer a combination of vCPU, memory, and temporary storage for most production workloads.

Data disk storage is billed separately from virtual machines. To use premium storage disks, use the Dsv3 sizes. The pricing and billing meters for Dsv3 sizes are the same as Dv3-series.

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_D2_v3	2	8	50	4	3000/46/23	2 / moderate
Standard_D4_v3	4	16	100	8	6000/93/46	2 / moderate
Standard_D8_v3	8	32	200	16	12000/187/93	4 / high
Standard_D16_v3	16	64	400	32	24000/375/187	8 / high
Standard_D32_v3	32	128	800	32	48000/750/375	8 / Extremely high
Standard_D64_v3	64	256	1600	32	96000/1000/500	8 / Extremely high

¹ Dv3-series VM's feature Intel® Hyper-Threading Technology

DSv2-series

ACU: 210-250

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D S1_v2	1	3.5	7	4	4,000 / 32 (43)	3,200 / 48	2 / 750
Standard_D S2_v2	2	7	14	8	8,000 / 64 (86)	6,400 / 96	2 / 1500
Standard_D S3_v2	4	14	28	16	16,000 / 128 (172)	12,800 / 192	4 / 3000

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D S4_v2	8	28	56	32	32,000 / 256 (344)	25,600 / 384	8 / 6000
Standard_D S5_v2	16	56	112	64	64,000 / 512 (688)	51,200 / 768	8 / 6000 - 12000 +

Dv2-series

ACU: 210-250

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D1_v2	1	3.5	50	3000 / 46 / 23	4 / 4x500	2 / 750
Standard_D2_v2	2	7	100	6000 / 93 / 46	8 / 8x500	2 / 1500
Standard_D3_v2	4	14	200	12000 / 187 / 93	16 / 16x500	4 / 3000
Standard_D4_v2	8	28	400	24000 / 375 / 187	32 / 32x500	8 / 6000
Standard_D5_v2	16	56	800	48000 / 750 / 375	64 / 64x500	8 / 6000 - 12000 +

DS-series

ACU: 160

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D S1	1	3.5	7	4	4,000 / 32 (43)	3,200 / 32	2 / 500

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D S2	2	7	14	8	8,000 / 64 (86)	6,400 / 64	2 / 1000
Standard_D S3	4	14	28	16	16,000 / 128 (172)	12,800 / 128	4 / 2000
Standard_D S4	8	28	56	32	32,000 / 256 (344)	25,600 / 256	8 / 4000

D-series

ACU: 160

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D1	1	3.5	50	3000 / 46 / 23	4 / 4x500	2 / 500
Standard_D2	2	7	100	6000 / 93 / 46	8 / 8x500	2 / 1000
Standard_D3	4	14	200	12000 / 187 / 93	16 / 16x500	4 / 2000
Standard_D4	8	28	400	24000 / 375 / 187	32 / 32x500	8 / 4000

Av2-series

ACU: 100

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_A1_v2	1	2	10	1000 / 20 / 10	2 / 2x500	2 / 250

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_A2_v2	2	4	20	2000 / 40 / 20	4 / 4x500	2 / 500
Standard_A4_v2	4	8	40	4000 / 80 / 40	8 / 8x500	4 / 1000
Standard_A8_v2	8	16	80	8000 / 160 / 80	16 / 16x500	8 / 2000
Standard_A2_m_v2	2	16	20	2000 / 40 / 20	4 / 4x500	2 / 500
Standard_A4_m_v2	4	32	40	4000 / 80 / 40	8 / 8x500	4 / 1000
Standard_A8_m_v2	8	64	80	8000 / 160 / 80	16 / 16x500	8 / 2000

A-series

ACU: 50-100

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (HDD): GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_A0 ¹	1	0.768	20	1	1x500	2 / 100
Standard_A1	1	1.75	70	2	2x500	2 / 500
Standard_A2	2	3.5	135	4	4x500	2 / 500
Standard_A3	4	7	285	8	8x500	2 / 1000
Standard_A4	8	14	605	16	16x500	4 / 2000
Standard_A5	2	14	135	4	4x500	2 / 500
Standard_A6	4	28	285	8	8x500	2 / 1000
Standard_A7	8	56	605	16	16x500	4 / 2000

¹ The A0 size is over-subscribed on the physical hardware. For this specific size only, other customer deployments may impact the performance of your running workload. The relative performance is outlined below as the

expected baseline, subject to an approximate variability of 15 percent.

Standard A0 - A4 using CLI and PowerShell

In the classic deployment model, some VM size names are slightly different in CLI and PowerShell:

- Standard_A0 is ExtraSmall
- Standard_A1 is Small
- Standard_A2 is Medium
- Standard_A3 is Large
- Standard_A4 is ExtraLarge

Basic A

SIZE – SIZE\NAME	VCPU	MEMORY	NICS (MAX)	MAX TEMPORARY DISK SIZE	MAX. DATA DISKS 1023 GB EACH)	MAX. IOPS (300 PER DISK)
A0\Basic_A0	1	768 MB	2	20 GB	1	1x300
A1\Basic_A1	1	1.75 GB	2	40 GB	2	2x300
A2\Basic_A2	2	3.5 GB	2	60 GB	4	4x300
A3\Basic_A3	4	7 GB	2	120 GB	8	8x300
A4\Basic_A4	8	14 GB	2	240 GB	16	16x300

Size table definitions

- Storage capacity is shown in units of GiB or 1024^3 bytes. When comparing disks measured in GB (1000^3 bytes) to disks measured in GiB (1024^3) remember that capacity numbers given in GiB may appear smaller. For example, 1023 GiB = 1098.4 GB
- Disk throughput is measured in input/output operations per second (IOPS) and MBps where MBps = 10^6 bytes/sec.
- Data disks can operate in cached or uncached modes. For cached data disk operation, the host cache mode is set to **ReadOnly** or **ReadWrite**. For uncached data disk operation, the host cache mode is set to **None**.
- If you want to get the best performance for your VMs, you should limit the number of data disks to 2 disks per vCPU.
- **Expected network bandwidth** is the maximum aggregated [bandwidth allocated per VM type](#) across all NICs, for all destinations. Upper limits are not guaranteed, but are intended to provide guidance for selecting the right VM type for the intended application. Actual network performance will depend on a variety of factors including network congestion, application loads, and network settings. For information on optimizing network throughput, see [Optimizing network throughput for Windows and Linux](#). To achieve the expected network performance on Linux or Windows, it may be necessary to select a specific version or optimize your VM. For more information, see [How to reliably test for virtual machine throughput](#).
- † 16 vCPU performance will consistently reach the upper limit in an upcoming release.

Other sizes

- [Compute optimized](#)
- [Memory optimized](#)

- Storage optimized
- GPU optimized
- High performance compute

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

B-series burstable virtual machine sizes

12/6/2017 • 3 min to read • [Edit Online](#)

The B-series VM family allows you to choose which VM size provides you the necessary base level performance for your workload, with the ability to burst CPU performance up to 100% of an Intel® Broadwell E5-2673 v4 2.3GHz, or an Intel® Haswell 2.4 GHz E5-2673 v3 processor vCPU.

The B-series VMs are ideal for workloads that do not need the full performance of the CPU continuously, like web servers, small databases and development and test environments. These workloads typically have burstable performance requirements. The B-series provides you with the ability to purchase a VM size with baseline performance and the VM instance builds up credits when it is using less than its baseline. When the VM has accumulated credit, the VM can burst above the baseline using up to 100% of the vCPU when your application requires higher CPU performance.

The B-series comes in the following six VM sizes:

SIZE	VCPU'S	MEMORY: GIB	TEMP STORAGE (SSD) GIB	BASE CPU PERF OF VM	MAX CPU PERF OF VM	CREDITS BANKED / HOUR	MAX BANKED CREDITS
Standard_B_1s	1	1	4	10%	100%	6	144
Standard_B_1ms	1	2	4	20%	100%	12	288
Standard_B_2s	2	4	8	40%	200%	24	576
Standard_B_2ms	2	8	16	60%	200%	36	864
Standard_B_4ms	4	16	32	90%	400%	54	1296
Standard_B_8ms	8	32	64	135%	800%	81	1944

Q & A

Q: How do you get 135% baseline performance from a VM?

A: The 135% is shared amongst the 8 vCPU's that make up the VM size. For example, if your application leverages 4 of the 8 cores working on batch processing and each of those 4 vCPU's are running at 30% utilization the total amount of VM CPU performance would equal 120%. Meaning that your VM would be building credit time based on the 15% delta from your baseline performance. But it also means that when you have credits available that same VM can use 100% of all 8 vCPU's giving that VM a Max CPU performance of 800%.

Q: How can I monitor my credit balance and consumption

A: We will be introducing 2 new metrics in the coming weeks, the **Credit** metric will allow you to view how many credits your VM has banked and the **ConsumedCredit** metric will show how many CPU credits your VM has consumed from the bank. You will be able to view these metrics from the metrics pane in the portal or

programmatically through the Azure Monitor APIs.

For more information on how to access the metrics data for Azure, see [Overview of metrics in Microsoft Azure](#).

Q: How are credits accumulated?

A: The VM accumulation and consumption rates are set such that a VM running at exactly its base performance level will have neither a net accumulation or consumption of bursting credits. A VM will have a net increase in credits whenever it is running below its base performance level and will have a net decrease in credits whenever the VM is utilizing the CPU more than its base performance level.

Example: I deploy a VM using the B1ms size for my small time and attendance database application. This size allows my application to use up to 20% of a vCPU as my baseline, which is .2 credits per minute I can use or bank.

My application is busy at the beginning and end of my employees work day, between 7:00-9:00 AM and 4:00 - 6:00PM. During the other 20 hours of the day, my application is typically at idle, only using 10% of the vCPU. For the non-peak hours I earn 0.2 credits per minute but only consume 0.1 credits per minute, so my VM will bank $.1 \times 60 = 6$ credits per hour. For the 20 hours that I am off-peak, I will bank 120 credits.

During peak hours my application averages 60% vCPU utilization, I still earn 0.2 credits per minute but I consume 0.6 credits per minute, for a net cost of .4 credits a minute or $.4 \times 60 = 24$ credits per hour. I have 4 hours per day of peak usage, so it costs $4 \times 24 = 96$ credits for my peak usage.

If I take the 120 credits I earned off-peak and subtract the 96 credits I used for my peak times, I bank an additional 24 credits per day that I can use for other bursts of activity.

Q: Does the B-Series support Premium Storage data disks?

A: Yes, all B-Series sizes support Premium Storage data disks.

Other sizes

- [General purpose](#)
- [Compute optimized](#)
- [Memory optimized](#)
- [Storage optimized](#)
- [GPU optimized](#)
- [High performance compute](#)

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Compute optimized virtual machine sizes

11/16/2017 • 4 min to read • [Edit Online](#)

Compute optimized VM sizes have a high CPU-to-memory ratio and are good for medium traffic web servers, network appliances, batch processes, and application servers. This article provides information about the number of vCPUs, data disks, and NICs as well as storage throughput and network bandwidth for each size in this grouping.

Fsv2-series is based on the Intel® Xeon® Platinum 8168 processor, featuring a base core frequency of 2.7 GHz and a maximum single-core turbo frequency of 3.7 GHz. Intel® AVX-512 instructions, which are new on Intel Scalable Processors, will provide up to a 2X performance boost to vector processing workloads on both single and double precision floating point operations. In other words, they are really fast for any computational workload.

At a lower per-hour list price, the Fsv2-series is the best value in price-performance in the Azure portfolio based on the Azure Compute Unit (ACU) per vCPU.

F-series is based on the 2.4 GHz Intel Xeon® E5-2673 v3 (Haswell) processor, which can achieve clock speeds as high as 3.1 GHz with the Intel Turbo Boost Technology 2.0. This is the same CPU performance as the Dv2-series of VMs.

F-series VMs are an excellent choice for workloads that demand faster CPUs but do not need as much memory or temporary storage per vCPU. Workloads such as analytics, gaming servers, web servers, and batch processing will benefit from the value of the F-series.

The Fs-series provides all the advantages of the F-series, in addition to Premium storage.

Fsv2-series¹

ACU: 195 - 210

SIZE	VCPU'S	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_F2s_v2	2	4	16	4	4000 (32)	Moderate
Standard_F4s_v2	4	8	32	8	8000 (64)	Moderate
Standard_F8s_v2	8	16	64	16	16000 (128)	High
Standard_F16s_v2	16	32	128	32	32000 (256)	High
Standard_F32s_v2	32	64	256	32	64000 (512)	Extremely High

SIZE	VCPU'S	MEMORY: GiB	TEMP STORAGE (SSD) GiB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GiB)	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_F64s_v2	64	128	512	32	128000 (1024)	Extremely High
Standard_F72s_v2	72	144	576	32	144000 (1520)	Extremely High

¹Fsv2-series VM's feature Intel® Hyper-Threading Technology

Fs-series ¹

ACU: 210 - 250

SIZE	VCPU	MEMORY: GiB	TEMP STORAGE (SSD) GiB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GiB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_F1s	1	2	4	4	4,000 / 32 (12)	3,200 / 48	2 / 750
Standard_F2s	2	4	8	8	8,000 / 64 (24)	6,400 / 96	2 / 1500
Standard_F4s	4	8	16	16	16,000 / 128 (48)	12,800 / 192	4 / 3000
Standard_F8s	8	16	32	32	32,000 / 256 (96)	25,600 / 384	8 / 6000
Standard_F16s	16	32	64	64	64,000 / 512 (192)	51,200 / 768	8 / 6000-12000 +

MBps = 10^6 bytes per second, and GiB = 1024^3 bytes.

¹ The maximum disk throughput (IOPS or MBps) possible with a Fs series VM may be limited by the number, size and striping of the attached disk(s). For details, see [Premium Storage: High-performance storage for Azure virtual machine workloads](#).

F-series

ACU: 210 - 250

SIZE	VCPU	MEMORY: GiB	TEMP STORAGE (SSD) GiB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_F1	1	2	16	3000 / 46 / 23	4 / 4x500	2 / 750
Standard_F2	2	4	32	6000 / 93 / 46	8 / 8x500	2 / 1500
Standard_F4	4	8	64	12000 / 187 / 93	16 / 16x500	4 / 3000
Standard_F8	8	16	128	24000 / 375 / 187	32 / 32x500	8 / 6000
Standard_F16	16	32	256	48000 / 750 / 375	64 / 64x500	8 / 6000 - 12000 †

Size table definitions

- Storage capacity is shown in units of GiB or 1024^3 bytes. When comparing disks measured in GB (1000^3 bytes) to disks measured in GiB (1024^3) remember that capacity numbers given in GiB may appear smaller. For example, 1023 GiB = 1098.4 GB
- Disk throughput is measured in input/output operations per second (IOPS) and MBps where MBps = 10^6 bytes/sec.
- Data disks can operate in cached or uncached modes. For cached data disk operation, the host cache mode is set to **ReadOnly** or **ReadWrite**. For uncached data disk operation, the host cache mode is set to **None**.
- If you want to get the best performance for your VMs, you should limit the number of data disks to 2 disks per vCPU.
- Expected network bandwidth** is the maximum aggregated [bandwidth allocated per VM type](#) across all NICs, for all destinations. Upper limits are not guaranteed, but are intended to provide guidance for selecting the right VM type for the intended application. Actual network performance will depend on a variety of factors including network congestion, application loads, and network settings. For information on optimizing network throughput, see [Optimizing network throughput for Windows and Linux](#). To achieve the expected network performance on Linux or Windows, it may be necessary to select a specific version or optimize your VM. For more information, see [How to reliably test for virtual machine throughput](#).
- † 16 vCPU performance will consistently reach the upper limit in an upcoming release.

Other sizes

- [General purpose](#)
- [Memory optimized](#)
- [Storage optimized](#)
- [GPU optimized](#)
- [High performance compute](#)

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Memory optimized virtual machine sizes

11/16/2017 • 9 min to read • [Edit Online](#)

Memory optimized VM sizes offer a high memory-to-CPU ratio that are great for relational database servers, medium to large caches, and in-memory analytics. This article provides information about the number of vCPUs, data disks and NICs as well as storage throughput and network bandwidth for each size in this grouping.

- The M-Series offers the highest vCPU count (up to 128 vCPUs) and largest memory (up to 3.8 TiB) of any VM in the cloud. It's ideal for extremely large databases or other applications that benefit from high vCPU counts and large amounts of memory.
- Dv2-series, D-series, G-series, and the DS/GS counterparts are ideal for applications that demand faster vCPUs, better temporary storage performance, or have higher memory demands. They offer a powerful combination for many enterprise-grade applications.
- D-series VMs are designed to run applications that demand higher compute power and temporary disk performance. D-series VMs provide faster processors, a higher memory-to-vCPU ratio, and a solid-state drive (SSD) for temporary storage. For details, see the announcement on the Azure blog, [New D-Series Virtual Machine Sizes](#).
- Dv2-series, a follow-on to the original D-series, features a more powerful CPU. The Dv2-series CPU is about 35% faster than the D-series CPU. It is based on the latest generation 2.4 GHz Intel Xeon® E5-2673 v3 (Haswell) processor, and with the Intel Turbo Boost Technology 2.0, can go up to 3.1 GHz. The Dv2-series has the same memory and disk configurations as the D-series.

ESv3-series¹

ACU: 160-190

ESv3-series instances are based on the 2.3 GHz Intel XEON® E5-2673 v4 (Broadwell) processor and can achieve 3.5GHz with Intel Turbo Boost Technology 2.0 and use premium storage. Ev3-series instances are ideal for memory-intensive enterprise applications.

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUHPUT: IOPS / MBPS	MAX NICS / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_E_2s_v3	2	16	32	4	4,000 / 32 (50)	3,200 / 48	2 / moderate
Standard_E_4s_v3	4	32	64	8	8,000 / 64 (100)	6,400 / 96	2 / moderate
Standard_E_8s_v3	8	64	128	16	16,000 / 128 (200)	12,800 / 192	4 / high
Standard_E_16s_v3	16	128	256	32	32,000 / 256 (400)	25,600 / 384	8 / high

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_E32s_v3 ²	32	256	512	32	64,000 / 512 (800)	51,200 / 768	8 / extremely high
Standard_E64s_v3 ²	64	432	864	32	128,000/1024 (1600)	80,000 / 1200	8 / extremely high

¹ Esv3-series VM's feature Intel® Hyper-Threading Technology ² Constrained core sizes available

Ev3-series ¹

ACU: 160 - 190

Ev3-series instances are based on the 2.3 GHz Intel XEON ® E5-2673 v4 (Broadwell) processor and can achieve 3.5GHz with Intel Turbo Boost Technology 2.0. Ev3-series instances are ideal for memory-intensive enterprise applications.

Data disk storage is billed separately from virtual machines. To use premium storage disks, use the ESv3 sizes. The pricing and billing meters for ESv3 sizes are the same as Ev3-series.

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_E2_v3	2	16	50	4	3000/46/23	2 / moderate
Standard_E4_v3	4	32	100	8	6000/93/46	2 / moderate
Standard_E8_v3	8	64	200	16	12000/187/93	4 / high
Standard_E16_v3	16	128	400	32	24000/375/187	8 / high
Standard_E32_v3	32	256	800	32	48000/750/375	8 / extremely high
Standard_E64_v3	64	432	1600	32	96000/1000/500	8 / extremely high

¹ Ev3-series VM's feature Intel® Hyper-Threading Technology

M-series ¹

ACU: 160-180

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_M 64s	64	1024	2048	64	80,000 / 800 (6348)	40,000 / 1,000	8 / 16000
Standard_M 64ms	64	1792	2048	64	80,000 / 800 (6348)	40,000 / 1,000	8 / 16000
Standard_M 128s ^{2,3}	128	2048	4096	64	160,000 / 1,600 (12,696)	80,000 / 2,000	8 / 25000
Standard_M 128ms ^{2,3}	128	3800	4096	64	160,000 / 1,600 (12,696)	80,000 / 2,000	8 / 25000

¹ M-series VM's feature Intel® Hyper-Threading Technology

² More than 64 vCPU's require one of these supported guest OSes: Windows Server 2016, Ubuntu 16.04 LTS, SLES 12 SP2, and Red Hat Enterprise Linux or CentOS 7.3 with LIS 4.2.1

³ Constrained core sizes available.

GS-series¹

ACU: 180 - 240

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_G S1	2	28	56	8	10,000 / 100 (264)	5,000 / 125	2 / 2000
Standard_G S2	4	56	112	16	20,000 / 200 (528)	10,000 / 250	2 / 4000
Standard_G S3	8	112	224	32	40,000 / 400 (1,056)	20,000 / 500	4 / 8000
Standard_G S4 ³	16	224	448	64	80,000 / 800 (2,112)	40,000 / 1,000	8 / 6000 - 16000 †

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_GS5 ^{2, 3}	32	448	896	64	160,000 / 1,600 (4,224)	80,000 / 2,000	8 / 20000

¹ The maximum disk throughput (IOPS or MBps) possible with a GS series VM may be limited by the number, size and striping of the attached disk(s). For details, see [Premium Storage: High-performance storage for Azure virtual machine workloads](#).

² Instance is isolated to hardware dedicated to a single customer.

³ Constrained core sizes available

G-series

ACU: 180 - 240

SIZE	VCPUs	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_G1	2	28	384	6000 / 93 / 46	8 / 8 x 500	2 / 2000
Standard_G2	4	56	768	12000 / 187 / 93	16 / 16 x 500	2 / 4000
Standard_G3	8	112	1,536	24000 / 375 / 187	32 / 32 x 500	4 / 8000
Standard_G4	16	224	3,072	48000 / 750 / 375	64 / 64 x 500	8 / 6000 - 16000 †
Standard_G5 ¹	32	448	6,144	96000 / 1500 / 750	64 / 64 x 500	8 / 20000

¹ Instance is isolated to hardware dedicated to a single customer.

DSv2-series¹

ACU: 210 - 250

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D S11_v2	2	14	28	8	8,000 / 64 (72)	6,400 / 96	2 / 1500
Standard_D S12_v2	4	28	56	16	16,000 / 128 (144)	12,800 / 192	4 / 3000
Standard_D S13_v2	8	56	112	32	32,000 / 256 (288)	25,600 / 384	8 / 6000
Standard_D S14_v2	16	112	224	64	64,000 / 512 (576)	51,200 / 768	8 / 6000 - 12000 [†]
Standard_D S15_v2 ²	20	140	280	64	80,000 / 640 (720)	64,000 / 960	8 / 20000 ³

¹ The maximum disk throughput (IOPS or MBps) possible with a DSv2 series VM may be limited by the number, size and striping of the attached disk(s). For details, see [Premium Storage: High-performance storage for Azure virtual machine workloads](#).

² Instance is an isolated node that guarantees that your VM is the only VM on our Intel Haswell node.

³ 25000 Mbps with Accelerated Networking.

Dv2-series

ACU: 210 - 250

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D11_v2	2	14	100	6000 / 93 / 46	8 / 8x500	2 / 1500
Standard_D12_v2	4	28	200	12000 / 187 / 93	16 / 16x500	4 / 3000
Standard_D13_v2	8	56	400	24000 / 375 / 187	32 / 32x500	8 / 6000
Standard_D14_v2	16	112	800	48000 / 750 / 375	64 / 64x500	8 / 6000 - 12000 [†]
Standard_D15_v2 ¹	20	140	1,000	60000 / 937 / 468	64 / 64x500	8 / 20000 ²

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
------	------	-------------	------------------------	--	-----------------------------------	--

¹ Instance is an isolated node that guarantees that your VM is the only VM on our Intel Haswell node.

² 25000 Mbps with Accelerated Networking.

DS-series ¹

ACU: 160

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX CACHED AND TEMP STORAGE THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D S11	2	14	28	8	8,000 / 64 (72)	6,400 / 64	2 / 1000
Standard_D S12	4	28	56	16	16,000 / 128 (144)	12,800 / 128	4 / 2000
Standard_D S13	8	56	112	32	32,000 / 256 (288)	25,600 / 256	8 / 4000
Standard_D S14	16	112	224	64	64,000 / 512 (576)	51,200 / 512	8 / 6000 - 8000 +

¹ The maximum disk throughput (IOPS or MBps) possible with a DS series VM may be limited by the number, size and striping of the attached disk(s). For details, see [Premium Storage: High-performance storage for Azure virtual machine workloads](#).

D-series

ACU: 160

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D11	2	14	100	6000 / 93 / 46	8 / 8x500	2 / 1000

SIZE	VCPU	MEMORY: GiB	TEMP STORAGE (SSD) GiB	MAX TEMP STORAGE THROUGHPUT: IOPS / READ MBPS / WRITE MBPS	MAX DATA DISKS / THROUGHPUT: IOPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_D12	4	28	200	12000 / 187 / 93	16 / 16x500	4 / 2000
Standard_D13	8	56	400	24000 / 375 / 187	32 / 32x500	8 / 4000
Standard_D14	16	112	800	48000 / 750 / 375	64 / 64x500	8 / 6000 - 8000 +

Size table definitions

- Storage capacity is shown in units of GiB or 1024^3 bytes. When comparing disks measured in GB (1000^3 bytes) to disks measured in GiB (1024^3) remember that capacity numbers given in GiB may appear smaller. For example, 1023 GiB = 1098.4 GB
- Disk throughput is measured in input/output operations per second (IOPS) and MBps where MBps = 10^6 bytes/sec.
- Data disks can operate in cached or uncached modes. For cached data disk operation, the host cache mode is set to **ReadOnly** or **ReadWrite**. For uncached data disk operation, the host cache mode is set to **None**.
- If you want to get the best performance for your VMs, you should limit the number of data disks to 2 disks per vCPU.
- Expected network bandwidth** is the maximum aggregated [bandwidth allocated per VM type](#) across all NICs, for all destinations. Upper limits are not guaranteed, but are intended to provide guidance for selecting the right VM type for the intended application. Actual network performance will depend on a variety of factors including network congestion, application loads, and network settings. For information on optimizing network throughput, see [Optimizing network throughput for Windows and Linux](#). To achieve the expected network performance on Linux or Windows, it may be necessary to select a specific version or optimize your VM. For more information, see [How to reliably test for virtual machine throughput](#).
- + 16 vCPU performance will consistently reach the upper limit in an upcoming release.

Other sizes

- [General purpose](#)
- [Compute optimized](#)
- [Storage optimized](#)
- [GPU optimized](#)
- [High performance compute](#)

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Constrained vCPU capable VM sizes

12/8/2017 • 1 min to read • [Edit Online](#)

Some database workloads like SQL Server or Oracle require high memory, storage, and I/O bandwidth, but not a high core count. Many database workloads are not CPU-intensive. Azure offers certain VM sizes where you can constrain the VM vCPU count to reduce the cost of software licensing, while maintaining the same memory, storage, and I/O bandwidth.

The vCPU count can be constrained to one half or one quarter of the original VM size. These new VM sizes have a suffix that specifies the number of active vCPUs to make them easier for you to identify.

For example, the current VM size Standard_GS5 comes with 32 vCPUs, 448 GB RAM, 64 disks (up to 256 TB), and 80,000 IOPs or 2 GB/s of I/O bandwidth. The new VM sizes Standard_GS5-16 and Standard_GS5-8 comes with 16 and 8 active vCPUs respectively, while maintaining the rest of the specs of the Standard_GS5 for memory, storage, and I/O bandwidth.

The licensing fees charged for SQL Server or Oracle are constrained to the new vCPU count, and other products should be charged based on the new vCPU count. This results in a 50% to 75% increase in the ratio of the VM specs to active (billable) vCPUs. These new VM sizes that are only available in Azure, allowing workloads to push higher CPU utilization at a fraction of the (per-core) licensing cost. At this time, the compute cost, which includes OS licensing, remains the same one as the original size. For more information, see [Azure VM sizes for more cost-effective database workloads](#).

NAME	VCPUs	SPECS
Standard_M64-32ms	32	Same as M64ms
Standard_M64-16ms	16	Same as M64ms
Standard_M128-64ms	64	Same as M128ms
Standard_M128-32ms	32	Same as M128ms
Standard_E32-16_v3	16	Same as E32s_v3
Standard_E32-8s_v3	8	Same as E32s_v3
Standard_E64-32s_v3	32	Same as E64s_v3
Standard_E64-16s_v3	16	Same as E64s_v3
Standard_GS4-8	8	Same as GS4
Standard_GS4-4	4	Same as GS4
Standard_GS5-16	16	Same as GS5
Standard_GS5-8	8	Same as GS5
Standard_DS13-4_v2	4	Same as DS13_v2

NAME	VCPUs	SPECS
Standard_DS13-2_v2	2	Same as DS13_v2
Standard_DS14-8_v2	8	Same as DS14_v2
Standard_DS14-4_v2	4	Same as DS14_v2

Other sizes

- [Compute optimized](#)
- [Memory optimized](#)
- [Storage optimized](#)
- [GPU](#)
- [High performance compute](#)

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Storage optimized virtual machine sizes

11/30/2017 • 2 min to read • [Edit Online](#)

Storage optimized VM sizes offer high disk throughput and IO, and are ideal for Big Data, SQL, and NoSQL databases. This article provides information about the number of vCPUs, data disks and NICs as well as storage throughput and network bandwidth for each size in this grouping.

The Ls-series offers up to 32 vCPUs, using the [Intel® Xeon® processor E5 v3 family](#). The Ls-series gets the same CPU performance as the G/GS-Series and comes with 8 GiB of memory per vCPU.

Ls-series

ACU: 180-240

SIZE	VCPUs	MEMORY: GiB	TEMP STORAGE (SSD) GiB	MAX DATA DISKS	MAX TEMP STORAGE THROUGHPUT: IOPS / MBPS	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / EXPECTED NETWORK BANDWIDTH (MBPS)
Standard_L 4s	4	32	678	16	20,000 / 200	10,000 / 250	2 / 4,000
Standard_L 8s	8	64	1,388	32	40,000 / 400	20,000 / 500	4 / 8,000
Standard_L 16s	16	128	2,807	64	80,000 / 800	40,000 / 1,000	8 / 6,000 - 16,000 †
Standard_L 32s ¹	32	256	5,630	64	160,000 / 1,600	80,000 / 2,000	8 / 20,000

The maximum disk throughput possible with Ls-series VMs may be limited by the number, size, and striping of any attached disks. For details, see [Premium Storage: High-performance storage for Azure virtual machine workloads](#).

¹ Instance is isolated to hardware dedicated to a single customer.

Size table definitions

- Storage capacity is shown in units of GiB or 1024^3 bytes. When comparing disks measured in GB (1000^3 bytes) to disks measured in GiB (1024^3) remember that capacity numbers given in GiB may appear smaller. For example, 1023 GiB = 1098.4 GB
- Disk throughput is measured in input/output operations per second (IOPS) and MBps where MBps = 10^6 bytes/sec.
- Data disks can operate in cached or uncached modes. For cached data disk operation, the host cache mode is set to **ReadOnly** or **ReadWrite**. For uncached data disk operation, the host cache mode is set to **None**.
- If you want to get the best performance for your VMs, you should limit the number of data disks to 2 disks per vCPU.
- **Expected network bandwidth** is the maximum aggregated [bandwidth allocated per VM type](#) across all NICs, for all destinations. Upper limits are not guaranteed, but are intended to provide guidance for

selecting the right VM type for the intended application. Actual network performance will depend on a variety of factors including network congestion, application loads, and network settings. For information on optimizing network throughput, see [Optimizing network throughput for Windows and Linux](#). To achieve the expected network performance on Linux or Windows, it may be necessary to select a specific version or optimize your VM. For more information, see [How to reliably test for virtual machine throughput](#).

- † 16 vCPU performance will consistently reach the upper limit in an upcoming release.

Other sizes

- [General purpose](#)
- [Compute optimized](#)
- [Memory optimized](#)
- [GPU optimized](#)
- [High performance compute](#)

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

GPU optimized virtual machine sizes

12/15/2017 • 6 min to read • [Edit Online](#)

GPU optimized VM sizes are specialized virtual machines available with single or multiple NVIDIA GPUs. These sizes are designed for compute-intensive, graphics-intensive, and visualization workloads. This article provides information about the number and type of GPUs, vCPUs, data disks, and NICs as well as storage throughput and network bandwidth for each size in this grouping.

- **NC, NCv2, and ND** sizes are optimized for compute-intensive and network-intensive applications and algorithms, including CUDA- and OpenCL-based applications and simulations, AI, and Deep Learning.
- **NV** sizes are optimized and designed for remote visualization, streaming, gaming, encoding, and VDI scenarios utilizing frameworks such as OpenGL and DirectX.

NC instances

The NC instances are powered by the [NVIDIA Tesla K80](#) card. Users can crunch through data faster by leveraging CUDA for energy exploration applications, crash simulations, ray traced rendering, deep learning and more. The NC24r configuration provides a low latency, high-throughput network interface optimized for tightly coupled parallel computing workloads.

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	GPU	MAXIMUM DATA DISKS
Standard_NC6	6	56	380	1	24
Standard_NC12	12	112	680	2	48
Standard_NC24	24	224	1440	4	64
Standard_NC24r *	24	224	1440	4	64

1 GPU = one-half K80 card.

*RDMA capable

NCv2 instances

NCv2 instances are the next generation of the NC-series machines, powered by [NVIDIA Tesla P100](#) GPUs. These GPUs can provide more than 2x the computational performance of the current NC-series. Customers can take advantage of these updated GPUs for traditional HPC workloads such as reservoir modeling, DNA sequencing, protein analysis, Monte Carlo simulations, and others. Like the NC-series, the NCv2-series offers a configuration with a low latency, high-throughput network interface optimized for tightly coupled parallel computing workloads.

IMPORTANT

For this size family, the vCPU (core) quota in your subscription is initially set to 0 in each region. [Request a vCPU quota increase](#) for this family in an [available region](#).

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	GPU	MAXIMUM DATA DISKS
Standard_NC6_v2	6	112	336	1	12
Standard_NC12_v2	12	224	672	2	24
Standard_NC24_v2	24	448	1344	4	32
Standard_NC24r_v2*	24	1448	1344	4	32

1 GPU = one P100 card.

*RDMA capable

ND instances

The ND-series virtual machines are a new addition to the GPU family designed for AI and Deep Learning workloads. They offer excellent performance for training and inference. ND instances are powered by [NVIDIA Tesla P40](#) GPUs. These instances provide excellent performance for single-precision floating point operations, for AI workloads utilizing Microsoft Cognitive Toolkit, TensorFlow, Caffe, and other frameworks. The ND-series also offers a much larger GPU memory size (24 GB), enabling to fit much larger neural net models. Like the NC-series, the ND-series offers a configuration with a secondary low-latency, high-throughput network through RDMA, and InfiniBand connectivity so you can run large-scale training jobs spanning many GPUs.

IMPORTANT

For this size family, the vCPU (core) quota per region in your subscription is initially set to 0. [Request a vCPU quota increase](#) for this family in an [available region](#).

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	GPU	MAXIMUM DATA DISKS
Standard_ND6	6	112	336	1	12
Standard_ND12	12	224	672	2	24
Standard_ND24	24	448	1344	4	32
Standard_ND24r_v2*	24	1448	1344	4	32

1 GPU = one P40 card.

*RDMA capable

NV instances

The NV instances are powered by [NVIDIA Tesla M60](#) GPUs and NVIDIA GRID technology for desktop accelerated applications and virtual desktops where customers are able to visualize their data or simulations. Users are able to visualize their graphics intensive workflows on the NV instances to get superior graphics capability and

additionally run single precision workloads such as encoding and rendering.

SIZE	vCPU	MEMORY: GiB	TEMP STORAGE (SSD) GiB	GPU	MAXIMUM DATA DISKS
Standard_NV6	6	56	380	1	24
Standard_NV12	12	112	680	2	48
Standard_NV24	24	224	1440	4	64

1 GPU = one-half M60 card.

Size table definitions

- Storage capacity is shown in units of GiB or 1024^3 bytes. When comparing disks measured in GB (1000^3 bytes) to disks measured in GiB (1024^3) remember that capacity numbers given in GiB may appear smaller. For example, 1023 GiB = 1098.4 GB
- Disk throughput is measured in input/output operations per second (IOPS) and MBps where MBps = 10^6 bytes/sec.
- Data disks can operate in cached or uncached modes. For cached data disk operation, the host cache mode is set to **ReadOnly** or **ReadWrite**. For uncached data disk operation, the host cache mode is set to **None**.
- If you want to get the best performance for your VMs, you should limit the number of data disks to 2 disks per vCPU.
- **Expected network bandwidth** is the maximum aggregated [bandwidth allocated per VM type](#) across all NICs, for all destinations. Upper limits are not guaranteed, but are intended to provide guidance for selecting the right VM type for the intended application. Actual network performance will depend on a variety of factors including network congestion, application loads, and network settings. For information on optimizing network throughput, see [Optimizing network throughput for Windows and Linux](#). To achieve the expected network performance on Linux or Windows, it may be necessary to select a specific version or optimize your VM. For more information, see [How to reliably test for virtual machine throughput](#).
- † 16 vCPU performance will consistently reach the upper limit in an upcoming release.

Supported operating systems and drivers

NC, NCv2, and ND instances - NVIDIA Tesla drivers

OS	DRIVER
Windows Server 2016	385.54 (.exe)
Windows Server 2012 R2	385.54 (.exe)

NOTE

Tesla driver download links are current at time of publication. For the latest drivers, visit the [NVIDIA](#) website.

NV instances - NVIDIA GRID drivers

OS	Driver
Windows Server 2016	385.41 (.exe)
Windows Server 2012 R2	385.41 (.exe)

NOTE

Microsoft redistributes NVIDIA GRID driver installers for NV VMs. Install only these GRID drivers on Azure NV VMs. These drivers include licensing for GRID Virtual GPU Software in Azure.

For driver installation and verification steps, see [N-series driver setup for Windows](#).

Deployment considerations

- For availability of N-series VMs, see [Products available by region](#).
- N-series VMs can only be deployed in the Resource Manager deployment model.
- When creating an N-series VM using the Azure portal, on the **Basics** blade, select a **VM disk type** of **HDD**. To choose an available N-series size, on the **Size** blade, click **View all**.
- NC and NV VMs do not support VM disks that are backed by Azure Premium storage.
- If you want to deploy more than a few N-series VMs, consider a pay-as-you-go subscription or other purchase options. If you're using an [Azure free account](#), you can use only a limited number of Azure compute cores.
- You might need to increase the cores quota (per region) in your Azure subscription, and increase the separate quota for NC, NCv2, ND, or NV cores. To request a quota increase, [open an online customer support request](#) at no charge. Default limits may vary depending on your subscription category.
- One VM image you can deploy on N-series VMs is the [Azure Data Science Virtual Machine](#). The Data Science Virtual Machine preinstalls and configures many popular data science and deep learning tools. It also preinstalls NVIDIA Tesla GPU drivers.

Other sizes

- [General purpose](#)
- [Compute optimized](#)
- [High performance compute](#)
- [Memory optimized](#)
- [Storage optimized](#)

Next steps

Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Set up GPU drivers for N-series VMs running Windows Server

12/15/2017 • 2 min to read • [Edit Online](#)

To take advantage of the GPU capabilities of Azure N-series VMs running Windows Server 2016 or Windows Server 2012 R2, install supported NVIDIA graphics drivers. This article provides driver setup steps after you deploy an N-series VM. Driver setup information is also available for [Linux VMs](#).

For basic specs, storage capacities, and disk details, see [GPU Windows VM sizes](#).

Supported operating systems and drivers

NC, NCv2, and ND instances - NVIDIA Tesla drivers

os	DRIVER
Windows Server 2016	385.54 (.exe)
Windows Server 2012 R2	385.54 (.exe)

NOTE

Tesla driver download links are current at time of publication. For the latest drivers, visit the [NVIDIA](#) website.

NV instances - NVIDIA GRID drivers

os	DRIVER
Windows Server 2016	385.41 (.exe)
Windows Server 2012 R2	385.41 (.exe)

NOTE

Microsoft redistributes NVIDIA GRID driver installers for NV VMs. Install only these GRID drivers on Azure NV VMs. These drivers include licensing for GRID Virtual GPU Software in Azure.

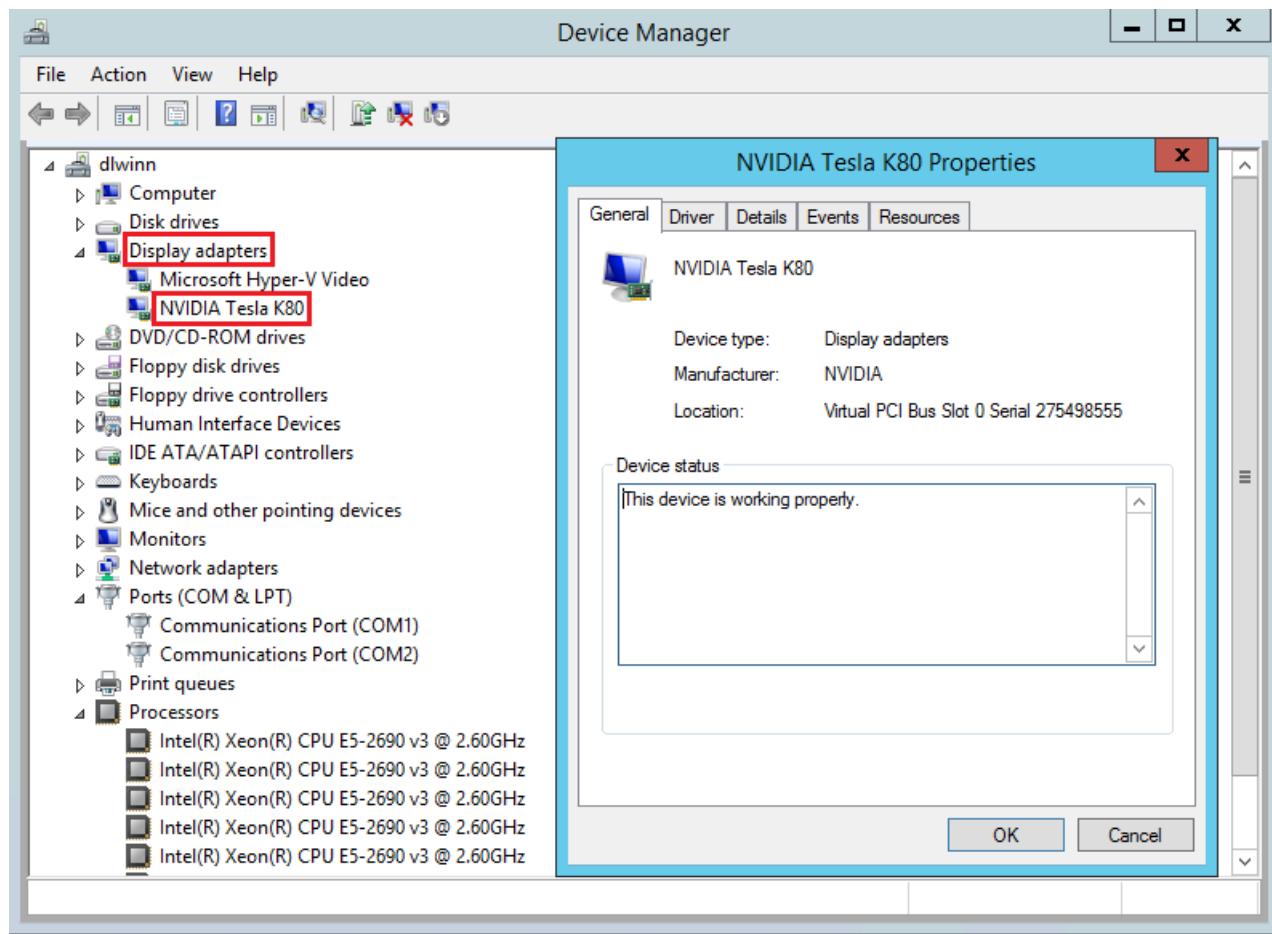
Driver installation

1. Connect by Remote Desktop to each N-series VM.
2. Download, extract, and install the supported driver for your Windows operating system.

On Azure NV VMs, a restart is required after driver installation. On NC VMs, a restart is not required.

Verify driver installation

You can verify driver installation in Device Manager. The following example shows successful configuration of the Tesla K80 card on an Azure NC VM.



To query the GPU device state, run the [nvidia-smi](#) command-line utility installed with the driver.

1. Open a command prompt and change to the **C:\Program Files\NVIDIA Corporation\NVSMI** directory.
2. Run `nvidia-smi`. If the driver is installed you will see output similar to the following. Note that **GPU-Util** shows **0%** unless you are currently running a GPU workload on the VM. Your driver version and GPU details may be different from the ones shown.

```
C:\Program Files\NVIDIA Corporation\NVSMI>nvidia-smi
Wed Nov 23 20:49:33 2016
+-----+-----+
| NVIDIA-SMI 369.73 | Driver Version: 369.73 |
+-----+-----+
| GPU  Name  TCC/WDDM | Bus-Id      Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap| Memory-Usage | GPU-Util  Compute M. |
|-----+-----+-----+-----+
|  0  Tesla K80      TCC | B794:00:00.0   Off |          0%          Default |
| N/A   56C    P8    28W / 149W |     0MiB / 11423MiB |          0%          Default |
+-----+-----+
+-----+-----+
| Processes:                               GPU Memory |
| GPU  PID  Type  Process name        Usage      |
|-----+-----+-----+-----|
| No running processes found            |
+-----+-----+
```

RDMA network connectivity

RDMA network connectivity can be enabled on RDMA enabled N-series VMs such as NC24r deployed in the same availability set. The HpcVmDrivers extension must be added to install Windows network device drivers that enable RDMA connectivity. To add the VM extension to an RDMA-enabled N-series VM, use [Azure PowerShell](#) cmdlets for Azure Resource Manager.

NOTE

Currently, only Windows Server 2012 R2 supports the RDMA network on N-series VMs.

To install the latest version 1.1 HpcVmDrivers extension on an existing RDMA-capable VM named myVM in the West US region:

```
Set-AzureRmVMEExtension -ResourceGroupName "myResourceGroup" -Location "westus" -VMName "myVM" -ExtensionName "HpcVmDrivers" -Publisher "Microsoft.HpcCompute" -Type "HpcVmDrivers" -TypeHandlerVersion "1.1"
```

For more information, see [Virtual machine extensions and features for Windows](#).

The RDMA network supports Message Passing Interface (MPI) traffic for applications running with [Microsoft MPI](#) or Intel MPI 5.x.

Next steps

- Developers building GPU-accelerated applications for the NVIDIA Tesla GPUs can also download and install the [CUDA Toolkit 9.1](#). For more information, see the [CUDA Installation Guide](#).

High performance compute VM sizes

11/16/2017 • 6 min to read • [Edit Online](#)

The A8-A11 and H-series sizes are also known as *compute-intensive instances*. The hardware that runs these sizes is designed and optimized for compute-intensive and network-intensive applications, including high-performance computing (HPC) cluster applications, modeling, and simulations. The A8-A11 series uses Intel Xeon E5-2670 @ 2.6 GHZ and the H-series uses Intel Xeon E5-2667 v3 @ 3.2 GHz. This article provides information about the number of vCPUs, data disks and NICs as well as storage throughput and network bandwidth for each size in this grouping.

Azure H-series virtual machines are the next generation high performance computing VMs aimed at high end computational needs, like molecular modeling, and computational fluid dynamics. These 8 and 16 vCPU VMs are built on the Intel Haswell E5-2667 V3 processor technology featuring DDR4 memory and SSD-based temporary storage.

In addition to the substantial CPU power, the H-series offers diverse options for low latency RDMA networking using FDR InfiniBand and several memory configurations to support memory intensive computational requirements.

H-series

ACU: 290-300

SIZE	VCPU	MEMORY: GIB	TEMP STORAGE (SSD) GIB	MAX DATA DISKS	MAX DISK THROUGHPUT: IOPS	MAX NICs
Standard_H8	8	56	1000	32	32 x 500	2
Standard_H16	16	112	2000	64	64 x 500	4
Standard_H8m	8	112	1000	32	32 x 500	2
Standard_H16m	16	224	2000	64	64 x 500	4
Standard_H16r ¹	16	112	2000	64	64 x 500	4
Standard_H16mr ¹	16	224	2000	64	64 x 500	4

¹ For MPI applications, dedicated RDMA backend network is enabled by FDR InfiniBand network, which delivers ultra-low-latency and high bandwidth.

A-series - compute-intensive instances

ACU: 225

SIZE	VCPUs	MEMORY: GiB	TEMP STORAGE (HDD): GiB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs
Standard_A8 ¹	8	56	382	32	32x500	2
Standard_A9 ¹	16	112	382	64	64x500	4
Standard_A10	8	56	382	32	32x500	2
Standard_A11	16	112	382	64	64x500	4

¹For MPI applications, dedicated RDMA backend network is enabled by FDR InfiniBand network, which delivers ultra-low-latency and high bandwidth.

Size table definitions

- Storage capacity is shown in units of GiB or 1024^3 bytes. When comparing disks measured in GB (1000^3 bytes) to disks measured in GiB (1024^3) remember that capacity numbers given in GiB may appear smaller. For example, 1023 GiB = 1098.4 GB
- Disk throughput is measured in input/output operations per second (IOPS) and MBps where MBps = 10^6 bytes/sec.
- Data disks can operate in cached or uncached modes. For cached data disk operation, the host cache mode is set to **ReadOnly** or **ReadWrite**. For uncached data disk operation, the host cache mode is set to **None**.
- If you want to get the best performance for your VMs, you should limit the number of data disks to 2 disks per vCPU.
- Expected network bandwidth** is the maximum aggregated [bandwidth allocated per VM type](#) across all NICs, for all destinations. Upper limits are not guaranteed, but are intended to provide guidance for selecting the right VM type for the intended application. Actual network performance will depend on a variety of factors including network congestion, application loads, and network settings. For information on optimizing network throughput, see [Optimizing network throughput for Windows and Linux](#). To achieve the expected network performance on Linux or Windows, it may be necessary to select a specific version or optimize your VM. For more information, see [How to reliably test for virtual machine throughput](#).
- + 16 vCPU performance will consistently reach the upper limit in an upcoming release.

Deployment considerations

- Azure subscription** – To deploy more than a few compute-intensive instances, consider a pay-as-you-go subscription or other purchase options. If you're using an [Azure free account](#), you can use only a limited number of Azure compute cores.
- Pricing and availability** - These VM sizes are offered only in the Standard pricing tier. Check [Products available by region](#) for availability in Azure regions.
- Cores quota** – You might need to increase the cores quota in your Azure subscription from the default value. Your subscription might also limit the number of cores you can deploy in certain VM size families, including the H-series. To request a quota increase, [open an online customer support request](#) at no charge. (Default limits may vary depending on your subscription category.)

NOTE

Contact Azure Support if you have large-scale capacity needs. Azure quotas are credit limits, not capacity guarantees. Regardless of your quota, you are only charged for cores that you use.

- **Virtual network** – An Azure [virtual network](#) is not required to use the compute-intensive instances. However, for many deployments you need at least a cloud-based Azure virtual network, or a site-to-site connection if you need to access on-premises resources. When needed, create a new virtual network to deploy the instances. Adding compute-intensive VMs to a virtual network in an affinity group is not supported.
- **Resizing** – Because of their specialized hardware, you can only resize compute-intensive instances within the same size family (H-series or compute-intensive A-series). For example, you can only resize an H-series VM from one H-series size to another. In addition, resizing from a non-compute-intensive size to a compute-intensive size is not supported.

RDMA-capable instances

A subset of the compute-intensive instances (H16r, H16mr, A8, and A9) feature a network interface for remote direct memory access (RDMA) connectivity. This interface is in addition to the standard Azure network interface available to other VM sizes.

This interface allows the RDMA-capable instances to communicate over an InfiniBand network, operating at FDR rates for H16r and H16mr virtual machines, and QDR rates for A8 and A9 virtual machines. These RDMA capabilities can boost the scalability and performance of Message Passing Interface (MPI) applications.

Following are requirements for RDMA-capable Windows VMs to access the Azure RDMA network:

- **Operating system**

Windows Server 2012 R2, Windows Server 2012

NOTE

Currently, Windows Server 2016 does not support RDMA connectivity in Azure.

- **Availability set or cloud service** – Deploy the RDMA-capable VMs in the same availability set (when you use the Azure Resource Manager deployment model) or the same cloud service (when you use the classic deployment model). If you use Azure Batch, the RDMA-capable VMs must be in the same pool.

- **MPI** - Microsoft MPI (MS-MPI) 2012 R2 or later, Intel MPI Library 5.x

Supported MPI implementations use the Microsoft Network Direct interface to communicate between instances.

- **RDMA network address space** - The RDMA network in Azure reserves the address space 172.16.0.0/16.

To run MPI applications on instances deployed in an Azure virtual network, make sure that the virtual network address space does not overlap the RDMA network.

- **HpcVmDrivers VM extension** - On RDMA-capable VMs, you must add the HpcVmDrivers extension to install Windows network device drivers for RDMA connectivity. (In certain deployments of A8 and A9 instances, the HpcVmDrivers extension is added automatically.) To add the VM extension to a VM, you can use [Azure PowerShell](#) cmdlets.

The following command installs the latest version 1.1 HpcVmDrivers extension on an existing RDMA-capable VM named *myVM* deployed in the resource group named *myResourceGroup* in the *West US* region:

```
Set-AzureRmVMExtension -ResourceGroupName "myResourceGroup" -Location "westus" -VMName "myVM" -ExtensionName "HpcVmDrivers" -Publisher "Microsoft.HpcCompute" -Type "HpcVmDrivers" -TypeHandlerVersion "1.1"
```

For more information, see [Virtual machine extensions and features](#). You can also work with extensions for VMs deployed in the [classic deployment model](#).

Using HPC Pack

[Microsoft HPC Pack](#), Microsoft's free HPC cluster and job management solution, is one option for you to create a compute cluster in Azure to run Windows-based MPI applications and other HPC workloads. HPC Pack 2012 R2 and later versions include a runtime environment for MS-MPI that uses the Azure RDMA network when deployed on RDMA-capable VMs.

Other sizes

- [General purpose](#)
- [Compute optimized](#)
- [Memory optimized](#)
- [Storage optimized](#)
- [GPU optimized](#)

Next steps

- For checklists to use the compute-intensive instances with HPC Pack on Windows Server, see [Set up a Windows RDMA cluster with HPC Pack to run MPI applications](#).
- To use compute-intensive instances when running MPI applications with Azure Batch, see [Use multi-instance tasks to run Message Passing Interface \(MPI\) applications in Azure Batch](#).
- Learn more about how [Azure compute units \(ACU\)](#) can help you compare compute performance across Azure SKUs.

Azure compute unit (ACU)

12/6/2017 • 1 min to read • [Edit Online](#)

We have created the concept of the Azure Compute Unit (ACU) to provide a way of comparing compute (CPU) performance across Azure SKUs. This will help you easily identify which SKU is most likely to satisfy your performance needs. ACU is currently standardized on a Small (Standard_A1) VM being 100 and all other SKUs then represent approximately how much faster that SKU can run a standard benchmark.

IMPORTANT

The ACU is only a guideline. The results for your workload may vary.

SKU FAMILY	ACU \ VCPU	VCPUs:Core
A0	50	1:1
A1-A4	100	1:1
A5-A7	100	1:1
A1_v2-A8_v2	100	1:1
A2m_v2-A8m_v2	100	1:1
A8-A11	225*	1:1
D1-D14	160	1:1
D1_v2-D15_v2	210 - 250*	1:1
DS1-DS14	160	1:1
DS1_v2-DS15_v2	210-250*	1:1
D_v3	160-190*	2:1**
Ds_v3	160-190*	2:1**
E_v3	160-190*	2:1**
Es_v3	160-190*	2:1**
F2s_v2-F72s_v2	195-210*	2:1**
F1-F16	210-250*	1:1
F1s-F16s	210-250*	1:1

SKU FAMILY	ACU \ VCPU	VCPU:CORE
------------	------------	-----------

G1-G5	180 - 240*	1:1
GS1-GS5	180 - 240*	1:1
H	290 - 300*	1:1
L4s-L32s	180 - 240*	1:1
M	160-180	2:1**

ACUs marked with a * use Intel® Turbo technology to increase CPU frequency and provide a performance boost. The amount of the boost can vary based on the VM size, workload, and other workloads running on the same host.

**Hyper-threaded.

Here are links to more information about the different sizes:

- [General-purpose](#)
- [Memory optimized](#)
- [Compute optimized](#)
- [GPU optimized](#)
- [High performance compute](#)
- [Storage optimized](#)

Planned maintenance for virtual machines in Azure

9/19/2017 • 4 min to read • [Edit Online](#)

Azure periodically performs updates to improve the reliability, performance, and security of the host infrastructure for virtual machines. These updates range from patching software components in the hosting environment (like operating system, hypervisor, and various agents deployed on the host), upgrading networking components, to hardware decommissioning. The majority of these updates are performed without any impact to the hosted virtual machines. However, there are cases where updates do have an impact:

- If the maintenance does not require a reboot, Azure uses in-place migration to pause the VM while the host is updated.
- If maintenance requires a reboot, you get a notice of when the maintenance is planned. In these cases, you'll also be given a time window where you can start the maintenance yourself, at a time that works for you.

This page describes how Microsoft Azure performs both types of maintenance. For more information about unplanned events (outages), see [Manage the availability of virtual machines for Windows](#) or [Linux](#).

Applications running in a virtual machine can gather information about upcoming updates by using the Azure Metadata Service for [Windows](#) or [Linux](#).

For "how-to" information on managing planned maintenance, see "Handling planned maintenance notifications" for [Linux](#) or [Windows](#).

In-place VM migration

When updates don't require a full reboot, an in-place live migration is used. During the update the virtual machine is paused for about 30 seconds, preserving the memory in RAM, while the hosting environment applies the necessary updates and patches. The virtual machine is then resumed and the clock of the virtual machine is automatically synchronized.

For VMs in availability sets, update domains are updated one at a time. All VMs in one update domain (UD) are paused, updated and then resumed before planned maintenance moves on to the next UD.

Some applications may be impacted by these types of updates. Applications that perform real-time event processing, like media streaming or transcoding, or high throughput networking scenarios, may not be designed to tolerate a 30 second pause.

Maintenance requiring a reboot

When VMs need to be rebooted for planned maintenance, you are notified in advance. Planned maintenance has two phases: the self-service window and a scheduled maintenance window.

The **self-service window** lets you initiate the maintenance on your VMs. During this time, you can query each VM to see their status and check the result of your last maintenance request.

When you start self-service maintenance, your VM is moved to a node that has already been updated and then powers it back on. Because the VM reboots, the temporary disk is lost and dynamic IP addresses associated with virtual network interface are updated.

If you start self-service maintenance and there is an error during the process, the operation is stopped, the VM is not updated and it is also removed from the planned maintenance iteration. You will be contacted in a later time with a new schedule and offered a new opportunity to do self-service maintenance.

When the self-service window has passed, the **scheduled maintenance window** begins. During this time window, you can still query for the maintenance window, but no longer be able to start the maintenance yourself.

Availability Considerations during Planned Maintenance

If you decide to wait until the planned maintenance window, there are a few things to consider for maintaining the highest availability of your VMs.

Paired Regions

Each Azure region is paired with another region within the same geography, together they make a regional pair. During planned maintenance, Azure will only update the VMs in a single region of a region pair. For example, when updating the Virtual Machines in North Central US, Azure will not update any Virtual Machines in South Central US at the same time. However, other regions such as North Europe can be under maintenance at the same time as East US. Understanding how region pairs work can help you better distribute your VMs across regions. For more information, see [Azure region pairs](#).

Availability sets and scale sets

When deploying a workload on Azure VMs, you can create the VMs within an availability set to provide high availability to your application. This ensures that during either an outage or maintenance events, at least one virtual machine is available.

Within an availability set, individual VMs are spread across up to 20 update domains (UDs). During planned maintenance, only a single update domain is impacted at any given time. Be aware that the order of update domains being impacted does not necessarily happen sequentially.

Virtual machine scale sets are an Azure compute resource that enables you to deploy and manage a set of identical VMs as a single resource. The scale set is automatically deployed across update domains, like VMs in an availability set. Just like with availability sets, with scale sets only a single update domain is impacted at any given time.

For more information about configuring your virtual machines for high availability, see Manage the availability of your virtual machines for [Windows](#) or [Linux](#).

Next steps

For information on managing planned maintenance, see [Handling planned maintenance notifications](#).

About disks storage for Azure Windows VMs

11/20/2017 • 8 min to read • [Edit Online](#)

Just like any other computer, virtual machines in Azure use disks as a place to store an operating system, applications, and data. All Azure virtual machines have at least two disks – a Windows operating system disk and a temporary disk. The operating system disk is created from an image, and both the operating system disk and the image are virtual hard disks (VHDs) stored in an Azure storage account. Virtual machines also can have one or more data disks, that are also stored as VHDs.

In this article, we will talk about the different uses for the disks, and then discuss the different types of disks you can create and use. This article is also available for [Linux virtual machines](#).

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager](#) and [classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

Disks used by VMs

Let's take a look at how the disks are used by the VMs.

Operating system disk

Every virtual machine has one attached operating system disk. It's registered as a SATA drive and labeled as the C: drive by default. This disk has a maximum capacity of 2048 gigabytes (GB).

Temporary disk

Each VM contains a temporary disk. The temporary disk provides short-term storage for applications and processes and is intended to only store data such as page or swap files. Data on the temporary disk may be lost during a [maintenance event](#) or when you [redeploy a VM](#). During a standard reboot of the VM, the data on the temporary drive should persist.

The temporary disk is labeled as the D: drive by default and it used for storing pagefile.sys. To remap this disk to a different drive letter, see [Change the drive letter of the Windows temporary disk](#). The size of the temporary disk varies, based on the size of the virtual machine. For more information, see [Sizes for Windows virtual machines](#).

For more information on how Azure uses the temporary disk, see [Understanding the temporary drive on Microsoft Azure Virtual Machines](#)

Data disk

A data disk is a VHD that's attached to a virtual machine to store application data, or other data you need to keep. Data disks are registered as SCSI drives and are labeled with a letter that you choose. Each data disk has a maximum capacity of 4095 GB. The size of the virtual machine determines how many data disks you can attach to it and the type of storage you can use to host the disks.

NOTE

For more information about virtual machines capacities, see [Sizes for Windows virtual machines](#).

Azure creates an operating system disk when you create a virtual machine from an image. If you use an image that includes data disks, Azure also creates the data disks when it creates the virtual machine. Otherwise, you add data

disks after you create the virtual machine.

You can add data disks to a virtual machine at any time, by **attaching** the disk to the virtual machine. You can use a VHD that you've uploaded or copied to your storage account, or one that Azure creates for you. Attaching a data disk associates the VHD file with the VM by placing a 'lease' on the VHD so it can't be deleted from storage while it's still attached.

About VHDs

The VHDs used in Azure are .vhd files stored as page blobs in a standard or premium storage account in Azure. For details about page blobs, see [Understanding block blobs and page blobs](#). For details about premium storage, see [High-performance premium storage and Azure VMs](#).

Azure supports the fixed disk VHD format. The fixed format lays the logical disk out linearly within the file, so that disk offset X is stored at blob offset X. A small footer at the end of the blob describes the properties of the VHD. Often, the fixed format wastes space because most disks have large unused ranges in them. However, Azure stores .vhd files in a sparse format, so you receive the benefits of both the fixed and dynamic disks at the same time. For more details, see [Getting started with virtual hard disks](#).

All .vhd files in Azure that you want to use as a source to create disks or images are read-only. When you create a disk or image, Azure makes copies of the .vhd files. These copies can be read-only or read-and-write, depending on how you use the VHD.

When you create a virtual machine from an image, Azure creates a disk for the virtual machine that is a copy of the source .vhd file. To protect against accidental deletion, Azure places a lease on any source .vhd file that's used to create an image, an operating system disk, or a data disk.

Before you can delete a source .vhd file, you'll need to remove the lease by deleting the disk or image. To delete a .vhd file that is being used by a virtual machine as an operating system disk, you can delete the virtual machine, the operating system disk, and the source .vhd file all at once by deleting the virtual machine and deleting all associated disks. However, deleting a .vhd file that's a source for a data disk requires several steps in a set order. First you detach the disk from the virtual machine, then delete the disk, and then delete the .vhd file.

WARNING

If you delete a source .vhd file from storage, or delete your storage account, Microsoft can't recover that data for you.

Types of disks

Azure Disks are designed for 99.999% availability. Azure Disks have consistently delivered enterprise-grade durability, with an industry-leading ZERO% Annualized Failure Rate.

There are two performance tiers for storage that you can choose from when creating your disks -- Standard Storage and Premium Storage. Also, there are two types of disks -- unmanaged and managed -- and they can reside in either performance tier.

Standard storage

Standard Storage is backed by HDDs, and delivers cost-effective storage while still being performant. Standard storage can be replicated locally in one datacenter, or be geo-redundant with primary and secondary data centers. For more information about storage replication, please see [Azure Storage replication](#).

For more information about using Standard Storage with VM disks, please see [Standard Storage and Disks](#).

Premium storage

Premium Storage is backed by SSDs, and delivers high-performance, low-latency disk support for VMs running

I/O-intensive workloads. You can use Premium Storage with DS, DSv2, GS, Ls, or FS series Azure VMs. For more information, please see [Premium Storage](#).

Unmanaged disks

Unmanaged disks are the traditional type of disks that have been used by VMs. With these, you create your own storage account and specify that storage account when you create the disk. You have to make sure you don't put too many disks in the same storage account, because you could exceed the [scalability targets](#) of the storage account (20,000 IOPS, for example), resulting in the VMs being throttled. With unmanaged disks, you have to figure out how to maximize the use of one or more storage accounts to get the best performance out of your VMs.

Managed disks

Managed Disks handles the storage account creation/management in the background for you, and ensures that you do not have to worry about the scalability limits of the storage account. You simply specify the disk size and the performance tier (Standard/Premium), and Azure creates and manages the disk for you. Even as you add disks or scale the VM up and down, you don't have to worry about the storage being used.

You can also manage your custom images in one storage account per Azure region, and use them to create hundreds of VMs in the same subscription. For more information about Managed Disks, please see the [Managed Disks Overview](#).

We recommend that you use Azure Managed Disks for new VMs, and that you convert your previous unmanaged disks to managed disks, to take advantage of the many features available in Managed Disks.

Disk comparison

The following table provides a comparison of Premium vs Standard for both unmanaged and managed disks to help you decide what to use.

	AZURE PREMIUM DISK	AZURE STANDARD DISK
Disk Type	Solid State Drives (SSD)	Hard Disk Drives (HDD)
Overview	SSD-based high-performance, low-latency disk support for VMs running IO-intensive workloads or hosting mission critical production environment	HDD-based cost effective disk support for Dev/Test VM scenarios
Scenario	Production and performance sensitive workloads	Dev/Test, non-critical, Infrequent access
Disk Size	P4: 32 GB (Managed Disks only) P6: 64 GB (Managed Disks only) P10: 128 GB P20: 512 GB P30: 1024 GB P40: 2048 GB P50: 4095 GB	Unmanaged Disks: 1 GB – 4 TB (4095 GB) Managed Disks: S4: 32 GB S6: 64 GB S10: 128 GB S20: 512 GB S30: 1024 GB S40: 2048 GB S50: 4095 GB
Max Throughput per Disk	250 MB/s	60 MB/s
Max IOPS per Disk	7500 IOPS	500 IOPS

One last recommendation: Use TRIM with unmanaged standard disks

If you use unmanaged standard disks (HDD), you should enable TRIM. TRIM discards unused blocks on the disk so you are only billed for storage that you are actually using. This can save on costs if you create large files and then delete them.

You can run this command to check the TRIM setting. Open a command prompt on your Windows VM and type:

```
fsutil behavior query DisableDeleteNotify
```

If the command returns 0, TRIM is enabled correctly. If it returns 1, run the following command to enable TRIM:

```
fsutil behavior set DisableDeleteNotify 0
```

NOTE

Note: Trim support starts with Windows Server 2012 / Windows 8 and above, see [New API allows apps to send "TRIM and Unmap" hints to storage media](#).

Next steps

- [Attach a disk](#) to add additional storage for your VM.
- [Create a snapshot](#).
- [Convert to managed disks](#).

Azure Managed Disks Overview

8/21/2017 • 8 min to read • [Edit Online](#)

Azure Managed Disks simplifies disk management for Azure IaaS VMs by managing the [storage accounts](#) associated with the VM disks. You only have to specify the type ([Premium](#) or [Standard](#)) and the size of disk you need, and Azure creates and manages the disk for you.

Benefits of managed disks

Let's take a look at some of the benefits you gain by using managed disks, starting with this Channel 9 video, [Better Azure VM Resiliency with Managed Disks](#).

Simple and scalable VM deployment

Managed Disks handles storage for you behind the scenes. Previously, you had to create storage accounts to hold the disks (VHD files) for your Azure VMs. When scaling up, you had to make sure you created additional storage accounts so you didn't exceed the IOPS limit for storage with any of your disks. With Managed Disks handling storage, you are no longer limited by the storage account limits (such as 20,000 IOPS / account). You also no longer have to copy your custom images (VHD files) to multiple storage accounts. You can manage them in a central location – one storage account per Azure region – and use them to create hundreds of VMs in a subscription.

Managed Disks will allow you to create up to 10,000 VM **disks** in a subscription, which will enable you to create thousands of **VMs** in a single subscription. This feature also further increases the scalability of [Virtual Machine Scale Sets \(VMSS\)](#) by allowing you to create up to a thousand VMs in a VMSS using a Marketplace image.

Better reliability for Availability Sets

Managed Disks provides better reliability for Availability Sets by ensuring that the disks of [VMs in an Availability Set](#) are sufficiently isolated from each other to avoid single points of failure. It does this by automatically placing the disks in different storage scale units (stamps). If a stamp fails due to hardware or software failure, only the VM instances with disks on those stamps fail. For example, let's say you have an application running on five VMs, and the VMs are in an Availability Set. The disks for those VMs won't all be stored in the same stamp, so if one stamp goes down, the other instances of the application continue to run.

Highly durable and available

Azure Disks are designed for 99.999% availability. Rest easier knowing that you have three replicas of your data that enables high durability. If one or even two replicas experience issues, the remaining replicas help ensure persistence of your data and high tolerance against failures. This architecture has helped Azure consistently deliver enterprise-grade durability for IaaS disks, with an industry-leading ZERO% Annualized Failure Rate.

Granular access control

You can use [Azure Role-Based Access Control \(RBAC\)](#) to assign specific permissions for a managed disk to one or more users. Managed Disks exposes a variety of operations, including read, write (create/update), delete, and retrieving a [shared access signature \(SAS\) URI](#) for the disk. You can grant access to only the operations a person

needs to perform his job. For example, if you don't want a person to copy a managed disk to a storage account, you can choose not to grant access to the export action for that managed disk. Similarly, if you don't want a person to use an SAS URI to copy a managed disk, you can choose not to grant that permission to the managed disk.

Azure Backup service support

Use Azure Backup service with Managed Disks to create a backup job with time-based backups, easy VM restoration and backup retention policies. Managed Disks only support Locally Redundant Storage (LRS) as the replication option; this means it keeps three copies of the data within a single region. For regional disaster recovery, you must backup your VM disks in a different region using [Azure Backup service](#) and a GRS storage account as backup vault. Currently Azure Backup supports data disk sizes up to 1TB for backup. Read more about this at [Using Azure Backup service for VMs with Managed Disks](#).

Pricing and Billing

When using Managed Disks, the following billing considerations apply:

- Storage Type
- Disk Size
- Number of transactions
- Outbound data transfers
- Managed Disk Snapshots (full disk copy)

Let's take a closer look at these.

Storage Type: Managed Disks offers 2 performance tiers: [Premium](#) (SSD-based) and [Standard](#) (HDD-based). The billing of a managed disk depends on which type of storage you have selected for the disk.

Disk Size: Billing for managed disks depends on the provisioned size of the disk. Azure maps the provisioned size (rounded up) to the nearest Managed Disks option as specified in the tables below. Each managed disk maps to one of the supported provisioned sizes and is billed accordingly. For example, if you create a standard managed disk and specify a provisioned size of 200 GB, you are billed as per the pricing of the S20 Disk type.

Here are the disk sizes available for a premium managed disk:

PREMIUM MANAGED DISK TYPE	P4	P6	P10	P20	P30	P40	P50
Disk Size	32 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)

Here are the disk sizes available for a standard managed disk:

STANDARD MANAGED DISK TYPE	S4	S6	S10	S20	S30	S40	S50
Disk Size	32 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)

Number of transactions: You are billed for the number of transactions that you perform on a standard managed disk. There is no cost for transactions for a premium managed disk.

Outbound data transfers: [Outbound data transfers](#) (data going out of Azure data centers) incur billing for

bandwidth usage.

For detailed information on pricing for Managed Disks, see [Managed Disks Pricing](#).

Managed Disk Snapshots

A Managed Snapshot is a read-only full copy of a managed disk which is stored as a standard managed disk by default. With snapshots, you can back up your managed disks at any point in time. These snapshots exist independent of the source disk and can be used to create new Managed Disks. They are billed based on the used size. For example, if you create a snapshot of a managed disk with provisioned capacity of 64 GB and actual used data size of 10 GB, snapshot will be billed only for the used data size of 10 GB.

[Incremental snapshots](#) are currently not supported for Managed Disks, but will be supported in the future.

To learn more about how to create snapshots with Managed Disks, please check out these resources:

- [Create copy of VHD stored as a Managed Disk using Snapshots in Windows](#)
- [Create copy of VHD stored as a Managed Disk using Snapshots in Linux](#)

Images

Managed Disks also support creating a managed custom image. You can create an image from your custom VHD in a storage account or directly from a generalized (sys-prepped) VM. This captures in a single image all managed disks associated with a VM, including both the OS and data disks. This enables creating hundreds of VMs using your custom image without the need to copy or manage any storage accounts.

For information on creating images, please check out the following articles:

- [How to capture a managed image of a generalized VM in Azure](#)
- [How to generalize and capture a Linux virtual machine using the Azure CLI 2.0](#)

Images versus snapshots

You often see the word "image" used with VMs, and now you see "snapshots" as well. It's important to understand the difference between these. With Managed Disks, you can take an image of a generalized VM that has been deallocated. This image will include all of the disks attached to the VM. You can use this image to create a new VM, and it will include all of the disks.

A snapshot is a copy of a disk at the point in time it is taken. It only applies to one disk. If you have a VM that only has one disk (the OS), you can take a snapshot or an image of it and create a VM from either the snapshot or the image.

What if a VM has five disks and they are striped? You could take a snapshot of each of the disks, but there is no awareness within the VM of the state of the disks – the snapshots only know about that one disk. In this case, the snapshots would need to be coordinated with each other, and that is not currently supported.

Managed Disks and Encryption

There are two kinds of encryption to discuss in reference to managed disks. The first one is Storage Service Encryption (SSE), which is performed by the storage service. The second one is Azure Disk Encryption, which you can enable on the OS and data disks for your VMs.

Storage Service Encryption (SSE)

[Azure Storage Service Encryption](#) provides encryption-at-rest and safeguard your data to meet your organizational security and compliance commitments. SSE is enabled by default for all Managed Disks, Snapshots and Images in all the regions where managed disks is available. Starting June 10th, 2017, all new managed

disks/snapshots/images and new data written to existing managed disks are automatically encrypted-at-rest with keys managed by Microsoft. Visit the [Managed Disks FAQ page](#) for more details.

Azure Disk Encryption (ADE)

Azure Disk Encryption allows you to encrypt the OS and Data disks used by an IaaS Virtual Machine. This includes managed disks. For Windows, the drives are encrypted using industry-standard BitLocker encryption technology. For Linux, the disks are encrypted using the DM-Crypt technology. This is integrated with Azure Key Vault to allow you to control and manage the disk encryption keys. For more information, please see [Azure Disk Encryption for Windows and Linux IaaS VMs](#).

Next steps

For more information about Managed Disks, please refer to the following articles.

Get started with Managed Disks

- [Create a VM using Resource Manager and PowerShell](#)
- [Create a Linux VM using the Azure CLI 2.0](#)
- [Attach a managed data disk to a Windows VM using PowerShell](#)
- [Add a managed disk to a Linux VM](#)
- [Managed Disks PowerShell Sample Scripts](#)
- [Use Managed Disks in Azure Resource Manager templates](#)

Compare Managed Disks storage options

- [Premium storage and disks](#)
- [Standard storage and disks](#)

Operational guidance

- [Migrate from AWS and other platforms to Managed Disks in Azure](#)
- [Convert Azure VMs to managed disks in Azure](#)

High-performance Premium Storage and managed disks for VMs

11/1/2017 • 21 min to read • [Edit Online](#)

Azure Premium Storage delivers high-performance, low-latency disk support for virtual machines (VMs) with input/output (I/O)-intensive workloads. VM disks that use Premium Storage store data on solid-state drives (SSDs). To take advantage of the speed and performance of premium storage disks, you can migrate existing VM disks to Premium Storage.

In Azure, you can attach several premium storage disks to a VM. Using multiple disks gives your applications up to 256 TB of storage per VM. With Premium Storage, your applications can achieve 80,000 I/O operations per second (IOPS) per VM, and a disk throughput of up to 2,000 megabytes per second (MB/s) per VM. Read operations give you very low latencies.

With Premium Storage, Azure offers the ability to truly lift-and-shift demanding enterprise applications like Dynamics AX, Dynamics CRM, Exchange Server, SAP Business Suite, and SharePoint farms to the cloud. You can run performance-intensive database workloads in applications like SQL Server, Oracle, MongoDB, MySQL, and Redis, which require consistent high performance and low latency.

NOTE

For the best performance for your application, we recommend that you migrate any VM disk that requires high IOPS to Premium Storage. If your disk does not require high IOPS, you can help limit costs by keeping it in standard Azure Storage. In standard storage, VM disk data is stored on hard disk drives (HDDs) instead of on SSDs.

Azure offers two ways to create premium storage disks for VMs:

- **Unmanaged disks**

The original method is to use unmanaged disks. In an unmanaged disk, you manage the storage accounts that you use to store the virtual hard disk (VHD) files that correspond to your VM disks. VHD files are stored as page blobs in Azure storage accounts.

- **Managed disks**

When you choose [Azure Managed Disks](#), Azure manages the storage accounts that you use for your VM disks. You specify the disk type (Premium or Standard) and the size of the disk that you need. Azure creates and manages the disk for you. You don't have to worry about placing the disks in multiple storage accounts to ensure that you stay within scalability limits for your storage accounts. Azure handles that for you.

We recommend that you choose managed disks, to take advantage of their many features.

To get started with Premium Storage, [create your free Azure account](#).

For information about migrating your existing VMs to Premium Storage, see [Convert a Windows VM from unmanaged disks to managed disks](#) or [Convert a Linux VM from unmanaged disks to managed disks](#).

NOTE

Premium Storage is available in most regions. For the list of available regions, see the row for **Disk Storage** in [Azure products available by region](#).

Features

Here are some of the features of Premium Storage:

- **Premium storage disks**

Premium Storage supports VM disks that can be attached to specific size-series VMs. Premium Storage supports DS-series, DSv2-series, GS-series, Ls-series, and Fs-series VMs. You have a choice of seven disk sizes: P4 (32GB), P6 (64GB), P10 (128GB), P20 (512GB), P30 (1024GB), P40 (2048GB), P50 (4095GB). P4 and P6 disk sizes are yet only supported for Managed Disks. Each disk size has its own performance specifications. Depending on your application requirements, you can attach one or more disks to your VM. We describe the specifications in more detail in [Premium Storage scalability and performance targets](#).

- **Premium page blobs**

Premium Storage supports page blobs. Use page blobs to store persistent, unmanaged disks for VMs in Premium Storage. Unlike standard Azure Storage, Premium Storage does not support block blobs, append blobs, files, tables, or queues. Premium page blobs supports six sizes from P10 to P50, and P60 (8191GiB). P60 Premium page blob is not supported to be attached as VM disks.

Any object placed in a premium storage account will be a page blob. The page blob snaps to one of the supported provisioned sizes. This is why a premium storage account is not intended to be used to store tiny blobs.

- **Premium storage account**

To start using Premium Storage, create a premium storage account for unmanaged disks. In the [Azure portal](#), to create a premium storage account, choose the **Premium** performance tier. Select the **Locally-redundant storage (LRS)** replication option. You also can create a premium storage account by setting the type to **Premium_LRS** in one of the following locations:

- [Storage REST API](#) (version 2014-02-14 or a later version)
- [Service Management REST API](#) (version 2014-10-01 or a later version; for Azure classic deployments)
- [Azure Storage Resource Provider REST API](#) (for Azure Resource Manager deployments)
- [Azure PowerShell](#) (version 0.8.10 or a later version)

To learn about premium storage account limits, see [Premium Storage scalability and performance targets](#).

- **Premium locally redundant storage**

A premium storage account supports only locally redundant storage as the replication option. Locally redundant storage keeps three copies of the data within a single region. For regional disaster recovery, you must back up your VM disks in a different region by using [Azure Backup](#). You also must use a geo-redundant storage (GRS) account as the backup vault.

Azure uses your storage account as a container for your unmanaged disks. When you create an Azure DS-series, DSv2-series, GS-series, or Fs-series VM with unmanaged disks, and you select a premium storage account, your operating system and data disks are stored in that storage account.

Supported VMs

Premium Storage supports DS-series, DSv2-series, GS-series, Ls-series, and Fs-series VMs. You can use standard and premium storage disks with these VM types. You cannot use premium storage disks with VM series that are not Premium Storage-compatible.

For information about VM types and sizes in Azure for Windows, see [Windows VM sizes](#). For information about VM types and sizes in Azure for Linux, see [Linux VM sizes](#).

These are some of the features of the DS-series, DSv2-series, GS-series, Ls-series, and Fs-series VMs:

- **Cloud service**

You can add DS-series VMs to a cloud service that has only DS-series VMs. Do not add DS-series VMs to an existing cloud service that has any type other than DS-series VMs. You can migrate your existing VHDs to a new cloud service that runs only DS-series VMs. If you want to use the same virtual IP address for the new cloud service that hosts your DS-series VMs, use [reserved IP addresses](#). GS-series VMs can be added to an existing cloud service that has only GS-series VMs.

- **Operating system disk**

You can set up your Premium Storage VM to use either a premium or a standard operating system disk. For the best experience, we recommend using a Premium Storage-based operating system disk.

- **Data disks**

You can use premium and standard disks in the same Premium Storage VM. With Premium Storage, you can provision a VM and attach several persistent data disks to the VM. If needed, to increase the capacity and performance of the volume, you can stripe across your disks.

NOTE

If you stripe premium storage data disks by using [Storage Spaces](#), set up Storage Spaces with 1 column for each disk that you use. Otherwise, overall performance of the striped volume might be lower than expected because of uneven distribution of traffic across the disks. By default, in Server Manager, you can set up columns for up to 8 disks. If you have more than 8 disks, use PowerShell to create the volume. Specify the number of columns manually. Otherwise, the Server Manager UI continues to use 8 columns, even if you have more disks. For example, if you have 32 disks in a single stripe set, specify 32 columns. To specify the number of columns the virtual disk uses, in the [New-VirtualDisk](#) PowerShell cmdlet, use the *NumberOfColumns* parameter. For more information, see [Storage Spaces Overview](#) and [Storage Spaces FAQs](#).

- **Cache**

VMs in the size series that support Premium Storage have a unique caching capability for high levels of throughput and latency. The caching capability exceeds underlying premium storage disk performance. You can set the disk caching policy on premium storage disks to **ReadOnly**, **ReadWrite**, or **None**. The default disk caching policy is **ReadOnly** for all premium data disks, and **ReadWrite** for operating system disks. For optimal performance for your application, use the correct cache setting. For example, for read-heavy or read-only data disks, such as SQL Server data files, set the disk caching policy to **ReadOnly**. For write-heavy or write-only data disks, such as SQL Server log files, set the disk caching policy to **None**. To learn more about optimizing your design with Premium Storage, see [Design for performance with Premium Storage](#).

- **Analytics**

To analyze VM performance by using disks in Premium Storage, turn on VM diagnostics in the [Azure portal](#). For more information, see [Azure VM monitoring with Azure Diagnostics Extension](#).

To see disk performance, use operating system-based tools like [Windows Performance Monitor](#) for Windows VMs and the `iostat` command for Linux VMs.

- **VM scale limits and performance**

Each Premium Storage-supported VM size has scale limits and performance specifications for IOPS, bandwidth, and the number of disks that can be attached per VM. When you use premium storage disks with VMs, make sure that there is sufficient IOPS and bandwidth on your VM to drive disk traffic.

For example, a STANDARD_DS1 VM has a dedicated bandwidth of 32 MB/s for premium storage disk traffic. A P10 premium storage disk can provide a bandwidth of 100 MB/s. If a P10 premium storage disk is attached to this VM, it can only go up to 32 MB/s. It cannot use the maximum 100 MB/s that the P10 disk can provide.

Currently, the largest VM in the DS-series is the Standard_DS15_v2. The Standard_DS15_v2 can provide up to 960 MB/s across all disks. The largest VM in the GS-series is the Standard_GS5. The Standard_GS5 can provide up to 2,000 MB/s across all disks.

Note that these limits are for disk traffic only. These limits don't include cache hits and network traffic. A separate bandwidth is available for VM network traffic. Bandwidth for network traffic is different from the dedicated bandwidth used by premium storage disks.

For the most up-to-date information about maximum IOPS and throughput (bandwidth) for Premium Storage-supported VMs, see [Windows VM sizes](#) or [Linux VM sizes](#).

For more information about premium storage disks and their IOPS and throughput limits, see the table in the next section.

Scalability and performance targets

In this section, we describe the scalability and performance targets to consider when you use Premium Storage.

Premium storage accounts have the following scalability targets:

TOTAL ACCOUNT CAPACITY	TOTAL BANDWIDTH FOR A LOCALLY REDUNDANT STORAGE ACCOUNT
Disk capacity: 35 TB Snapshot capacity: 10 TB	Up to 50 gigabits per second for inbound ¹ + outbound ²

¹ All data (requests) that are sent to a storage account

² All data (responses) that are received from a storage account

For more information, see [Azure Storage scalability and performance targets](#).

If you are using premium storage accounts for unmanaged disks and your application exceeds the scalability targets of a single storage account, you might want to migrate to managed disks. If you don't want to migrate to managed disks, build your application to use multiple storage accounts. Then, partition your data across those storage accounts. For example, if you want to attach 51-TB disks across multiple VMs, spread them across two storage accounts. 35 TB is the limit for a single premium storage account. Make sure that a single premium storage account never has more than 35 TB of provisioned disks.

Premium Storage disk limits

When you provision a premium storage disk, the size of the disk determines the maximum IOPS and throughput (bandwidth). Azure offers seven types of premium storage disks: P4(Managed Disks only), P6(Managed Disks only), P10, P20, P30, P40, and P50. Each premium storage disk type has specific limits for IOPS and throughput. Limits for the disk types are described in the following table:

Premium Disks Type	P4	P6	P10	P20	P30	P40	P50
Disk size	32 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)
IOPS per disk	120	240	500	2300	5000	7500	7500
Throughput per disk	25 MB per second	50 MB per second	100 MB per second	150 MB per second	200 MB per second	250 MB per second	250 MB per second

NOTE

Make sure sufficient bandwidth is available on your VM to drive disk traffic, as described in [Premium Storage-supported VMs](#). Otherwise, your disk throughput and IOPS is constrained to lower values. Maximum throughput and IOPS are based on the VM limits, not on the disk limits described in the preceding table.

Here are some important things to know about Premium Storage scalability and performance targets:

- **Provisioned capacity and performance**

When you provision a premium storage disk, unlike standard storage, you are guaranteed the capacity, IOPS, and throughput of that disk. For example, if you create a P50 disk, Azure provisions 4,095-GB storage capacity, 7,500 IOPS, and 250 MB/s throughput for that disk. Your application can use all or part of the capacity and performance.

- **Disk size**

Azure maps the disk size (rounded up) to the nearest premium storage disk option, as specified in the table in the preceding section. For example, a disk size of 100 GB is classified as a P10 option. It can perform up to 500 IOPS, with up to 100 MB/s throughput. Similarly, a disk of size 400 GB is classified as a P20. It can perform up to 2,300 IOPS, with 150 MB/s throughput.

NOTE

You can easily increase the size of existing disks. For example, you might want to increase the size of a 30-GB disk to 128 GB, or even to 1 TB. Or, you might want to convert your P20 disk to a P30 disk because you need more capacity or more IOPS and throughput.

- **I/O size**

The size of an I/O is 256 KB. If the data being transferred is less than 256 KB, it is considered 1 I/O unit. Larger I/O sizes are counted as multiple I/Os of size 256 KB. For example, 1,100 KB I/O is counted as 5 I/O units.

- **Throughput**

The throughput limit includes writes to the disk, and it includes disk read operations that aren't served from the cache. For example, a P10 disk has 100 MB/s throughput per disk. Some examples of valid throughput for a P10 disk are shown in the following table:

MAX THROUGHPUT PER P10 DISK	NON-CACHE READS FROM DISK	NON-CACHE WRITES TO DISK
100 MB/s	100 MB/s	0

MAX THROUGHPUT PER P10 DISK	NON-CACHE READS FROM DISK	NON-CACHE WRITES TO DISK
100 MB/s	0	100 MB/s
100 MB/s	60 MB/s	40 MB/s

- **Cache hits**

Cache hits are not limited by the allocated IOPS or throughput of the disk. For example, when you use a data disk with a **ReadOnly** cache setting on a VM that is supported by Premium Storage, reads that are served from the cache are not subject to the IOPS and throughput caps of the disk. If the workload of a disk is predominantly reads, you might get very high throughput. The cache is subject to separate IOPS and throughput limits at the VM level, based on the VM size. DS-series VMs have roughly 4,000 IOPS and 33 MB/s throughput per core for cache and local SSD I/Os. GS-series VMs have a limit of 5,000 IOPS and 50 MB/s throughput per core for cache and local SSD I/Os.

Throttling

Throttling might occur, if your application IOPS or throughput exceeds the allocated limits for a premium storage disk. Throttling also might occur if your total disk traffic across all disks on the VM exceeds the disk bandwidth limit available for the VM. To avoid throttling, we recommend that you limit the number of pending I/O requests for the disk. Use a limit based on scalability and performance targets for the disk you have provisioned, and on the disk bandwidth available to the VM.

Your application can achieve the lowest latency when it is designed to avoid throttling. However, if the number of pending I/O requests for the disk is too small, your application cannot take advantage of the maximum IOPS and throughput levels that are available to the disk.

The following examples demonstrate how to calculate throttling levels. All calculations are based on an I/O unit size of 256 KB.

Example 1

Your application has processed 495 I/O units of 16-KB size in one second on a P10 disk. The I/O units are counted as 495 IOPS. If you try a 2-MB I/O in the same second, the total of I/O units is equal to 495 + 8 IOPS. This is because $2 \text{ MB} / 256 \text{ KB} = 8 \text{ I/O units}$, when the I/O unit size is 256 KB. Because the sum of 495 + 8 exceeds the 500 IOPS limit for the disk, throttling occurs.

Example 2

Your application has processed 400 I/O units of 256-KB size on a P10 disk. The total bandwidth consumed is $(400 \times 256) / 1,024 \text{ KB} = 100 \text{ MB/s}$. A P10 disk has a throughput limit of 100 MB/s. If your application tries to perform more I/O operations in that second, it is throttled because it exceeds the allocated limit.

Example 3

You have a DS4 VM with two P30 disks attached. Each P30 disk is capable of 200 MB/s throughput. However, a DS4 VM has a total disk bandwidth capacity of 256 MB/s. You cannot drive both attached disks to the maximum throughput on this DS4 VM at the same time. To resolve this, you can sustain traffic of 200 MB/s on one disk and 56 MB/s on the other disk. If the sum of your disk traffic goes over 256 MB/s, disk traffic is throttled.

NOTE

If your disk traffic mostly consists of small I/O sizes, your application likely will hit the IOPS limit before the throughput limit. However, if the disk traffic mostly consists of large I/O sizes, your application likely will hit the throughput limit first, instead of the IOPS limit. You can maximize your application's IOPS and throughput capacity by using optimal I/O sizes. Also, you can limit the number of pending I/O requests for a disk.

To learn more about designing for high performance by using Premium Storage, see [Design for performance with Premium Storage](#).

Snapshots and Copy Blob

To the Storage service, the VHD file is a page blob. You can take snapshots of page blobs, and copy them to another location, such as to a different storage account.

Unmanaged disks

Create [incremental snapshots](#) for unmanaged premium disks the same way you use snapshots with standard storage. Premium Storage supports only locally redundant storage as the replication option. We recommend that you create snapshots, and then copy the snapshots to a geo-redundant standard storage account. For more information, see [Azure Storage redundancy options](#).

If a disk is attached to a VM, some API operations on the disk are not permitted. For example, you cannot perform a [Copy Blob](#) operation on that blob if the disk is attached to a VM. Instead, first create a snapshot of that blob by using the [Snapshot Blob](#) REST API. Then, perform the [Copy Blob](#) of the snapshot to copy the attached disk. Alternatively, you can detach the disk, and then perform any necessary operations.

The following limits apply to premium storage blob snapshots:

PREMIUM STORAGE LIMIT	VALUE
Maximum number of snapshots per blob	100
Storage account capacity for snapshots (Includes data in snapshots only. Does not include data in base blob.)	10 TB
Minimum time between consecutive snapshots	10 minutes

To maintain geo-redundant copies of your snapshots, you can copy snapshots from a premium storage account to a geo-redundant standard storage account by using AzCopy or Copy Blob. For more information, see [Transfer data with the AzCopy command-line utility](#) and [Copy Blob](#).

For detailed information about performing REST operations against page blobs in a premium storage account, see [Blob service operations with Azure Premium Storage](#).

Managed disks

A snapshot for a managed disk is a read-only copy of the managed disk. The snapshot is stored as a standard managed disk. Currently, [incremental snapshots](#) are not supported for managed disks. To learn how to take a snapshot for a managed disk, see [Create a copy of a VHD stored as an Azure managed disk by using managed snapshots in Windows](#) or [Create a copy of a VHD stored as an Azure managed disk by using managed snapshots in Linux](#).

If a managed disk is attached to a VM, some API operations on the disk are not permitted. For example, you cannot generate a shared access signature (SAS) to perform a copy operation while the disk is attached to a VM. Instead, first create a snapshot of the disk, and then perform the copy of the snapshot. Alternately, you can detach the disk and then generate an SAS to perform the copy operation.

Premium Storage for Linux VMs

You can use the following information to help you set up your Linux VMs in Premium Storage:

To achieve scalability targets in Premium Storage, for all premium storage disks with cache set to **ReadOnly** or **None**, you must disable "barriers" when you mount the file system. You don't need barriers in this scenario

because the writes to premium storage disks are durable for these cache settings. When the write request successfully finishes, data has been written to the persistent store. To disable "barriers," use one of the following methods. Choose the one for your file system:

- For **reiserFS**, to disable barriers, use the `barrier=none` mount option. (To enable barriers, use `barrier=flush`.)
- For **ext3/ext4**, to disable barriers, use the `barrier=0` mount option. (To enable barriers, use `barrier=1`.)
- For **XFS**, to disable barriers, use the `nobarrier` mount option. (To enable barriers, use `barrier`.)
- For premium storage disks with cache set to **ReadWrite**, enable barriers for write durability.
- For volume labels to persist after you restart the VM, you must update /etc/fstab with the universally unique identifier (UUID) references to the disks. For more information, see [Add a managed disk to a Linux VM](#).

The following Linux distributions have been validated for Azure Premium Storage. For better performance and stability with Premium Storage, we recommend that you upgrade your VMs to one of these versions, at a minimum (or to a later version). Some of the versions require the latest Linux Integration Services (LIS), v4.0, for Azure. To download and install a distribution, follow the link listed in the following table. We add images to the list as we complete validation. Note that our validations show that performance varies for each image.

Performance depends on workload characteristics and your image settings. Different images are tuned for different kinds of workloads.

DISTRIBUTION	VERSION	SUPPORTED KERNEL	DETAILS
Ubuntu	12.04	3.2.0-75.110+	Ubuntu-12_04_5-LTS- amd64-server-20150119-en-us-30GB
Ubuntu	14.04	3.13.0-44.73+	Ubuntu-14_04_1-LTS- amd64-server-20150123-en-us-30GB
Debian	7.x, 8.x	3.16.7-ckt4-1+	
SUSE	SLES 12	3.12.36-38.1+	suse-sles-12-priority-v20150213 suse-sles-12-v20150213
SUSE	SLES 11 SP4	3.0.101-0.63.1+	
CoreOS	584.0.0+	3.18.4+	CoreOS 584.0.0
CentOS	6.5, 6.6, 6.7, 7.0		LIS4 required <i>See note in the next section</i>
CentOS	7.1+	3.10.0-229.1.2.el7+	LIS4 recommended <i>See note in the next section</i>
Red Hat Enterprise Linux (RHEL)	6.8+, 7.2+		
Oracle	6.0+, 7.2+		UEK4 or RHCK
Oracle	7.0-7.1		UEK4 or RHCK w/ LIS 4.1+
Oracle	6.4-6.7		UEK4 or RHCK w/ LIS 4.1+

LIS drivers for OpenLogic CentOS

If you are running OpenLogic CentOS VMs, run the following command to install the latest drivers:

```
sudo rpm -e hypervpd ## (Might return an error if not installed. That's OK.)  
sudo yum install microsoft-hyper-v
```

To activate the new drivers, restart the computer.

Pricing and billing

When you use Premium Storage, the following billing considerations apply:

- **Premium storage disk and blob size**

Billing for a premium storage disk or blob depends on the provisioned size of the disk or blob. Azure maps the provisioned size (rounded up) to the nearest premium storage disk option. For details, see the table in [Premium Storage scalability and performance targets](#). Each disk maps to a supported provisioned disk size, and is billed accordingly. Billing for any provisioned disk is prorated hourly by using the monthly price for the Premium Storage offer. For example, if you provisioned a P10 disk and deleted it after 20 hours, you are billed for the P10 offering prorated to 20 hours. This is regardless of the amount of actual data written to the disk or the IOPS and throughput used.

- **Premium unmanaged disks snapshots**

Snapshots on premium unmanaged disks are billed for the additional capacity used by the snapshots. For more information about snapshots, see [Create a snapshot of a blob](#).

- **Premium managed disks snapshots**

A snapshot of a managed disk is a read-only copy of the disk. The disk is stored as a standard managed disk. A snapshot costs the same as a standard managed disk. For example, if you take a snapshot of a 128-GB premium managed disk, the cost of the snapshot is equivalent to a 128-GB standard managed disk.

- **Outbound data transfers**

[Outbound data transfers](#) (data going out of Azure datacenters) incur billing for bandwidth usage.

For detailed information about pricing for Premium Storage, Premium Storage-supported VMs, and managed disks, see these articles:

- [Azure Storage pricing](#)
- [Virtual Machines pricing](#)
- [Managed disks pricing](#)

Azure Backup support

For regional disaster recovery, you must back up your VM disks in a different region by using [Azure Backup](#) and a GRS storage account as a backup vault.

To create a backup job with time-based backups, easy VM restoration, and backup retention policies, use Azure Backup. You can use Backup both with unmanaged and managed disks. For more information, see [Azure Backup for VMs with unmanaged disks](#) and [Azure Backup for VMs with managed disks](#).

Next steps

For more information about Premium Storage, see the following articles.

Design and implement with Premium Storage

- Design for performance with Premium Storage
- Blob storage operations with Premium Storage

Operational guidance

- Migrate to Azure Premium Storage

Blog posts

- Azure Premium Storage generally available
- Announcing the GS-series: Adding Premium Storage support to the largest VMs in the public cloud

Azure Premium Storage: Design for High Performance

11/1/2017 • 41 min to read • [Edit Online](#)

Overview

This article provides guidelines for building high performance applications using Azure Premium Storage. You can use the instructions provided in this document combined with performance best practices applicable to technologies used by your application. To illustrate the guidelines, we have used SQL Server running on Premium Storage as an example throughout this document.

While we address performance scenarios for the Storage layer in this article, you will need to optimize the application layer. For example, if you are hosting a SharePoint Farm on Azure Premium Storage, you can use the SQL Server examples from this article to optimize the database server. Additionally, optimize the SharePoint Farm's Web server and Application server to get the most performance.

This article will help answer following common questions about optimizing application performance on Azure Premium Storage,

- How to measure your application performance?
- Why are you not seeing expected high performance?
- Which factors influence your application performance on Premium Storage?
- How do these factors influence performance of your application on Premium Storage?
- How can you optimize for IOPS, Bandwidth and Latency?

We have provided these guidelines specifically for Premium Storage because workloads running on Premium Storage are highly performance sensitive. We have provided examples where appropriate. You can also apply some of these guidelines to applications running on IaaS VMs with Standard Storage disks.

Before you begin, if you are new to Premium Storage, first read the [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#) and [Azure Storage Scalability and Performance Targets](#) articles.

Application Performance Indicators

We assess whether an application is performing well or not using performance indicators like, how fast an application is processing a user request, how much data an application is processing per request, how many requests an application processing in a specific period of time, how long a user has to wait to get a response after submitting their request. The technical terms for these performance indicators are, IOPS, Throughput or Bandwidth, and Latency.

In this section, we will discuss the common performance indicators in the context of Premium Storage. In the following section, Gathering Application Requirements, you will learn how to measure these performance indicators for your application. Later in Optimizing Application Performance, you will learn about the factors affecting these performance indicators and recommendations to optimize them.

IOPS

IOPS is number of requests that your application is sending to the storage disks in one second. An input/output operation could be read or write, sequential or random. OLTP applications like an online retail website need to process many concurrent user requests immediately. The user requests are insert and update intensive database

transactions, which the application must process quickly. Therefore, OLTP applications require very high IOPS. Such applications handle millions of small and random IO requests. If you have such an application, you must design the application infrastructure to optimize for IOPS. In the later section, *Optimizing Application Performance*, we discuss in detail all the factors that you must consider to get high IOPS.

When you attach a premium storage disk to your high scale VM, Azure provisions for you a guaranteed number of IOPS as per the disk specification. For example, a P50 disk provisions 7500 IOPS. Each high scale VM size also has a specific IOPS limit that it can sustain. For example, a Standard GS5 VM has 80,000 IOPS limit.

Throughput

Throughput or Bandwidth is the amount of data that your application is sending to the storage disks in a specified interval. If your application is performing input/output operations with large IO unit sizes, it requires high Throughput. Data warehouse applications tend to issue scan intensive operations that access large portions of data at a time and commonly perform bulk operations. In other words, such applications require higher Throughput. If you have such an application, you must design its infrastructure to optimize for Throughput. In the next section, we discuss in detail the factors you must tune to achieve this.

When you attach a premium storage disk to a high scale VM, Azure provisions Throughput as per that disk specification. For example, a P50 disk provisions 250 MB per second disk Throughput. Each high scale VM size also has a specific Throughput limit that it can sustain. For example, Standard GS5 VM has a maximum throughput of 2,000 MB per second.

There is a relation between Throughput and IOPS as shown in the formula below.

$$\text{IOPS} \times \text{IO Size} = \text{Throughput}$$

Therefore, it is important to determine the optimal Throughput and IOPS values that your application requires. As you try to optimize one, the other also gets affected. In a later section, *Optimizing Application Performance*, we will discuss in more details about optimizing IOPS and Throughput.

Latency

Latency is the time it takes an application to receive a single request, send it to the storage disks and send the response to the client. This is a critical measure of an application's performance in addition to IOPS and Throughput. The Latency of a premium storage disk is the time it takes to retrieve the information for a request and communicate it back to your application. Premium Storage provides consistent low latencies. If you enable ReadOnly host caching on premium storage disks, you can get much lower read latency. We will discuss Disk Caching in more detail in later section on *Optimizing Application Performance*.

When you are optimizing your application to get higher IOPS and Throughput, it will affect the Latency of your application. After tuning the application performance, always evaluate the Latency of the application to avoid unexpected high latency behavior.

Gather Application Performance Requirements

The first step in designing high performance applications running on Azure Premium Storage is, to understand the performance requirements of your application. After you gather performance requirements, you can optimize your application to achieve the most optimal performance.

In the previous section, we explained the common performance indicators, IOPS, Throughput and Latency. You must identify which of these performance indicators are critical to your application to deliver the desired user experience. For example, high IOPS matters most to OLTP applications processing millions of transactions in a

second. Whereas, high Throughput is critical for Data Warehouse applications processing large amounts of data in a second. Extremely low Latency is crucial for real-time applications like live video streaming websites.

Next, measure the maximum performance requirements of your application throughout its lifetime. Use the sample checklist below as a start. Record the maximum performance requirements during normal, peak and off-hours workload periods. By identifying requirements for all workloads levels, you will be able to determine the overall performance requirement of your application. For example, the normal workload of an e-commerce website will be the transactions it serves during most days in a year. The peak workload of the website will be the transactions it serves during holiday season or special sale events. The peak workload is typically experienced for a limited period, but can require your application to scale two or more times its normal operation. Find out the 50 percentile, 90 percentile and 99 percentile requirements. This helps filter out any outliers in the performance requirements and you can focus your efforts on optimizing for the right values.

Application Performance Requirements Checklist

PERFORMANCE REQUIREMENTS	50 PERCENTILE	90 PERCENTILE	99 PERCENTILE
Max. Transactions per second			
% Read operations			
% Write operations			
% Random operations			
% Sequential operations			
IO request size			
Average Throughput			
Max. Throughput			
Min. Latency			
Average Latency			
Max. CPU			
Average CPU			
Max. Memory			
Average Memory			
Queue Depth			

NOTE

You should consider scaling these numbers based on expected future growth of your application. It is a good idea to plan for growth ahead of time, because it could be harder to change the infrastructure for improving performance later.

If you have an existing application and want to move to Premium Storage, first build the checklist above for the existing application. Then, build a prototype of your application on Premium Storage and design the application based on guidelines described in *Optimizing Application Performance* in a later section of this document. The next section describes the tools you can use to gather the performance measurements.

Create a checklist similar to your existing application for the prototype. Using Benchmarking tools you can simulate the workloads and measure performance on the prototype application. See the section on [Benchmarking](#) to learn more. By doing so you can determine whether Premium Storage can match or surpass your application performance requirements. Then you can implement the same guidelines for your production application.

Counters to measure application performance requirements

The best way to measure performance requirements of your application, is to use performance-monitoring tools provided by the operating system of the server. You can use PerfMon for Windows and iostat for Linux. These tools capture counters corresponding to each measure explained in the above section. You must capture the values of these counters when your application is running its normal, peak and off-hours workloads.

The PerfMon counters are available for processor, memory and, each logical disk and physical disk of your server. When you use premium storage disks with a VM, the physical disk counters are for each premium storage disk, and logical disk counters are for each volume created on the premium storage disks. You must capture the values for the disks that host your application workload. If there is a one to one mapping between logical and physical disks, you can refer to physical disk counters; otherwise refer to the logical disk counters. On Linux, the iostat command generates a CPU and disk utilization report. The disk utilization report provides statistics per physical device or partition. If you have a database server with its data and log on separate disks, collect this data for both disks. Below table describes counters for disks, processor and memory:

COUNTER	DESCRIPTION	PERFMON	IOSTAT
IOPS or Transactions per second	Number of I/O requests issued to the storage disk per second.	Disk Reads/sec Disk Writes/sec	tps r/s w/s
Disk Reads and Writes	% of Reads and Write operations performed on the disk.	% Disk Read Time % Disk Write Time	r/s w/s
Throughput	Amount of data read from or written to the disk per second.	Disk Read Bytes/sec Disk Write Bytes/sec	kB_read/s kB_wrtn/s
Latency	Total time to complete a disk IO request.	Average Disk sec/Read Average disk sec/Write	await svctm
IO size	The size of I/O requests issues to the storage disks.	Average Disk Bytes/Read Average Disk Bytes/Write	avgrq-sz
Queue Depth	Number of outstanding I/O requests waiting to be read form or written to the storage disk.	Current Disk Queue Length	avgqu-sz
Max. Memory	Amount of memory required to run application smoothly	% Committed Bytes in Use	Use vmstat
Max. CPU	Amount CPU required to run application smoothly	% Processor time	%util

Learn more about [iostat](#) and [PerfMon](#).

Optimizing Application Performance

The main factors that influence performance of an application running on Premium Storage are Nature of IO Requests, VM size, Disk size, Number of disks, Disk Caching, Multithreading and Queue Depth. You can control some of these factors with knobs provided by the system. Most applications may not give you an option to alter the IO size and Queue Depth directly. For example, if you are using SQL Server, you cannot choose the IO size and queue depth. SQL Server chooses the optimal IO size and queue depth values to get the most performance. It is important to understand the effects of both types of factors on your application performance, so that you can provision appropriate resources to meet performance needs.

Throughout this section, refer to the application requirements checklist that you created, to identify how much you need to optimize your application performance. Based on that, you will be able to determine which factors from this section you will need to tune. To witness the effects of each factor on your application performance, run benchmarking tools on your application setup. Refer to the [Benchmarking](#) section at the end of this article for steps to run common benchmarking tools on Windows and Linux VMs.

Optimizing IOPS, Throughput and Latency at a glance

The table below summarizes all the performance factors and the steps to optimize IOPS, Throughput and Latency. The sections following this summary will describe each factor in much more depth.

	IOPS	THROUGHPUT	LATENCY
Example Scenario	Enterprise OLTP application requiring very high transactions per second rate.	Enterprise Data warehousing application processing large amounts of data.	Near real-time applications requiring instant responses to user requests, like online gaming.
Performance factors			
IO size	Smaller IO size yields higher IOPS.	Larger IO size yields higher Throughput.	
VM size	Use a VM size that offers IOPS greater than your application requirement. See VM sizes and their IOPS limits here.	Use a VM size with Throughput limit greater than your application requirement. See VM sizes and their Throughput limits here.	Use a VM size that offers scale limits greater than your application requirement. See VM sizes and their limits here.
Disk size	Use a disk size that offers IOPS greater than your application requirement. See disk sizes and their IOPS limits here.	Use a disk size with Throughput limit greater than your application requirement. See disk sizes and their Throughput limits here.	Use a disk size that offers scale limits greater than your application requirement. See disk sizes and their limits here.
VM and Disk Scale Limits	IOPS limit of the VM size chosen should be greater than total IOPS driven by premium storage disks attached to it.	Throughput limit of the VM size chosen should be greater than total Throughput driven by premium storage disks attached to it.	Scale limits of the VM size chosen must be greater than total scale limits of attached premium storage disks.

	IOPS	THROUGHPUT	LATENCY
Disk Caching	Enable ReadOnly Cache on premium storage disks with Read heavy operations to get higher Read IOPS.		Enable ReadOnly Cache on premium storage disks with Ready heavy operations to get very low Read latencies.
Disk Striping	Use multiple disks and stripe them together to get a combined higher IOPS and Throughput limit. Note that the combined limit per VM should be higher than the combined limits of attached premium disks.		
Stripe Size	Smaller stripe size for random small IO pattern seen in OLTP applications. E.g., use stripe size of 64KB for SQL Server OLTP application.	Larger stripe size for sequential large IO pattern seen in Data Warehouse applications. E.g., use 256KB stripe size for SQL Server Data warehouse application.	
Multithreading	Use multithreading to push higher number of requests to Premium Storage that will lead to higher IOPS and Throughput. For example, on SQL Server set a high MAXDOP value to allocate more CPUs to SQL Server.		
Queue Depth	Larger Queue Depth yields higher IOPS.	Larger Queue Depth yields higher Throughput.	Smaller Queue Depth yields lower latencies.

Nature of IO Requests

An IO request is a unit of input/output operation that your application will be performing. Identifying the nature of IO requests, random or sequential, read or write, small or large, will help you determine the performance requirements of your application. It is very important to understand the nature of IO requests, to make the right decisions when designing your application infrastructure.

IO size is one of the more important factors. The IO size is the size of the input/output operation request generated by your application. The IO size has a significant impact on performance especially on the IOPS and Bandwidth that the application is able to achieve. The following formula shows the relationship between IOPS, IO size and Bandwidth/Throughput.

$$\text{IOPS} \times \text{IO Size} = \text{Throughput}$$

Some applications allow you to alter their IO size, while some applications do not. For example, SQL Server determines the optimal IO size itself, and does not provide users with any knobs to change it. On the other hand, Oracle provides a parameter called `DB_BLOCK_SIZE` using which you can configure the I/O request size of the database.

If you are using an application, which does not allow you to change the IO size, use the guidelines in this article to optimize the performance KPI that is most relevant to your application. For example,

- An OLTP application generates millions of small and random IO requests. To handle these type of IO requests, you must design your application infrastructure to get higher IOPS.
- A data warehousing application generates large and sequential IO requests. To handle these type of IO requests, you must design your application infrastructure to get higher Bandwidth or Throughput.

If you are using an application, which allows you to change the IO size, use this rule of thumb for the IO size in addition to other performance guidelines,

- Smaller IO size to get higher IOPS. For example, 8 KB for an OLTP application.
- Larger IO size to get higher Bandwidth/Throughput. For example, 1024 KB for a data warehouse application.

Here is an example on how you can calculate the IOPS and Throughput/Bandwidth for your application. Consider an application using a P30 disk. The maximum IOPS and Throughput/Bandwidth a P30 disk can achieve is 5000 IOPS and 200 MB per second respectively. Now, if your application requires the maximum IOPS from the P30 disk and you use a smaller IO size like 8 KB, the resulting Bandwidth you will be able to get is 40 MB per second. However, if your application requires the maximum Throughput/Bandwidth from P30 disk, and you use a larger IO size like 1024 KB, the resulting IOPS will be less, 200 IOPS. Therefore, tune the IO size such that it meets both your application's IOPS and Throughput/Bandwidth requirement. Table below summarizes the different IO sizes and their corresponding IOPS and Throughput for a P30 disk.

APPLICATION REQUIREMENT	I/O SIZE	IOPS	THROUGHPUT/BANDWIDTH
Max IOPS	8 KB	5,000	40 MB per second
Max Throughput	1024 KB	200	200 MB per second
Max Throughput + high IOPS	64 KB	3,200	200 MB per second
Max IOPS + high Throughput	32 KB	5,000	160 MB per second

To get IOPS and Bandwidth higher than the maximum value of a single premium storage disk, use multiple premium disks striped together. For example, stripe two P30 disks to get a combined IOPS of 10,000 IOPS or a combined Throughput of 400 MB per second. As explained in the next section, you must use a VM size that supports the combined disk IOPS and Throughput.

NOTE

As you increase either IOPS or Throughput the other also increases, make sure you do not hit throughput or IOPS limits of the disk or VM when increasing either one.

To witness the effects of IO size on application performance, you can run benchmarking tools on your VM and disks. Create multiple test runs and use different IO size for each run to see the impact. Refer to the [Benchmarking](#) section at the end of this article for more details.

High Scale VM Sizes

When you start designing an application, one of the first things to do is, choose a VM to host your application. Premium Storage comes with High Scale VM sizes that can run applications requiring higher compute power and a high local disk I/O performance. These VMs provide faster processors, a higher memory-to-core ratio, and a Solid-State Drive (SSD) for the local disk. Examples of High Scale VMs supporting Premium Storage are the DS, DSv2 and GS series VMs.

High Scale VMs are available in different sizes with a different number of CPU cores, memory, OS and temporary disk size. Each VM size also has maximum number of data disks that you can attach to the VM. Therefore, the chosen VM size will affect how much processing, memory, and storage capacity is available for your application. It also affects the Compute and Storage cost. For example, below are the specifications of the largest VM size in a DS series, DSv2 series and a GS series:

VM SIZE	CPU CORES	MEMORY	VM DISK SIZES	MAX. DATA DISKS	CACHE SIZE	IOPS	BANDWIDTH CACHE IO LIMITS
Standard_D S14	16	112 GB	OS = 1023 GB Local SSD = 224 GB	32	576 GB	50,000 IOPS 512 MB per second	4,000 IOPS and 33 MB per second
Standard_G S5	32	448 GB	OS = 1023 GB Local SSD = 896 GB	64	4224 GB	80,000 IOPS 2,000 MB per second	5,000 IOPS and 50 MB per second

To view a complete list of all available Azure VM sizes, refer to [Windows VM sizes](#) or [Linux VM sizes](#). Choose a VM size that can meet and scale to your desired application performance requirements. In addition to this, take into account following important considerations when choosing VM sizes.

Scale Limits

The maximum IOPS limits per VM and per disk are different and independent of each other. Make sure that the application is driving IOPS within the limits of the VM as well as the premium disks attached to it. Otherwise, application performance will experience throttling.

As an example, suppose an application requirement is a maximum of 4,000 IOPS. To achieve this, you provision a P30 disk on a DS1 VM. The P30 disk can deliver up to 5,000 IOPS. However, the DS1 VM is limited to 3,200 IOPS. Consequently, the application performance will be constrained by the VM limit at 3,200 IOPS and there will be degraded performance. To prevent this situation, choose a VM and disk size that will both meet application requirements.

Cost of Operation

In many cases, it is possible that your overall cost of operation using Premium Storage is lower than using Standard Storage.

For example, consider an application requiring 16,000 IOPS. To achieve this performance, you will need a Standard_D14 Azure IaaS VM, which can give a maximum IOPS of 16,000 using 32 standard storage 1TB disks. Each 1TB standard storage disk can achieve a maximum of 500 IOPS. The estimated cost of this VM per month will be \$1,570. The monthly cost of 32 standard storage disks will be \$1,638. The estimated total monthly cost will be \$3,208.

However, if you hosted the same application on Premium Storage, you will need a smaller VM size and fewer premium storage disks, thus reducing the overall cost. A Standard_DS13 VM can meet the 16,000 IOPS requirement using four P30 disks. The DS13 VM has a maximum IOPS of 25,600 and each P30 disk has a maximum IOPS of 5,000. Overall, this configuration can achieve $5,000 \times 4 = 20,000$ IOPS. The estimated cost of this VM per month will be \$1,003. The monthly cost of four P30 premium storage disks will be \$544.34. The estimated total monthly cost will be \$1,544.

Table below summarizes the cost breakdown of this scenario for Standard and Premium Storage.

	STANDARD	PREMIUM
Cost of VM per month	\$1,570.58 (Standard_D14)	\$1,003.66 (Standard_DS13)
Cost of Disks per month	\$1,638.40 (32 x 1 TB disks)	\$544.34 (4 x P30 disks)
Overall Cost per month	\$3,208.98	\$1,544.34

Linux Distros

With Azure Premium Storage, you get the same level of Performance for VMs running Windows and Linux. We support many flavors of Linux distros, and you can see the complete list [here](#). It is important to note that different distros are better suited for different types of workloads. You will see different levels of performance depending on the distro your workload is running on. Test the Linux distros with your application and choose the one that works best.

When running Linux with Premium Storage, check the latest updates about required drivers to ensure high performance.

Premium Storage Disk Sizes

Azure Premium Storage offers seven disk sizes currently. Each disk size has a different scale limit for IOPS, Bandwidth and Storage. Choose the right Premium Storage Disk size depending on the application requirements and the high scale VM size. The table below shows the seven disks sizes and their capabilities. P4 and P6 sizes are currently only supported for Managed Disks.

PREMIUM DISKS TYPE	P4	P6	P10	P20	P30	P40	P50
Disk size	32 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)
IOPS per disk	120	240	500	2300	5000	7500	7500
Throughput per disk	25 MB per second	50 MB per second	100 MB per second	150 MB per second	200 MB per second	250 MB per second	250 MB per second

How many disks you choose depends on the disk size chosen. You could use a single P50 disk or multiple P10 disks to meet your application requirement. Take into account considerations listed below when making the choice.

Scale Limits (IOPS and Throughput)

The IOPS and Throughput limits of each Premium disk size is different and independent from the VM scale limits. Make sure that the total IOPS and Throughput from the disks are within scale limits of the chosen VM size.

For example, if an application requirement is a maximum of 250 MB/sec Throughput and you are using a DS4 VM with a single P30 disk. The DS4 VM can give up to 256 MB/sec Throughput. However, a single P30 disk has Throughput limit of 200 MB/sec. Consequently, the application will be constrained at 200 MB/sec due to the disk limit. To overcome this limit, provision more than one data disks to the VM or resize your disks to P40 or P50.

NOTE

Reads served by the cache are not included in the disk IOPS and Throughput, hence not subject to disk limits. Cache has its separate IOPS and Throughput limit per VM.

For example, initially your reads and writes are 60MB/sec and 40MB/sec respectively. Over time, the cache warms up and serves more and more of the reads from the cache. Then, you can get higher write Throughput from the disk.

Number of Disks

Determine the number of disks you will need by assessing application requirements. Each VM size also has a limit on the number of disks that you can attach to the VM. Typically, this is twice the number of cores. Ensure that the VM size you choose can support the number of disks needed.

Remember, the Premium Storage disks have higher performance capabilities compared to Standard Storage disks. Therefore, if you are migrating your application from Azure IaaS VM using Standard Storage to Premium Storage, you will likely need fewer premium disks to achieve the same or higher performance for your application.

Disk Caching

High Scale VMs that leverage Azure Premium Storage have a multi-tier caching technology called BlobCache. BlobCache uses a combination of the Virtual Machine RAM and local SSD for caching. This cache is available for the Premium Storage persistent disks and the VM local disks. By default, this cache setting is set to Read/Write for OS disks and ReadOnly for data disks hosted on Premium Storage. With disk caching enabled on the Premium Storage disks, the high scale VMs can achieve extremely high levels of performance that exceed the underlying disk performance.

WARNING

Changing the cache setting of an Azure disk detaches and re-attaches the target disk. If it is the operating system disk, the VM is restarted. Stop all applications/services that might be affected by this disruption before changing the disk cache setting.

To learn more about how BlobCache works, refer to the [Inside Azure Premium Storage](#) blog post.

It is important to enable cache on the right set of disks. Whether you should enable disk caching on a premium disk or not will depend on the workload pattern that disk will be handling. Table below shows the default cache settings for OS and Data disks.

DISK TYPE	DEFAULT CACHE SETTING
OS disk	ReadWrite
Data disk	None

Following are the recommended disk cache settings for data disks,

DISK CACHING SETTING	RECOMMENDATION ON WHEN TO USE THIS SETTING
None	Configure host-cache as None for write-only and write-heavy disks.
ReadOnly	Configure host-cache as ReadOnly for read-only and read-write disks.

DISK CACHING SETTING	RECOMMENDATION ON WHEN TO USE THIS SETTING
ReadWrite	Configure host-cache as ReadWrite only if your application properly handles writing cached data to persistent disks when needed.

ReadOnly

By configuring ReadOnly caching on Premium Storage data disks, you can achieve low Read latency and get very high Read IOPS and Throughput for your application. This is due two reasons,

1. Reads performed from cache, which is on the VM memory and local SSD, are much faster than reads from the data disk, which is on the Azure blob storage.
2. Premium Storage does not count the Reads served from cache, towards the disk IOPS and Throughput.
Therefore, your application is able to achieve higher total IOPS and Throughput.

ReadWrite

By default, the OS disks have ReadWrite caching enabled. We have recently added support for ReadWrite caching on data disks as well. If you are using ReadWrite caching, you must have a proper way to write the data from cache to persistent disks. For example, SQL Server handles writing cached data to the persistent storage disks on its own. Using ReadWrite cache with an application that does not handle persisting the required data can lead to data loss, if the VM crashes.

As an example, you can apply these guidelines to SQL Server running on Premium Storage by doing the following,

1. Configure "ReadOnly" cache on premium storage disks hosting data files.
 - a. The fast reads from cache lower the SQL Server query time since data pages are retrieved much faster from the cache compared to directly from the data disks.
 - b. Serving reads from cache, means there is additional Throughput available from premium data disks. SQL Server can use this additional Throughput towards retrieving more data pages and other operations like backup/restore, batch loads, and index rebuilds.
2. Configure "None" cache on premium storage disks hosting the log files.
 - a. Log files have primarily write-heavy operations. Therefore, they do not benefit from the ReadOnly cache.

Disk Striping

When a high scale VM is attached with several premium storage persistent disks, the disks can be striped together to aggregate their IOPs, bandwidth, and storage capacity.

On Windows, you can use Storage Spaces to stripe disks together. You must configure one column for each disk in a pool. Otherwise, the overall performance of striped volume can be lower than expected, due to uneven distribution of traffic across the disks.

Important: Using Server Manager UI, you can set the total number of columns up to 8 for a striped volume. When attaching more than 8 disks, use PowerShell to create the volume. Using PowerShell, you can set the number of columns equal to the number of disks. For example, if there are 16 disks in a single stripe set; specify 16 columns in the *NumberOfColumns* parameter of the *New-VirtualDisk* PowerShell cmdlet.

On Linux, use the MDADM utility to stripe disks together. For detailed steps on striping disks on Linux refer to [Configure Software RAID on Linux](#).

Stripe Size

An important configuration in disk striping is the stripe size. The stripe size or block size is the smallest chunk of data that application can address on a striped volume. The stripe size you configure depends on the type of application and its request pattern. If you choose the wrong stripe size, it could lead to IO misalignment, which leads to degraded performance of your application.

For example, if an IO request generated by your application is bigger than the disk stripe size, the storage system writes it across stripe unit boundaries on more than one disk. When it is time to access that data, it will have to seek across more than one stripe units to complete the request. The cumulative effect of such behavior can lead to substantial performance degradation. On the other hand, if the IO request size is smaller than stripe size, and if it is random in nature, the IO requests may add up on the same disk causing a bottleneck and ultimately degrading the IO performance.

Depending on the type of workload your application is running, choose an appropriate stripe size. For random small IO requests, use a smaller stripe size. Whereas, for large sequential IO requests use a larger stripe size. Find out the stripe size recommendations for the application you will be running on Premium Storage. For SQL Server, configure stripe size of 64KB for OLTP workloads and 256KB for data warehousing workloads. See [Performance best practices for SQL Server on Azure VMs](#) to learn more.

NOTE

You can stripe together a maximum of 32 premium storage disks on a DS series VM and 64 premium storage disks on a GS series VM.

Multi-threading

Azure has designed Premium Storage platform to be massively parallel. Therefore, a multi-threaded application achieves much higher performance than a single-threaded application. A multi-threaded application splits up its tasks across multiple threads and increases efficiency of its execution by utilizing the VM and disk resources to the maximum.

For example, if your application is running on a single core VM using two threads, the CPU can switch between the two threads to achieve efficiency. While one thread is waiting on a disk IO to complete, the CPU can switch to the other thread. In this way, two threads can accomplish more than a single thread would. If the VM has more than one core, it further decreases running time since each core can execute tasks in parallel.

You may not be able to change the way an off-the-shelf application implements single threading or multi-threading. For example, SQL Server is capable of handling multi-CPU and multi-core. However, SQL Server decides under what conditions it will leverage one or more threads to process a query. It can run queries and build indexes using multi-threading. For a query that involves joining large tables and sorting data before returning to the user, SQL Server will likely use multiple threads. However, a user cannot control whether SQL Server executes a query using a single thread or multiple threads.

There are configuration settings that you can alter to influence this multi-threading or parallel processing of an application. For example, in case of SQL Server it is the maximum Degree of Parallelism configuration. This setting called MAXDOP, allows you to configure the maximum number of processors SQL Server can use when parallel processing. You can configure MAXDOP for individual queries or index operations. This is beneficial when you want to balance resources of your system for a performance critical application.

For example, say your application using SQL Server is executing a large query and an index operation at the same time. Let us assume that you wanted the index operation to be more performant compared to the large query. In such a case, you can set MAXDOP value of the index operation to be higher than the MAXDOP value for the query. This way, SQL Server has more number of processors that it can leverage for the index operation compared to the number of processors it can dedicate to the large query. Remember, you do not control the number of threads SQL Server will use for each operation. You can control the maximum number of processors being dedicated for multi-threading.

Learn more about [Degrees of Parallelism](#) in SQL Server. Find out such settings that influence multi-threading in your application and their configurations to optimize performance.

Queue Depth

The Queue Depth or Queue Length or Queue Size is the number of pending IO requests in the system. The value of Queue Depth determines how many IO operations your application can line up, which the storage disks will be processing. It affects all the three application performance indicators that we discussed in this article viz., IOPS, Throughput and Latency.

Queue Depth and multi-threading are closely related. The Queue Depth value indicates how much multi-threading can be achieved by the application. If the Queue Depth is large, application can execute more operations concurrently, in other words, more multi-threading. If the Queue Depth is small, even though application is multi-threaded, it will not have enough requests lined up for concurrent execution.

Typically, off the shelf applications do not allow you to change the queue depth, because if set incorrectly it will do more harm than good. Applications will set the right value of queue depth to get the optimal performance.

However, it is important to understand this concept so that you can troubleshoot performance issues with your application. You can also observe the effects of queue depth by running benchmarking tools on your system.

Some applications provide settings to influence the Queue Depth. For example, the MAXDOP (maximum degree of parallelism) setting in SQL Server explained in previous section. MAXDOP is a way to influence Queue Depth and multi-threading, although it does not directly change the Queue Depth value of SQL Server.

High Queue Depth

A high queue depth lines up more operations on the disk. The disk knows the next request in its queue ahead of time. Consequently, the disk can schedule operations ahead of time and process them in an optimal sequence. Since the application is sending more requests to the disk, the disk can process more parallel IOs. Ultimately, the application will be able to achieve higher IOPS. Since application is processing more requests, the total Throughput of the application also increases.

Typically, an application can achieve maximum Throughput with 8-16+ outstanding IOs per attached disk. If a Queue Depth is one, application is not pushing enough IOs to the system, and it will process less amount of in a given period. In other words, less Throughput.

For example, in SQL Server, setting the MAXDOP value for a query to "4" informs SQL Server that it can use up to four cores to execute the query. SQL Server will determine what is best queue depth value and the number of cores for the query execution.

Optimal Queue Depth

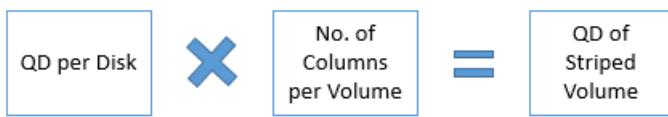
Very high queue depth value also has its drawbacks. If queue depth value is too high, the application will try to drive very high IOPS. Unless application has persistent disks with sufficient provisioned IOPS, this can negatively affect application latencies. Following formula shows the relationship between IOPS, Latency and Queue Depth.

$$\text{IOPS} \times \text{Latency} = \text{Queue Depth}$$

You should not configure Queue Depth to any high value, but to an optimal value, which can deliver enough IOPS for the application without affecting latencies. For example, if the application latency needs to be 1 millisecond, the Queue Depth required to achieve 5,000 IOPS is, $QD = 5000 \times 0.001 = 5$.

Queue Depth for Striped Volume

For a striped volume, maintain a high enough queue depth such that, every disk has a peak queue depth individually. For example, consider an application that pushes a queue depth of 2 and there are 4 disks in the stripe. The two IO requests will go to two disks and remaining two disks will be idle. Therefore, configure the queue depth such that all the disks can be busy. Formula below shows how to determine the queue depth of striped volumes.



Throttling

Azure Premium Storage provisions specified number of IOPS and Throughput depending on the VM sizes and disk sizes you choose. Anytime your application tries to drive IOPS or Throughput above these limits of what the VM or disk can handle, Premium Storage will throttle it. This manifests in the form of degraded performance in your application. This can mean higher latency, lower Throughput or lower IOPS. If Premium Storage does not throttle, your application could completely fail by exceeding what its resources are capable of achieving. So, to avoid performance issues due to throttling, always provision sufficient resources for your application. Take into consideration what we discussed in the VM sizes and Disk sizes sections above. Benchmarking is the best way to figure out what resources you will need to host your application.

Benchmarking

Benchmarking is the process of simulating different workloads on your application and measuring the application performance for each workload. Using the steps described in an earlier section, you have gathered the application performance requirements. By running benchmarking tools on the VMs hosting the application, you can determine the performance levels that your application can achieve with Premium Storage. In this section, we provide you examples of benchmarking a Standard DS14 VM provisioned with Azure Premium Storage disks.

We have used common benchmarking tools lometer and FIO, for Windows and Linux respectively. These tools spawn multiple threads simulating a production like workload, and measure the system performance. Using the tools you can also configure parameters like block size and queue depth, which you normally cannot change for an application. This gives you more flexibility to drive the maximum performance on a high scale VM provisioned with premium disks for different types of application workloads. To learn more about each benchmarking tool visit [lometer](#) and [FIO](#).

To follow the examples below, create a Standard DS14 VM and attach 11 Premium Storage disks to the VM. Of the 11 disks, configure 10 disks with host caching as "None" and stripe them into a volume called NoCacheWrites. Configure host caching as "ReadOnly" on the remaining disk and create a volume called CacheReads with this disk. Using this setup, you will be able to see the maximum Read and Write performance from a Standard DS14 VM. For detailed steps about creating a DS14 VM with premium disks, go to [Create and use a Premium Storage account for a virtual machine data disk](#).

Warming up the Cache

The disk with ReadOnly host caching will be able to give higher IOPS than the disk limit. To get this maximum read performance from the host cache, first you must warm up the cache of this disk. This ensures that the Read IOs which benchmarking tool will drive on CacheReads volume actually hits the cache and not the disk directly. The cache hits result in additional IOPS from the single cache enabled disk.

Important:

You must warm up the cache before running benchmarking, every time VM is rebooted.

lometer

[Download the lometer tool](#) on the VM.

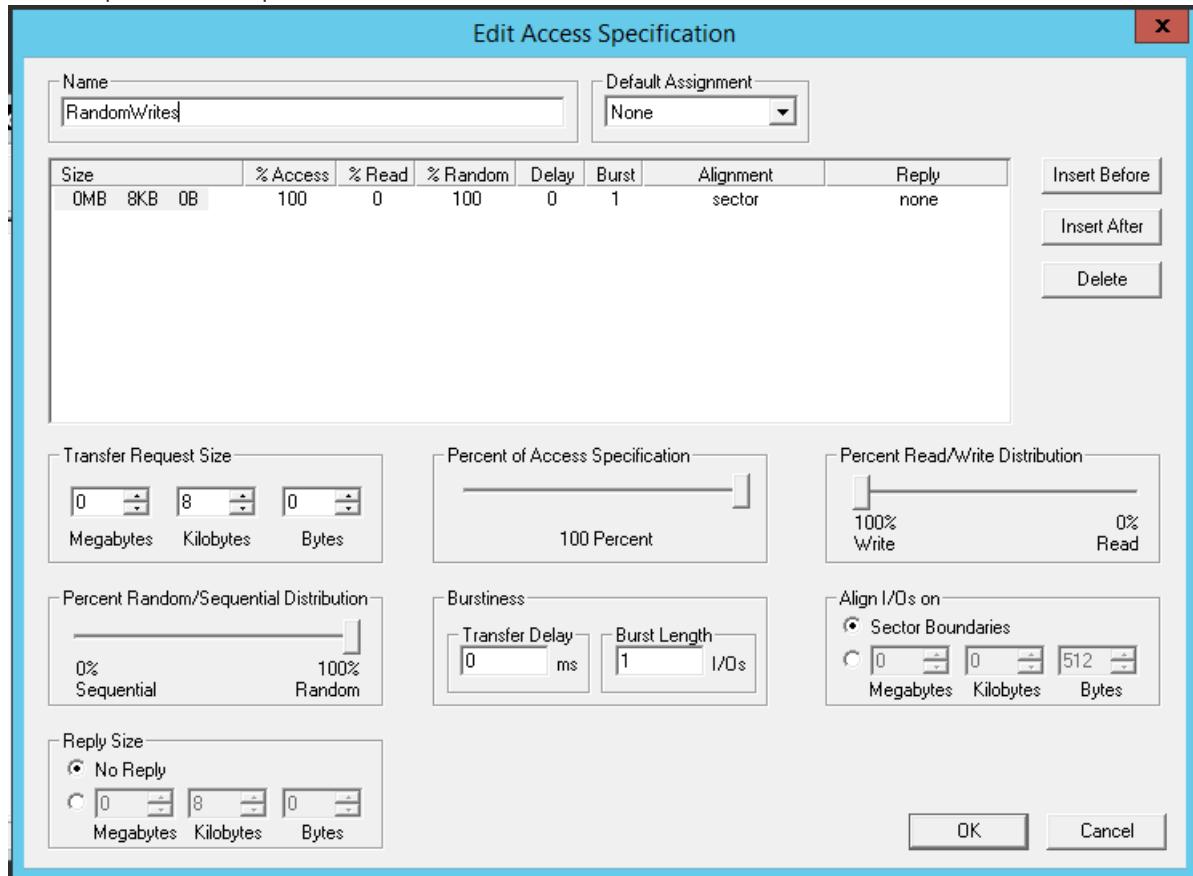
Test file

lometer uses a test file that is stored on the volume on which you will run the benchmarking test. It drives Reads and Writes on this test file to measure the disk IOPS and Throughput. lometer creates this test file if you have not provided one. Create a 200GB test file called iobw.tst on the CacheReads and NoCacheWrites volumes.

Access Specifications

The specifications, request IO size, % read/write, % random/sequential are configured using the "Access Specifications" tab in lometer. Create an access specification for each of the scenarios described below. Create the access specifications and "Save" with an appropriate name like – RandomWrites_8K, RandomReads_8K. Select the corresponding specification when running the test scenario.

An example of access specifications for maximum Write IOPS scenario is shown below,



Maximum IOPS Test Specifications

To demonstrate maximum IOPs, use smaller request size. Use 8K request size and create specifications for Random Writes and Reads.

ACCESS SPECIFICATION	REQUEST SIZE	RANDOM %	READ %
RandomWrites_8K	8K	100	0
RandomReads_8K	8K	100	100

Maximum Throughput Test Specifications

To demonstrate maximum Throughput, use larger request size. Use 64K request size and create specifications for Random Writes and Reads.

ACCESS SPECIFICATION	REQUEST SIZE	RANDOM %	READ %
RandomWrites_64K	64K	100	0
RandomReads_64K	64K	100	100

Running the lometer Test

Perform the steps below to warm up cache

1. Create two access specifications with values shown below,

NAME	REQUEST SIZE	RANDOM %	READ %
RandomWrites_1MB	1MB	100	0
RandomReads_1MB	1MB	100	100

2. Run the lometer test for initializing cache disk with following parameters. Use three worker threads for the target volume and a queue depth of 128. Set the "Run time" duration of the test to 2hrs on the "Test Setup" tab.

SCENARIO	TARGET VOLUME	NAME	DURATION
Initialize Cache Disk	CacheReads	RandomWrites_1MB	2hrs

3. Run the lometer test for warming up cache disk with following parameters. Use three worker threads for the target volume and a queue depth of 128. Set the "Run time" duration of the test to 2hrs on the "Test Setup" tab.

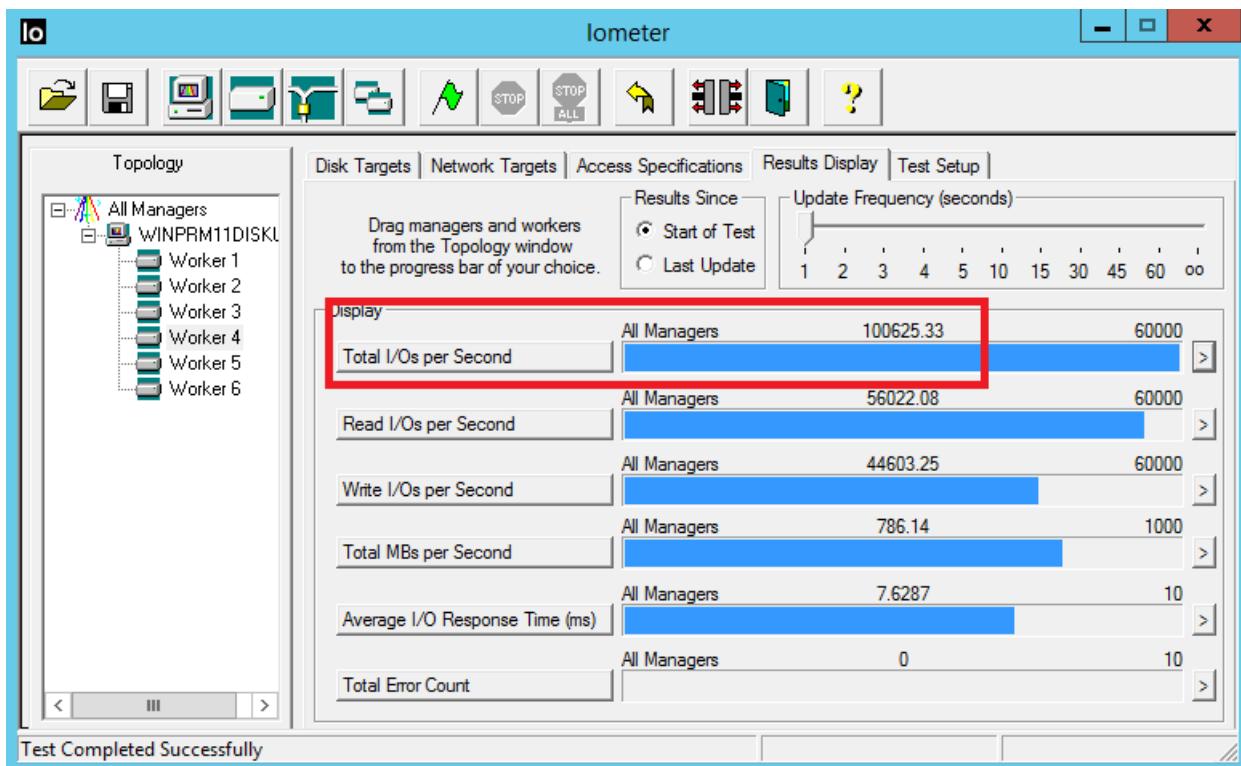
SCENARIO	TARGET VOLUME	NAME	DURATION
Warm up Cache Disk	CacheReads	RandomReads_1MB	2hrs

After cache disk is warmed up, proceed with the test scenarios listed below. To run the lometer test, use at least three worker threads for **each** target volume. For each worker thread, select the target volume, set queue depth and select one of the saved test specifications, as shown in the table below, to run the corresponding test scenario. The table also shows expected results for IOPS and Throughput when running these tests. For all scenarios, a small IO size of 8KB and a high queue depth of 128 is used.

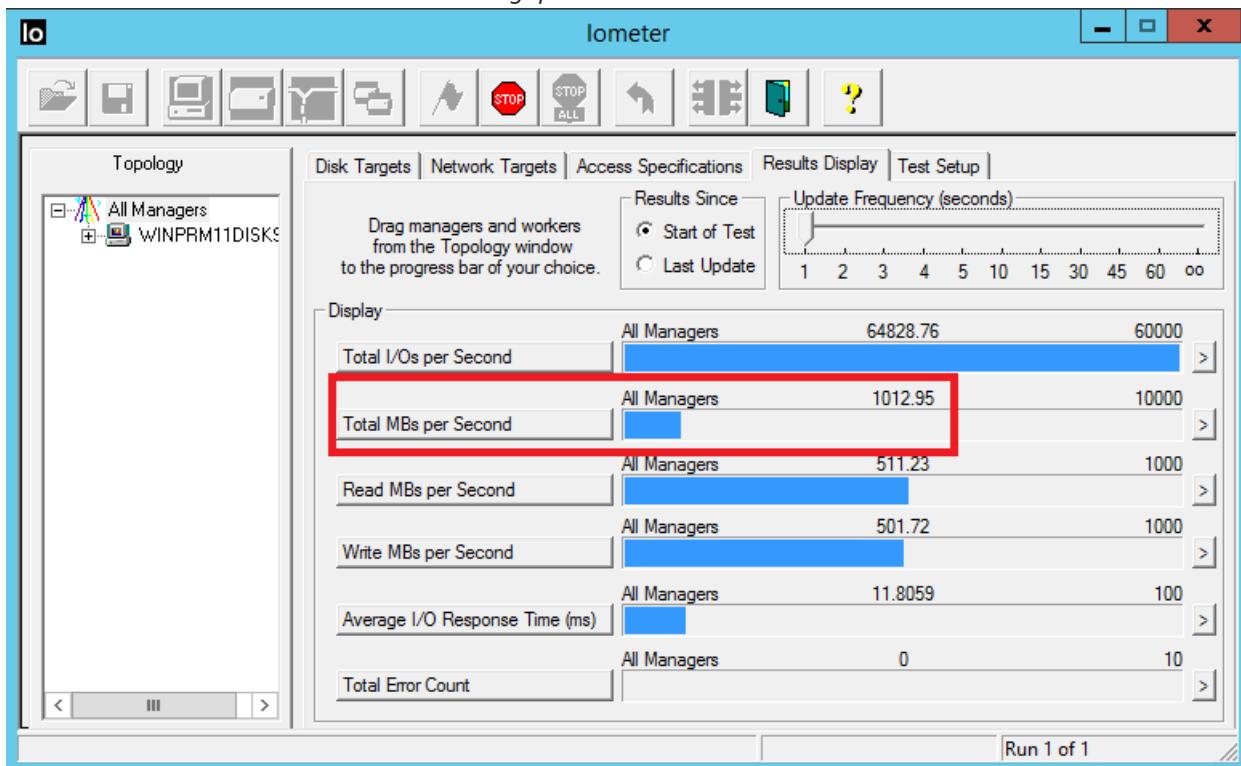
TEST SCENARIO	TARGET VOLUME	NAME	RESULT
Max. Read IOPS	CacheReads	RandomWrites_8K	50,000 IOPS
Max. Write IOPS	NoCacheWrites	RandomReads_8K	64,000 IOPS
Max. Combined IOPS	CacheReads	RandomWrites_8K	100,000 IOPS
NoCacheWrites	RandomReads_8K		
Max. Read MB/sec	CacheReads	RandomWrites_64K	524 MB/sec
Max. Write MB/sec	NoCacheWrites	RandomReads_64K	524 MB/sec
Combined MB/sec	CacheReads	RandomWrites_64K	1000 MB/sec
NoCacheWrites	RandomReads_64K		

Below are screenshots of the lometer test results for combined IOPS and Throughput scenarios.

Combined Reads and Writes Maximum IOPS



Combined Reads and Writes Maximum Throughput



FIO

FIO is a popular tool to benchmark storage on the Linux VMs. It has the flexibility to select different IO sizes, sequential or random reads and writes. It spawns worker threads or processes to perform the specified I/O operations. You can specify the type of I/O operations each worker thread must perform using job files. We created one job file per scenario illustrated in the examples below. You can change the specifications in these job files to benchmark different workloads running on Premium Storage. In the examples, we are using a Standard DS 14 VM running **Ubuntu**. Use the same setup described in the beginning of the [Benchmarking section](#) and warm up the cache before running the benchmarking tests.

Before you begin, [download FIO](#) and install it on your virtual machine.

Run the following command for Ubuntu,

```
apt-get install fio
```

We will use four worker threads for driving Write operations and four worker threads for driving Read operations on the disks. The Write workers will be driving traffic on the "nocache" volume, which has 10 disks with cache set to "None". The Read workers will be driving traffic on the "readcache" volume, which has 1 disk with cache set to "ReadOnly".

Maximum Write IOPS

Create the job file with following specifications to get maximum Write IOPS. Name it "fiowrite.ini".

```
[global]
size=30g
direct=1
iodepth=256
ioengine=libaio
bs=8k

[writer1]
rw=randwrite
directory=/mnt/nocache
[writer2]
rw=randwrite
directory=/mnt/nocache
[writer3]
rw=randwrite
directory=/mnt/nocache
[writer4]
rw=randwrite
directory=/mnt/nocache
```

Note the follow key things that are in line with the design guidelines discussed in previous sections. These specifications are essential to drive maximum IOPS,

- A high queue depth of 256.
- A small block size of 8KB.
- Multiple threads performing random writes.

Run the following command to kick off the FIO test for 30 seconds,

```
sudo fio --runtime 30 fiowrite.ini
```

While the test runs, you will be able to see the number of write IOPS the VM and Premium disks are delivering. As shown in the sample below, the DS14 VM is delivering its maximum write IOPS limit of 50,000 IOPS.

```
demo@DS-VM-Linux-Demo:~$ sudo fio --runtime 30 fiowrite.ini
[sudo] password for demo:
writer1: (g=0): rw=randwrite, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
writer2: (g=0): rw=randwrite, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
writer3: (g=0): rw=randwrite, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
writer4: (g=0): rw=randwrite, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
fio-2.1.11
Starting 4 processes
Jobs: 4 (f=4): [w(4)] [63.3% done] [0KB/396.4MB/0KB /s] [0/50.8K/0 iops] [eta 00m:11s]
```

Maximum Read IOPS

Create the job file with following specifications to get maximum Read IOPS. Name it "fioread.ini".

```

[global]
size=30g
direct=1
iodepth=256
ioengine=libaio
bs=8k

[reader1]
rw=randread
directory=/mnt/readcache
[reader2]
rw=randread
directory=/mnt/readcache
[reader3]
rw=randread
directory=/mnt/readcache
[reader4]
rw=randread
directory=/mnt/readcache

```

Note the follow key things that are in line with the design guidelines discussed in previous sections. These specifications are essential to drive maximum IOPS,

- A high queue depth of 256.
- A small block size of 8KB.
- Multiple threads performing random writes.

Run the following command to kick off the FIO test for 30 seconds,

```
sudo fio --runtime 30 fioread.ini
```

While the test runs, you will be able to see the number of read IOPS the VM and Premium disks are delivering. As shown in the sample below, the DS14 VM is delivering more than 64,000 Read IOPS. This is a combination of the disk and the cache performance.

```

demo@DS-VM-Linux-Demo:~$ sudo fio --runtime 30 fioread.ini
[sudo] password for demo:
reader1: (g=0): rw=randread, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
reader2: (g=0): rw=randread, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
reader3: (g=0): rw=randread, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
reader4: (g=0): rw=randread, bs=8K-8K/8K-8K/8K-8K, ioengine=libaio, iodepth=256
fio-2.1.11
Starting 4 processes
Jobs: 4 (f=4): [r(4)] [70.0% done] [514.8MB/0KB/0KB /s] [65.9K/0/0 iops] [eta 00m:09s]
```

Maximum Read and Write IOPS

Create the job file with following specifications to get maximum combined Read and Write IOPS. Name it "fioreadwrite.ini".

```

[global]
size=30g
direct=1
iodepth=128
ioengine=libaio
bs=4k

[reader1]
rw=randread
directory=/mnt/readcache
[reader2]
rw=randread
directory=/mnt/readcache
[reader3]
rw=randread
directory=/mnt/readcache
[reader4]
rw=randread
directory=/mnt/readcache

[writer1]
rw=randwrite
directory=/mnt/nocache
rate_iops=12500
[writer2]
rw=randwrite
directory=/mnt/nocache
rate_iops=12500
[writer3]
rw=randwrite
directory=/mnt/nocache
rate_iops=12500
[writer4]
rw=randwrite
directory=/mnt/nocache
rate_iops=12500

```

Note the follow key things that are in line with the design guidelines discussed in previous sections. These specifications are essential to drive maximum IOPS,

- A high queue depth of 128.
- A small block size of 4KB.
- Multiple threads performing random reads and writes.

Run the following command to kick off the FIO test for 30 seconds,

```
sudo fio --runtime 30 fioreadwrite.ini
```

While the test runs, you will be able to see the number of combined read and write IOPS the VM and Premium disks are delivering. As shown in the sample below, the DS14 VM is delivering more than 100,000 combined Read and Write IOPS. This is a combination of the disk and the cache performance.

```

demo@DS-VM-Linux-Demo:~$ sudo fio --runtime 30 fioreadwrite.ini
reader1: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
reader2: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
reader3: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
reader4: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
writer1: (g=0): rw=randwrite, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
writer2: (g=0): rw=randwrite, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
writer3: (g=0): rw=randwrite, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
writer4: (g=0): rw=randwrite, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio, iodepth=128
fio-2.1.11
Starting 8 processes
Jobs: 8 (f=8), CR=50000/0 IOPS: [r(4),w(4)] [22.6% done] [251.2MB/183.3MB/0KB /s] [64.3K/46.1K/0 iops] [eta 00m:24s]
```

Maximum Combined Throughput

To get the maximum combined Read and Write Throughput, use a larger block size and large queue depth with

multiple threads performing reads and writes. You can use a block size of 64KB and queue depth of 128.

Next Steps

Learn more about Azure Premium Storage:

- [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#)

For SQL Server users, read articles on Performance Best Practices for SQL Server:

- [Performance Best Practices for SQL Server in Azure Virtual Machines](#)
- [Azure Premium Storage provides highest performance for SQL Server in Azure VM](#)

Cost-effective Standard Storage and unmanaged and managed Azure VM disks

11/1/2017 • 7 min to read • [Edit Online](#)

Azure Standard Storage delivers reliable, low-cost disk support for VMs running latency-insensitive workloads. It also supports blobs, tables, queues, and files. With Standard Storage, the data is stored on hard disk drives (HDDs). When working with VMs, you can use standard storage disks for Dev/Test scenarios and less critical workloads, and premium storage disks for mission-critical production applications. Standard Storage is available in all Azure regions.

This article focuses on the use of standard storage for VM Disks. For more information about the use of storage with blobs, tables, queues, and files, please refer to the [Introduction to Storage](#).

Disk types

There are two ways to create standard disks for Azure VMs:

Unmanaged disks: This is the original method where you manage the storage accounts used to store the VHD files that correspond to the VM disks. VHD files are stored as page blobs in storage accounts. Unmanaged disks can be attached to any Azure VM size, including the VMs that primarily use Premium Storage, such as the DSv2 and GS series. Azure VMs support attaching several standard disks, allowing up to 256 TB of storage per VM.

Azure Managed Disks: This feature manages the storage accounts used for the VM disks for you. You specify the type (Premium or Standard) and size of disk you need, and Azure creates and manages the disk for you. You don't have to worry about placing the disks across multiple storage accounts in order to ensure you stay within the scalability limits for the storage accounts -- Azure handles that for you.

Even though both types of disks are available, we recommend using Managed Disks to take advantage of their many features.

To get started with Azure Standard Storage, visit [Get started for free](#).

For information on how to create a VM with Managed Disks, please see one of the following articles.

- [Create a VM using Resource Manager and PowerShell](#)
- [Create a Linux VM using the Azure CLI 2.0](#)

Standard Storage Features

Let's take a look at some of the features of Standard Storage. For more details, please see [Introduction to Azure Storage](#).

Standard Storage: Azure Standard Storage supports Azure Disks, Azure Blobs, Azure Files, Azure Tables, and Azure Queues. To use Standard Storage services, start with [Create an Azure Storage account](#).

Standard storage disks: Standard storage disks can be attached to all Azure VMs including size-series VMs used with Premium Storage such as the DSv2 and GS series. A standard storage disk can only be attached to one VM. However, you can attach one or more of these disks to a VM, up to the maximum disk count defined for that VM size. In the following section on Standard Storage Scalability and Performance Targets, we describe the specifications in more detail.

Standard page blob: Standard page blobs are used to hold persistent disks for VMs and can also be accessed

directly through REST like other types of Azure Blobs. [Page blobs](#) are a collection of 512-byte pages optimized for random read and write operations.

Storage Replication: In most regions, data in a standard storage account can be replicated locally or geo-replicated across multiple data centers. The four types of replication available are Locally-Redundant Storage (LRS), Zone-Redundant Storage (ZRS), Geo-Redundant Storage (GRS), and Read Access Geo-Redundant Storage (RA-GRS). Managed Disks in Standard Storage currently support Locally-Redundant Storage (LRS) only. For more information, please see [Storage Replication](#).

Scalability and Performance Targets

In this section, we describe the Scalability and Performance targets you need to consider when using standard storage.

Account limits – does not apply to managed disks

RESOURCE	DEFAULT LIMIT
TB per storage account	500 TB
Max ingress ¹ per storage account (US Regions)	10 Gbps if GRS/ZRS enabled, 20 Gbps for LRS
Max egress ¹ per storage account (US Regions)	20 Gbps if RA-GRS/GRS/ZRS enabled, 30 Gbps for LRS
Max ingress ¹ per storage account (European and Asian Regions)	5 Gbps if GRS/ZRS enabled, 10 Gbps for LRS
Max egress ¹ per storage account (European and Asian Regions)	10 Gbps if RA-GRS/GRS/ZRS enabled, 15 Gbps for LRS
Total Request Rate (assuming 1 KB object size) per storage account	Up to 20,000 IOPS, entities per second, or messages per second

¹ Ingress refers to all data (requests) being sent to a storage account. Egress refers to all data (responses) being received from a storage account.

For more information, see [Azure Storage Scalability and Performance Targets](#).

If the needs of your application exceed the scalability targets of a single storage account, build your application to use multiple storage accounts and partition your data across those storage accounts. Alternately, you can use Azure Managed Disks, and Azure manages the partitioning and placement of your data for you.

Standard Disks Limits

Unlike Premium Disks, the input/output operations per second (IOPS) and throughput (bandwidth) of Standard Disks are not provisioned. The performance of standard disks varies with the VM size to which the disk is attached, not to the size of the disk. You could expect to achieve up to the performance limit listed in the table below.

Standard disks limits (managed and unmanaged)

VM TIER	BASIC TIER VM	STANDARD TIER VM
Max Disk size	4095 GB	4095 GB
Max 8 KB IOPS per disk	Up to 300	Up to 500
Max Bandwidth per disk	Up to 60 MB/s	Up to 60 MB/s

If your workload requires high-performance, low-latency disk support, you should consider using Premium Storage. To know more benefits of Premium Storage, visit [High-Performance Premium Storage and Azure VM Disks](#).

Snapshots and copy blob

To the Storage service, the VHD file is a page blob. You can take snapshots of page blobs, and copy them to another location, such as a different storage account.

Unmanaged disks

You can create [incremental snapshots](#) for unmanaged standard disks in the same way you use snapshots with Standard Storage. We recommend that you create snapshots and then copy those snapshots to a geo-redundant standard storage account if your source disk is in a locally-redundant storage account. For more information, see [Azure Storage Redundancy Options](#).

If a disk is attached to a VM, certain API operations are not permitted on the disks. For example, you cannot perform a [Copy Blob](#) operation on that blob as long as the disk is attached to a VM. Instead, first create a snapshot of that blob by using the [Snapshot Blob](#) REST API method, and then perform the [Copy Blob](#) of the snapshot to copy the attached disk. Alternatively, you can detach the disk and then perform any necessary operations.

To maintain geo-redundant copies of your snapshots, you can copy snapshots from a locally-redundant storage account to a geo-redundant standard storage account by using AzCopy or Copy Blob. For more information, see [Transfer data with the AzCopy Command-Line Utility](#) and [Copy Blob](#).

For detailed information on performing REST operations against page blobs in standard storage accounts, see [Azure Storage Services REST API](#).

Managed disks

A snapshot for a managed disk is a read-only copy of the managed disk which is stored as a standard managed disk. Incremental Snapshots are not currently supported for Managed Disks but will be supported in the future.

If a managed disk is attached to a VM, certain API operations are not permitted on the disks. For example, you cannot generate a shared access signature (SAS) to perform a copy operation while the disk is attached to a VM. Instead, first create a snapshot of the disk, and then perform the copy of the snapshot. Alternately, you can detach the disk and then generate a shared access signature (SAS) to perform the copy operation.

Pricing and Billing

When using Standard Storage, the following billing considerations apply:

- Standard storage unmanaged disks/data size
- Standard managed disks
- Standard storage snapshots
- Outbound data transfers
- Transactions

Unmanaged storage data and disk size: For unmanaged disks and other data (blobs, tables, queues, and files), you are charged only for the amount of space you are using. For example, if you have a VM whose page blob is provisioned as 127 GB, but the VM is really only using 10 GB of space, you are billed for 10 GB of space. We support Standard storage up to 8191 GB, and Standard unmanaged disks up to 4095 GB.

Managed disks: Managed disks are billed on the provisioned size. If your disk is provisioned as a 10 GB disk and you are only using 5 GB, you are charged for the provision size of 10 GB.

Snapshots: Snapshots of standard disks are billed for the additional capacity used by the snapshots. For information on snapshots, see [Creating a Snapshot of a Blob](#).

Outbound data transfers: Outbound data transfers (data going out of Azure data centers) incur billing for bandwidth usage.

Transaction: Azure charges \$0.0036 per 100,000 transactions for standard storage. Transactions include both read and write operations to storage.

For detailed information on pricing for Standard Storage, Virtual Machines, and Managed Disks, see:

- [Azure Storage Pricing](#)
- [Virtual Machines Pricing](#)
- [Managed Disks Pricing](#)

Azure Backup service support

Virtual machines with unmanaged disks can be backed up using Azure Backup. [More details](#).

You can also use the Azure Backup service with Managed Disks to create a backup job with time-based backups, easy VM restoration and backup retention policies. You can read more about this at [Using Azure Backup service for VMs with Managed Disks](#).

Next steps

- [Introduction to Azure Storage](#)
- [Create a storage account](#)
- [Managed Disks Overview](#)
- [Create a VM using Resource Manager and PowerShell](#)
- [Create a Linux VM using the Azure CLI 2.0](#)

Scalability and performance targets for VM disks on Windows

11/16/2017 • 3 min to read • [Edit Online](#)

An Azure virtual machine supports attaching a number of data disks. This article describes scalability and performance targets for a VM's data disks. Use these targets to help decide the number and type of disk that you need to meet your performance and capacity requirements.

IMPORTANT

For optimal performance, limit the number of highly utilized disks attached to the virtual machine to avoid possible throttling. If all attached disks are not highly utilized at the same time, then the virtual machine can support a larger number of disks.

- **For Azure Managed Disks:** The disk limit for managed disks is per region and per disk type. The maximum limit, and also the default limit, is 10,000 managed disks per region and per disk type for a subscription. For example, you can create up to 10,000 standard managed disks and also 10,000 premium managed disks in a region, per subscription.

Managed snapshots and images count against the managed disks limit.

- **For standard storage accounts:** A standard storage account has a maximum total request rate of 20,000 IOPS. The total IOPS across all of your virtual machine disks in a standard storage account should not exceed this limit.

You can roughly calculate the number of highly utilized disks supported by a single standard storage account based on the request rate limit. For example, for a Basic Tier VM, the maximum number of highly utilized disks is about 66 (20,000/300 IOPS per disk), and for a Standard Tier VM, it is about 40 (20,000/500 IOPS per disk).

- **For premium storage accounts:** A premium storage account has a maximum total throughput rate of 50 Gbps. The total throughput across all of your VM disks should not exceed this limit.

See [Windows VM sizes](#) for additional details.

Managed virtual machine disks

Standard managed virtual machine disks

STANDARD DISK TYPE	S4	S6	S10	S20	S30	S40	S50
Disk size	32 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2TB)	4095 GB (4 TB)
IOPS per disk	500	500	500	500	500	500	500
Throughput per disk	60 MB/sec	60 MB/sec	60 MB/sec				

Premium managed virtual machine disks: per disk limits

PREMIUM DISKS TYPE	P4	P6	P10	P20	P30	P40	P50
Disk size	32 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)
IOPS per disk	120	240	500	2300	5000	7500	7500
Throughput per disk	25 MB/sec	50 MB/sec	100 MB/sec	150 MB/sec	200 MB/sec	250 MB/sec	250 MB/sec

Premium managed virtual machine disks: per VM limits

RESOURCE	DEFAULT LIMIT
Max IOPS Per VM	80,000 IOPS with GS5 VM
Max throughput per VM	2,000 MB/s with GS5 VM

Unmanaged virtual machine disks

Standard unmanaged virtual machine disks: per disk limits

VM TIER	BASIC TIER VM	STANDARD TIER VM
Disk size	4095 GB	4095 GB
Max 8 KB IOPS per persistent disk	300	500
Max number of disks performing max IOPS	66	40

Premium unmanaged virtual machine disks: per account limits

RESOURCE	DEFAULT LIMIT
Total disk capacity per account	35 TB
Total snapshot capacity per account	10 TB
Max bandwidth per account (ingress + egress ¹)	<=50 Gbps

¹Ingress refers to all data (requests) being sent to a storage account. Egress refers to all data (responses) being received from a storage account.

Premium unmanaged virtual machine disks: per disk limits

PREMIUM STORAGE DISK TYPE	P10	P20	P30	P40	P50
Disk size	128 GiB	512 GiB	1024 GiB (1 TB)	2048 GiB (2 TB)	4095 GiB (4 TB)

PREMIUM STORAGE DISK TYPE	P10	P20	P30	P40	P50
Max IOPS per disk	500	2300	5000	7500	7500
Max throughput per disk	100 MB/s	150 MB/s	200 MB/s	250 MB/s	250 MB/s
Max number of disks per storage account	280	70	35	17	8

Premium unmanaged virtual machine disks: per VM limits

RESOURCE	DEFAULT LIMIT
Max IOPS Per VM	80,000 IOPS with GS5 VM
Max throughput per VM	2,000 MB/s with GS5 VM

See also

[Azure subscription and service limits, quotas, and constraints](#)

Backup and disaster recovery for Azure IaaS disks

11/1/2017 • 21 min to read • [Edit Online](#)

This article explains how to plan for backup and disaster recovery (DR) of IaaS virtual machines (VMs) and disks in Azure. This document covers both managed and unmanaged disks.

First, we cover the built-in fault tolerance capabilities in the Azure platform that helps guard against local failures. We then discuss the disaster scenarios not fully covered by the built-in capabilities. This is the main topic addressed by this document. We also show several examples of workload scenarios where different backup and DR considerations can apply. We then review possible solutions for the DR of IaaS disks.

Introduction

The Azure platform uses various methods for redundancy and fault tolerance to help protect customers from localized hardware failures. Local failures can include problems with an Azure Storage server machine that stores part of the data for a virtual disk or failures of an SSD or HDD on that server. Such isolated hardware component failures can happen during normal operations.

The Azure platform is designed to be resilient to these failures. Major disasters can result in failures or the inaccessibility of many storage servers or even a whole datacenter. Although your VMs and disks are normally protected from localized failures, additional steps are necessary to protect your workload from region-wide catastrophic failures, such as a major disaster, that can affect your VM and disks.

In addition to the possibility of platform failures, problems with a customer application or data can occur. For example, a new version of your application might inadvertently make a change to the data that causes it to break. In that case, you might want to revert the application and the data to a prior version that contains the last known good state. This requires maintaining regular backups.

For regional disaster recovery, you must back up your IaaS VM disks to a different region.

Before we look at backup and DR options, let's recap a few methods available for handling localized failures.

Azure IaaS resiliency

Resiliency refers to the tolerance for normal failures that occur in hardware components. Resiliency is the ability to recover from failures and continue to function. It's not about avoiding failures, but responding to failures in a way that avoids downtime or data loss. The goal of resiliency is to return the application to a fully functioning state following a failure. Azure virtual machines and disks are designed to be resilient to common hardware faults. Let's look at how the Azure IaaS platform provides this resiliency.

A virtual machine consists mainly of two parts: a compute server and the persistent disks. Both affect the fault tolerance of a virtual machine.

If the Azure compute host server that houses your VM experiences a hardware failure, which is rare, Azure is designed to automatically restore the VM on another server. If this happens, your computer reboots, and the VM comes back up after some time. Azure automatically detects such hardware failures and executes recoveries to help ensure the customer VM is available as soon as possible.

Regarding IaaS disks, the durability of data is critical for a persistent storage platform. Azure customers have important business applications running on IaaS, and they depend on the persistence of the data. Azure designs protection for these IaaS disks, with three redundant copies of the data that is stored locally. These copies provide for high durability against local failures. If one of the hardware components that holds your disk fails, your VM is not affected, because there are two additional copies to support disk requests. It works fine, even if two different

hardware components that support a disk fail at the same time (which is very rare).

To ensure that you always maintain three replicas, Azure Storage automatically spawns a new copy of the data in the background if one of the three copies becomes unavailable. Therefore, it should not be necessary to use RAID with Azure disks for fault tolerance. A simple RAID 0 configuration should be sufficient for striping the disks, if necessary, to create larger volumes.

Because of this architecture, Azure has consistently delivered enterprise-grade durability for IaaS disks, with an industry-leading zero percent [annualized failure rate](#).

Localized hardware faults on the compute host or in the Storage platform can sometimes result in the temporary unavailability of the VM that is covered by the [Azure SLA](#) for VM availability. Azure also provides an industry-leading SLA for single VM instances that use Azure Premium Storage disks.

To safeguard application workloads from downtime due to the temporary unavailability of a disk or VM, customers can use [availability sets](#). Two or more virtual machines in an availability set provide redundancy for the application. Azure then creates these VMs and disks in separate fault domains with different power, network, and server components.

Because of these separate fault domains, localized hardware failures typically do not affect multiple VMs in the set at the same time. Having separate fault domains provides high availability for your application. It's considered a good practice to use availability sets when high availability is required. The next section covers the disaster recovery aspect.

Backup and disaster recovery

Disaster recovery is the ability to recover from rare, but major, incidents. This includes non-transient, wide-scale failures, such as service disruption that affects an entire region. Disaster recovery includes data backup and archiving, and might include manual intervention, such as restoring a database from a backup.

The Azure platform's built-in protection against localized failures might not fully protect the VMs/disks if a major disaster causes large-scale outages. This includes catastrophic events, such as if a datacenter is hit by a hurricane, earthquake, fire, or if there is a large-scale hardware unit failures. In addition, you might encounter failures due to application or data issues.

To help protect your IaaS workloads from outages, you should plan for redundancy and have backups to enable recovery. For disaster recovery, you should back up in a different geographic location away from the primary site. This helps ensure your backup is not affected by the same event that originally affected the VM or disks. For more information, see [Disaster recovery for Azure applications](#).

Your DR considerations might include the following aspects:

- High availability: The ability of the application to continue running in a healthy state, without significant downtime. By *healthy state*, we mean the application is responsive, and users can connect to the application and interact with it. Certain mission-critical applications and databases might be required to always be available, even when there are failures in the platform. For these workloads, you might need to plan redundancy for the application, as well as the data.
- Data durability: In some cases, the main consideration is ensuring that the data is preserved if a disaster happens. Therefore, you might need a backup of your data in a different site. For such workloads, you might not need full redundancy for the application, but only a regular backup of the disks.

Backup and DR scenarios

Let's look at a few typical examples of application workload scenarios and the considerations for planning for disaster recovery.

Scenario 1: Major database solutions

Consider a production database server, like SQL Server or Oracle, that can support high availability. Critical production applications and users depend on this database. The disaster recovery plan for this system might need to support the following requirements:

- The data must be protected and recoverable.
- The server must be available for use.

The disaster recovery plan might require maintaining a replica of the database in a different region as a backup. Depending on the requirements for server availability and data recovery, the solution might range from an active-active or active-passive replica site to periodic offline backups of the data. Relational databases, such as SQL Server and Oracle, provide various options for replication. For SQL Server, use [SQL Server AlwaysOn Availability Groups](#) for high availability.

NoSQL databases, like MongoDB, also support [replicas](#) for redundancy. The replicas for high availability are used.

Scenario 2: A cluster of redundant VMs

Consider a workload handled by a cluster of VMs that provide redundancy and load balancing. One example is a Cassandra cluster deployed in a region. This type of architecture already provides a high level of redundancy within that region. However, to protect the workload from a regional-level failure, you should consider spreading the cluster across two regions or making periodic backups to another region.

Scenario 3: IaaS application workload

Let's look at the IaaS application workload. For example, this might be a typical production workload running on an Azure VM. It might be a web server or file server holding the content and other resources of a site. It might also be a custom-built business application running on a VM that stored its data, resources, and application state on the VM disks. In this case, it's important to make sure you take backups on a regular basis. Backup frequency should be based on the nature of the VM workload. For example, if the application runs every day and modifies data, then the backup should be taken every hour.

Another example is a reporting server that pulls data from other sources and generates aggregated reports. The loss of this VM or disks might lead to the loss of the reports. However, it might be possible to rerun the reporting process and regenerate the output. In that case, you don't really have a loss of data, even if the reporting server is hit with a disaster. As a result, you might have a higher level of tolerance for losing part of the data on the reporting server. In that case, less frequent backups are an option to reduce costs.

Scenario 4: IaaS application data issues

IaaS application data issues are another possibility. Consider an application that computes, maintains, and serves critical commercial data, such as pricing information. A new version of your application had a software bug that incorrectly computed the pricing and corrupted the existing commerce data served by the platform. Here, the best course of action is to revert to the earlier version of the application and the data. To enable this, take periodic backups of your system.

Disaster recovery solution: Azure Backup

[Azure Backup](#) is used for backups and DR, and it works with [managed disks](#) as well as [unmanaged disks](#). You can create a backup job with time-based backups, easy VM restoration, and backup retention policies.

If you use [Premium Storage disks](#), [managed disks](#), or other disk types with the [locally redundant storage](#) option, it's especially important to make periodic DR backups. Azure Backup stores the data in your recovery services vault for long-term retention. Choose the [geo-redundant storage](#) option for the backup recovery services vault. That option ensures that backups are replicated to a different Azure region for safeguarding from regional disasters.

For [unmanaged disks](#), you can use the locally redundant storage type for IaaS disks, but ensure that Azure Backup is enabled with the geo-redundant storage option for the recovery services vault.

NOTE

If you use the [geo-redundant storage](#) or [read-access geo-redundant storage](#) option for your unmanaged disks, you still need consistent snapshots for backup and DR. Use either [Azure Backup](#) or [consistent snapshots](#).

The following table is a summary of the solutions available for DR.

SCENARIO	AUTOMATIC REPLICATION	DR SOLUTION
Premium Storage disks	Local (locally redundant storage)	Azure Backup
Managed disks	Local (locally redundant storage)	Azure Backup
Unmanaged locally redundant storage disks	Local (locally redundant storage)	Azure Backup
Unmanaged geo-redundant storage disks	Cross region (geo-redundant storage)	Azure Backup Consistent snapshots
Unmanaged read-access geo-redundant storage disks	Cross region (read-access geo-redundant storage)	Azure Backup Consistent snapshots

High availability is best met by using managed disks in an availability set along with Azure Backup. If you use unmanaged disks, you can still use Azure Backup for DR. If you are unable to use Azure Backup, then taking [consistent snapshots](#), as described in a later section, is an alternative solution for backup and DR.

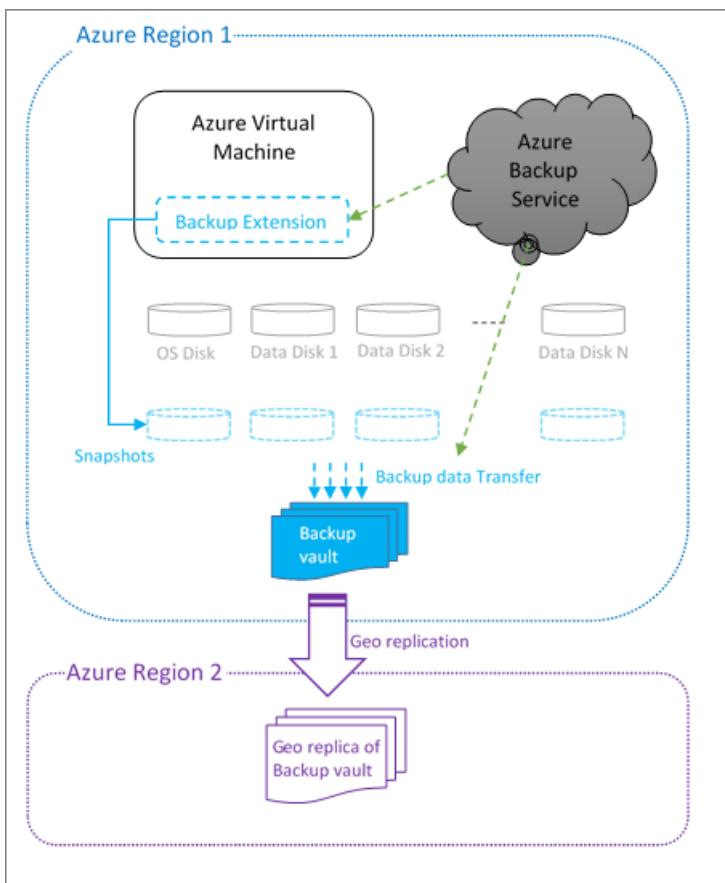
Your choices for high availability, backup, and DR at application or infrastructure levels can be represented as follows:

LEVEL	HIGH AVAILABILITY	BACKUP OR DR
Application	SQL Server AlwaysOn	Azure Backup
Infrastructure	Availability set	Geo-redundant storage with consistent snapshots

Using Azure Backup

[Azure Backup](#) can back up your VMs running Windows or Linux to the Azure recovery services vault. Backing up and restoring business-critical data is complicated by the fact that business-critical data must be backed up while the applications that produce the data are running.

To address this issue, Azure Backup provides application-consistent backups for Microsoft workloads. It uses the volume shadow service to ensure that data is written correctly to storage. For Linux VMs, only file-consistent backups are possible, because Linux does not have functionality equivalent to the volume shadow service.



When Azure Backup initiates a backup job at the scheduled time, it triggers the backup extension installed in the VM to take a point-in-time snapshot. A snapshot is taken in coordination with the volume shadow service to get a consistent snapshot of the disks in the virtual machine without having to shut it down. The backup extension in the VM flushes all writes before taking a consistent snapshot of all of the disks. After taking the snapshot, the data is transferred by Azure Backup to the backup vault. To make the backup process more efficient, the service identifies and transfers only the blocks of data that have changed after the last backup.

To restore, you can view the available backups through Azure Backup and then initiate a restore. You can create and restore Azure backups through the [Azure portal](#), by [using PowerShell](#), or by using the [Azure CLI](#).

Steps to enable a backup

Use the following steps to enable backups of your VMs by using the [Azure portal](#). There is some variation depending on your exact scenario. Refer to the [Azure Backup](#) documentation for full details. Azure Backup also [supports VMs with managed disks](#).

1. Create a recovery services vault for a VM:
 - a. In the [Azure portal](#), browse **All resources** and find **Recovery Services vaults**.
 - b. On the **Recovery Services vaults** menu, click **Add** and follow the steps to create a new vault in the same region as the VM. For example, if your VM is in the West US region, pick West US for the vault.
2. Verify the storage replication for the newly created vault. Access the vault under **Recovery Services vaults** and go to **Settings > Backup Configuration**. Ensure the **geo-redundant storage** option is selected by default. This ensures that your vault is automatically replicated to a secondary datacenter. For example, your vault in West US is automatically replicated to East US.
3. Configure the backup policy and select the VM from the same UI.
4. Make sure the Backup Agent is installed on the VM. If your VM is created by using an Azure gallery image, then the Backup Agent is already installed. Otherwise (that is, if you use a custom image), use the instructions to [install the VM agent on a virtual machine](#).

5. Make sure that the VM allows network connectivity for the backup service to function. Follow the instructions for [network connectivity](#).
6. After the previous steps are completed, the backup runs at regular intervals as specified in the backup policy. If necessary, you can trigger the first backup manually from the vault dashboard on the Azure portal.

For automating Azure Backup by using scripts, refer to [PowerShell cmdlets for VM backup](#).

Steps for recovery

If you need to repair or rebuild a VM, you can restore the VM from any of the backup recovery points in the vault. There are a couple of different options for performing the recovery:

- You can create a new VM as a point-in-time representation of your backed-up VM.
- You can restore the disks, and then use the template for the VM to customize and rebuild the restored VM.

For more information, see the instructions to [use the Azure portal to restore virtual machines](#). This document also explains the specific steps for restoring backed-up VMs to a paired datacenter by using your geo-redundant backup vault if there is a disaster at the primary datacenter. In that case, Azure Backup uses the Compute service from the secondary region to create the restored virtual machine.

You can also use PowerShell for [restoring a VM](#) or for [creating a new VM from restored disks](#).

Alternative solution: Consistent snapshots

If you are unable to use Azure Backup, you can implement your own backup mechanism by using snapshots. Creating consistent snapshots for all the disks used by a VM and then replicating those snapshots to another region is complicated. For this reason, Azure considers using the Backup service as a better option than building a custom solution.

If you use read-access geo-redundant storage/geo-redundant storage for disks, snapshots are automatically replicated to a secondary datacenter. If you use locally redundant storage for disks, you need to replicate the data yourself. For more information, see [Back up Azure-unmanaged VM disks with incremental snapshots](#).

A snapshot is a representation of an object at a specific point in time. A snapshot incurs billing for the incremental size of the data it holds. For more information, see [Create a blob snapshot](#).

Create snapshots while the VM is running

Although you can take a snapshot at any time, if the VM is running, there is still data being streamed to the disks, and the snapshots might contain partial operations that were in flight. Also, if there are several disks involved, the snapshots of different disks might have occurred at different times. This means these snapshots might not be coordinated. This is especially problematic for striped volumes whose files might be corrupted if changes were being made during backup.

To avoid this situation, the backup process must implement the following steps:

1. Freeze all the disks.
2. Flush all the pending writes.
3. [Create a blob snapshot](#) for all the disks.

Some Windows applications, like SQL Server, provide a coordinated backup mechanism via a volume shadow service to create application-consistent backups. On Linux, you can use a tool like `fsfreeze` for coordinating the disks. This tool provides file-consistent backups, but not application-consistent snapshots. This process is complex, so you should consider using [Azure Backup](#) or a third-party backup solution that already implements this procedure.

The previous process results in a collection of coordinated snapshots for all of the VM disks, representing a specific

point-in-time view of the VM. This is a backup restore point for the VM. You can repeat the process at scheduled intervals to create periodic backups. See [Copy the backups to another region](#) for steps to copy the snapshots to another region for DR.

Create snapshots while the VM is offline

Another option to create consistent backups is to shut down the VM and take blob snapshots of each disk. Taking blob snapshots is easier than coordinating snapshots of a running VM, but it requires a few minutes of downtime.

1. Shut down the VM.
2. Create a snapshot of each virtual hard drive blob, which only takes a few seconds.

To create a snapshot, you can use [PowerShell](#), the [Azure Storage REST API](#), [Azure CLI](#), or one of the Azure Storage client libraries, such as [the Storage client library for .NET](#).

3. Start the VM, which ends the downtime. Typically, the entire process finishes within a few minutes.

This process yields a collection of consistent snapshots for all the disks, providing a backup restore point for the VM.

Copy the snapshots to another region

Creation of the snapshots alone might not be sufficient for DR. You must also replicate the snapshot backups to another region.

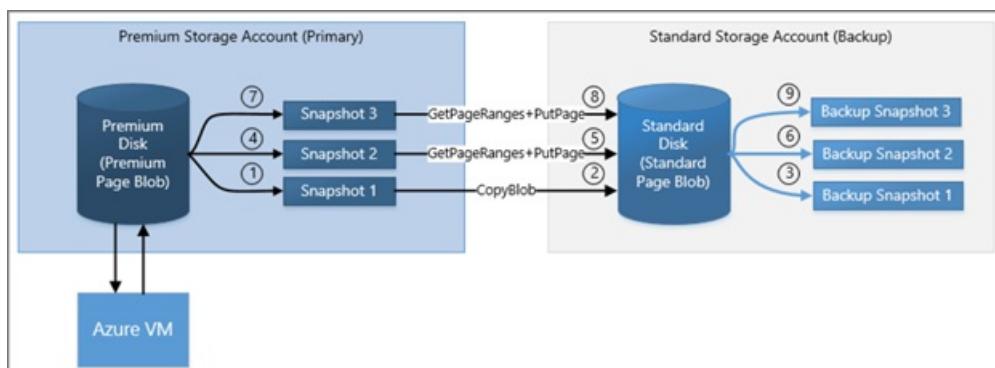
If you use geo-redundant storage or read-access geo-redundant storage for your disks, then the snapshots are replicated to the secondary region automatically. There can be a few minutes of lag before the replication. If the primary datacenter goes down before the snapshots finish replicating, you cannot access the snapshots from the secondary datacenter. The likelihood of this is small.

NOTE

Only having the disks in a geo-redundant storage or read-access geo-redundant storage account does not protect the VM from disasters. You must also create coordinated snapshots or use Azure Backup. This is required to recover a VM to a consistent state.

If you use locally redundant storage, you must copy the snapshots to a different storage account immediately after creating the snapshot. The copy target might be a locally redundant storage account in a different region, resulting in the copy being in a remote region. You can also copy the snapshot to a read-access geo-redundant storage account in the same region. In this case, the snapshot is lazily replicated to the remote secondary region. Your backup is protected from disasters at the primary site after the copying and replication is complete.

To copy your incremental snapshots for DR efficiently, review the instructions in [Back up Azure unmanaged VM disks with incremental snapshots](#).



Recovery from snapshots

To retrieve a snapshot, copy it to make a new blob. If you are copying the snapshot from the primary account, you

can copy the snapshot over to the base blob of the snapshot. This process reverts the disk to the snapshot. This process is known as promoting the snapshot. If you are copying the snapshot backup from a secondary account, in the case of a read-access geo-redundant storage account, you must copy it to a primary account. You can copy a snapshot by [using PowerShell](#) or by using the AzCopy utility. For more information, see [Transfer data with the AzCopy command-line utility](#).

For VMs with multiple disks, you must copy all the snapshots that are part of the same coordinated restore point. After you copy the snapshots to writable VHD blobs, you can use the blobs to recreate your VM by using the template for the VM.

Other options

SQL Server

SQL Server running in a VM has its own built-in capabilities to back up your SQL Server database to Azure Blob storage or a file share. If the storage account is geo-redundant storage or read-access geo-redundant storage, you can access those backups in the storage account's secondary datacenter in the event of a disaster, with the same restrictions as previously discussed. For more information, see [Back up and restore for SQL Server in Azure virtual machines](#). In addition to back up and restore, [SQL Server AlwaysOn availability groups](#) can maintain secondary replicas of databases. This greatly reduces the disaster recovery time.

Other considerations

This article has discussed how to back up or take snapshots of your VMs and their disks to support disaster recovery and how to use those to recover your data. With the Azure Resource Manager model, many people use templates to create their VMs and other infrastructures in Azure. You can use a template to create a VM that has the same configuration every time. If you use custom images for creating your VMs, you must also make sure that your images are protected by using a read-access geo-redundant storage account to store them.

Consequently, your backup process can be a combination of two things:

- Back up the data (disks).
- Back up the configuration (templates and custom images).

Depending on the backup option you choose, you might have to handle the backup of both the data and the configuration, or the backup service might handle all of that for you.

Appendix: Understanding the impact of data redundancy

For storage accounts in Azure, there are three types of data redundancy that you should consider regarding disaster recovery: locally redundant, geo-redundant, or geo-redundant with read access.

Locally redundant storage retains three copies of the data in the same datacenter. When the VM writes the data, all three copies are updated before success is returned to the caller, so you know they are identical. Your disk is protected from local failures, because it's extremely unlikely that all three copies are affected at the same time. In the case of locally redundant storage, there is no geo-redundancy, so the disk is not protected from catastrophic failures that can affect an entire datacenter or storage unit.

With geo-redundant storage and read-access geo-redundant storage, three copies of your data are retained in the primary region that is selected by you. Three more copies of your data are retained in a corresponding secondary region that is set by Azure. For example, if you store data in West US, the data is replicated to East US. Copy retention is done asynchronously, and there is a small delay between updates to the primary and secondary sites. Replicas of the disks on the secondary site are consistent on a per-disk basis (with the delay), but replicas of multiple active disks might not be in sync with each other. To have consistent replicas across multiple disks, consistent snapshots are needed.

The main difference between geo-redundant storage and read-access geo-redundant storage is that with read-access geo-redundant storage, you can read the secondary copy at any time. If there is a problem that renders the data in the primary region inaccessible, the Azure team makes every effort to restore access. While the primary is down, if you have read-access geo-redundant storage enabled, you can access the data in the secondary datacenter. Therefore, if you plan to read from the replica while the primary is inaccessible, then read-access geo-redundant storage should be considered.

If it turns out to be a significant outage, the Azure team might trigger a geo-failover and change the primary DNS entries to point to secondary storage. At this point, if you have either geo-redundant storage or read-access geo-redundant storage enabled, you can access the data in the region that used to be the secondary. In other words, if your storage account is geo-redundant storage and there is a problem, you can access the secondary storage only if there is a geo-failover.

For more information, see [What to do if an Azure Storage outage occurs](#).

NOTE

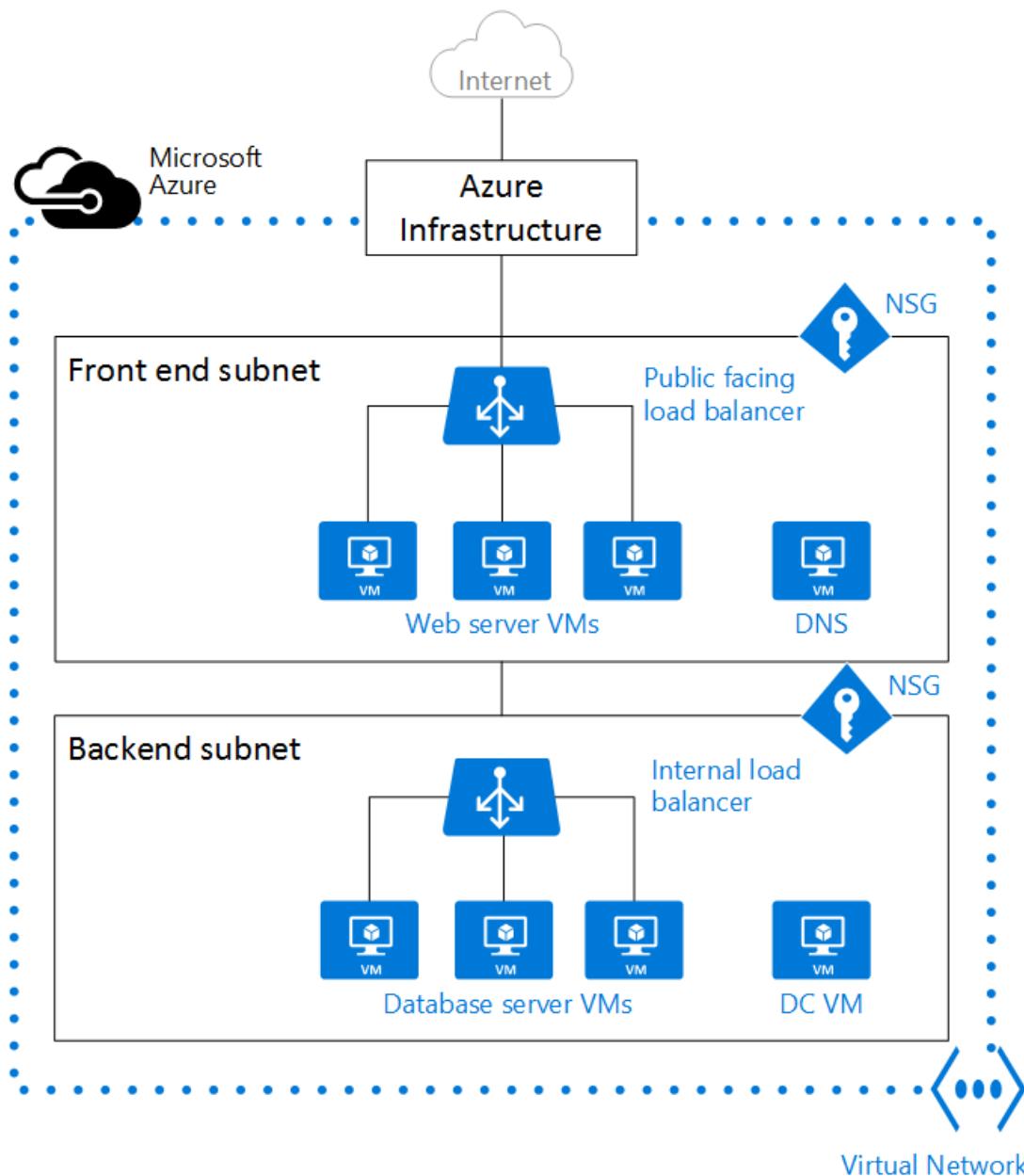
Microsoft controls whether a failover occurs. Failover is not controlled per storage account, so it's not decided by individual customers. To implement disaster recovery for specific storage accounts or virtual machine disks, you must use the techniques described previously in this article.

Virtual networks and virtual machines in Azure

9/14/2017 • 13 min to read • [Edit Online](#)

When you create an Azure virtual machine (VM), you must create a [virtual network](#) (VNet) or use an existing VNet. You also need to decide how your VMs are intended to be accessed on the VNet. It is important to [plan before creating resources](#) and make sure that you understand the [limits of networking resources](#).

In the following figure, VMs are represented as web servers and database servers. Each set of VMs are assigned to separate subnets in the VNet.



You can create a VNet before you create a VM or you can as you create a VM. You create these resources to support communication with a VM:

- Network interfaces
- IP addresses
- Virtual network and subnets

In addition to those basic resources, you should also consider these optional resources:

- Network security groups
- Load balancers

Network interfaces

A [network interface \(NIC\)](#) is the interconnection between a VM and a virtual network (VNet). A VM must have at least one NIC, but can have more than one, depending on the size of the VM you create. Learn about how many NICs each VM size supports for [Windows](#) or [Linux](#).

You can create a VM with multiple NICs, and add or remove NICs through the lifecycle of a VM. Multiple NICs allow a VM to connect to different subnets and send or receive traffic over the most appropriate interface.

If the VM is added to an availability set, all VMs within the availability set must have one or multiple NICs. VMs with more than one NIC aren't required to have the same number of NICs, but they must all have at least two.

Each NIC attached to a VM must exist in the same location and subscription as the VM. Each NIC must be connected to a VNet that exists in the same Azure location and subscription as the NIC. You can change the subnet a VM is connected to after it's created, but you cannot change the VNet. Each NIC attached to a VM is assigned a MAC address that doesn't change until the VM is deleted.

This table lists the methods that you can use to create a network interface.

METHOD	DESCRIPTION
Azure portal	When you create a VM in the Azure portal, a network interface is automatically created for you (you cannot use a NIC you create separately). The portal creates a VM with only one NIC. If you want to create a VM with more than one NIC, you must create it with a different method.
Azure PowerShell	Use New-AzureRmNetworkInterface with the -PublicIpAddressId parameter to provide the identifier of the public IP address that you previously created.
Azure CLI	To provide the identifier of the public IP address that you previously created, use az network nic create with the --public-ip-address parameter.
Template	Use Network Interface in a Virtual Network with Public IP Address as a guide for deploying a network interface using a template.

IP addresses

You can assign these types of [IP addresses](#) to a NIC in Azure:

- **Public IP addresses** - Used to communicate inbound and outbound (without network address translation (NAT)) with the Internet and other Azure resources not connected to a VNet. Assigning a public IP address to a NIC is optional. Public IP addresses have a nominal charge, and there's a maximum number that can be used per subscription.
- **Private IP addresses** - Used for communication within a VNet, your on-premises network, and the Internet (with NAT). You must assign at least one private IP address to a VM. To learn more about NAT in Azure, read [Understanding outbound connections in Azure](#).

You can assign public IP addresses to VMs or internet-facing load balancers. You can assign private IP addresses to VMs and internal load balancers. You assign IP addresses to a VM using a network interface.

There are two methods in which an IP address is allocated to a resource - dynamic or static. The default allocation method is dynamic, where an IP address is not allocated when it's created. Instead, the IP address is allocated when you create a VM or start a stopped VM. The IP address is released when you stop or delete the VM.

To ensure the IP address for the VM remains the same, you can set the allocation method explicitly to static. In this case, an IP address is assigned immediately. It is released only when you delete the VM or change its allocation method to dynamic.

This table lists the methods that you can use to create an IP address.

METHOD	DESCRIPTION
Azure portal	By default, public IP addresses are dynamic and the address associated to them may change when the VM is stopped or deleted. To guarantee that the VM always uses the same public IP address, create a static public IP address. By default, the portal assigns a dynamic private IP address to a NIC when creating a VM. You can change this IP address to static after the VM is created.
Azure PowerShell	You use New-AzureRmPublicIpAddress with the -AllocationMethod parameter as Dynamic or Static.
Azure CLI	You use az network public-ip create with the --allocation-method parameter as Dynamic or Static.
Template	Use Network Interface in a Virtual Network with Public IP Address as a guide for deploying a public IP address using a template.

After you create a public IP address, you can associate it with a VM by assigning it to a NIC.

Virtual network and subnets

A subnet is a range of IP addresses in the VNet. You can divide a VNet into multiple subnets for organization and security. Each NIC in a VM is connected to one subnet in one VNet. NICs connected to subnets (same or different) within a VNet can communicate with each other without any extra configuration.

When you set up a VNet, you specify the topology, including the available address spaces and subnets. If the VNet is to be connected to other VNets or on-premises networks, you must select address ranges that don't overlap. The IP addresses are private and can't be accessed from the Internet, which was true only for the non-routeable IP addresses such as 10.0.0.0/8, 172.16.0.0/12, or 192.168.0.0/16. Now, Azure treats any address range as part of the private VNet IP address space that is only reachable within the VNet, within interconnected VNets, and from your on-premises location.

If you work within an organization in which someone else is responsible for the internal networks, you should talk to that person before selecting your address space. Make sure there is no overlap and let them know the space you want to use so they don't try to use the same range of IP addresses.

By default, there is no security boundary between subnets, so VMs in each of these subnets can talk to one another. However, you can set up Network Security Groups (NSGs), which allow you to control the traffic flow to and from subnets and to and from VMs.

This table lists the methods that you can use to create a VNet and subnets.

METHOD	DESCRIPTION
Azure portal	If you let Azure create a VNet when you create a VM, the name is a combination of the resource group name that contains the VNet and -vnet . The address space is 10.0.0.0/24, the required subnet name is default , and the subnet address range is 10.0.0.0/24.
Azure PowerShell	You use New-AzureRmVirtualNetworkSubnetConfig and New-AzureRmVirtualNetwork to create a subnet and a VNet. You can also use Add-AzureRmVirtualNetworkSubnetConfig to add a subnet to an existing VNet.
Azure CLI	The subnet and the VNet are created at the same time. Provide a --subnet-name parameter to az network vnet create with the subnet name.
Template	The easiest way to create a VNet and subnets is to download an existing template, such as Virtual Network with two Subnets , and modify it for your needs.

Network security groups

A [network security group \(NSG\)](#) contains a list of Access Control List (ACL) rules that allow or deny network traffic to subnets, NICs, or both. NSGs can be associated with either subnets or individual NICs connected to a subnet. When an NSG is associated with a subnet, the ACL rules apply to all the VMs in that subnet. In addition, traffic to an individual NIC can be restricted by associating an NSG directly to a NIC.

NSGs contain two sets of rules: inbound and outbound. The priority for a rule must be unique within each set. Each rule has properties of protocol, source and destination port ranges, address prefixes, direction of traffic, priority, and access type.

All NSGs contain a set of default rules. The default rules cannot be deleted, but because they are assigned the lowest priority, they can be overridden by the rules that you create.

When you associate an NSG to a NIC, the network access rules in the NSG are applied only to that NIC. If an NSG is applied to a single NIC on a multi-NIC VM, it does not affect traffic to the other NICs. You can associate different NSGs to a NIC (or VM, depending on the deployment model) and the subnet that a NIC or VM is bound to. Priority is given based on the direction of traffic.

Be sure to [plan](#) your NSGs when you plan your VMs and VNet.

This table lists the methods that you can use to create a network security group.

METHOD	DESCRIPTION
Azure portal	When you create a VM in the Azure portal, an NSG is automatically created and associated to the NIC the portal creates. The name of the NSG is a combination of the name of the VM and -nsg . This NSG contains one inbound rule with a priority of 1000, service set to RDP, the protocol set to TCP, port set to 3389, and action set to Allow. If you want to allow any other inbound traffic to the VM, you must add additional rules to the NSG.

METHOD	DESCRIPTION
Azure PowerShell	Use New-AzureRmNetworkSecurityRuleConfig and provide the required rule information. Use New-AzureRmNetworkSecurityGroup to create the NSG. Use Set-AzureRmVirtualNetworkSubnetConfig to configure the NSG for the subnet. Use Set-AzureRmVirtualNetwork to add the NSG to the VNet.
Azure CLI	Use az network nsg create to initially create the NSG. Use az network nsg rule create to add rules to the NSG. Use az network vnet subnet update to add the NSG to the subnet.
Template	Use Create a Network Security Group as a guide for deploying a network security group using a template.

Load balancers

[Azure Load Balancer](#) delivers high availability and network performance to your applications. A load balancer can be configured to [balance incoming Internet traffic](#) to VMs or [balance traffic between VMs in a VNet](#). A load balancer can also balance traffic between on-premises computers and VMs in a cross-premises network, or forward external traffic to a specific VM.

The load balancer maps incoming and outgoing traffic between the public IP address and port on the load balancer and the private IP address and port of the VM.

When you create a load balancer, you must also consider these configuration elements:

- **Front-end IP configuration** – A load balancer can include one or more front-end IP addresses, otherwise known as virtual IPs (VIPs). These IP addresses serve as ingress for the traffic.
- **Back-end address pool** – IP addresses that are associated with the NIC to which load is distributed.
- **NAT rules** - Defines how inbound traffic flows through the front-end IP and distributed to the back-end IP.
- **Load balancer rules** - Maps a given front-end IP and port combination to a set of back-end IP addresses and port combination. A single load balancer can have multiple load balancing rules. Each rule is a combination of a front-end IP and port and back-end IP and port associated with VMs.
- **Probes** - Monitors the health of VMs. When a probe fails to respond, the load balancer stops sending new connections to the unhealthy VM. The existing connections are not affected, and new connections are sent to healthy VMs.

This table lists the methods that you can use to create an internet-facing load balancer.

METHOD	DESCRIPTION
Azure portal	You can't currently create an internet-facing load balancer using the Azure portal.

METHOD	DESCRIPTION
Azure PowerShell	To provide the identifier of the public IP address that you previously created, use New-AzureRmLoadBalancerFrontendIpConfig with the -PublicIpAddress parameter. Use New-AzureRmLoadBalancerBackendAddressPoolConfig to create the configuration of the back-end address pool. Use New-AzureRmLoadBalancerInboundNatRuleConfig to create inbound NAT rules associated with the front-end IP configuration that you created. Use New-AzureRmLoadBalancerProbeConfig to create the probes that you need. Use New-AzureRmLoadBalancerRuleConfig to create the load balancer configuration. Use New-AzureRmLoadBalancer to create the load balancer.
Azure CLI	Use <code>az network lb create</code> to create the initial load balancer configuration. Use <code>az network lb frontend-ip create</code> to add the public IP address that you previously created. Use <code>az network lb address-pool create</code> to add the configuration of the back-end address pool. Use <code>az network lb inbound-nat-rule create</code> to add NAT rules. Use <code>az network lb rule create</code> to add the load balancer rules. Use <code>az network lb probe create</code> to add the probes.
Template	Use 2 VMs in a Load Balancer and configure NAT rules on the LB as a guide for deploying a load balancer using a template.

This table lists the methods that you can use to create an internal load balancer.

METHOD	DESCRIPTION
Azure portal	You can't currently create an internal load balancer using the Azure portal.
Azure PowerShell	To provide a private IP address in the network subnet, use New-AzureRmLoadBalancerFrontendIpConfig with the -PrivateIpAddress parameter. Use New-AzureRmLoadBalancerBackendAddressPoolConfig to create the configuration of the back-end address pool. Use New-AzureRmLoadBalancerInboundNatRuleConfig to create inbound NAT rules associated with the front-end IP configuration that you created. Use New-AzureRmLoadBalancerProbeConfig to create the probes that you need. Use New-AzureRmLoadBalancerRuleConfig to create the load balancer configuration. Use New-AzureRmLoadBalancer to create the load balancer.
Azure CLI	Use the <code>az network lb create</code> command to create the initial load balancer configuration. To define the private IP address, use <code>az network lb frontend-ip create</code> with the --private-ip-address parameter. Use <code>az network lb address-pool create</code> to add the configuration of the back-end address pool. Use <code>az network lb inbound-nat-rule create</code> to add NAT rules. Use <code>az network lb rule create</code> to add the load balancer rules. Use <code>az network lb probe create</code> to add the probes.

Template

Use [2 VMs in a Load Balancer and configure NAT rules on the LB](#) as a guide for deploying a load balancer using a template.

VMs

VMs can be created in the same VNet and they can connect to each other using private IP addresses. They can connect even if they are in different subnets without the need to configure a gateway or use public IP addresses. To put VMs into a VNet, you create the VNet and then as you create each VM, you assign it to the VNet and subnet. VMs acquire their network settings during deployment or startup.

VMs are assigned an IP address when they are deployed. If you deploy multiple VMs into a VNet or subnet, they are assigned IP addresses as they boot up. A dynamic IP address (DIP) is the internal IP address associated with a VM. You can allocate a static DIP to a VM. If you allocate a static DIP, you should consider using a specific subnet to avoid accidentally reusing a static DIP for another VM.

If you create a VM and later want to migrate it into a VNet, it is not a simple configuration change. You must redeploy the VM into the VNet. The easiest way to redeploy is to delete the VM, but not any disks attached to it, and then re-create the VM using the original disks in the VNet.

This table lists the methods that you can use to create a VM in a VNet.

METHOD	DESCRIPTION
Azure portal	Uses the default network settings that were previously mentioned to create a VM with a single NIC. To create a VM with multiple NICs, you must use a different method.
Azure PowerShell	Includes the use of <code>Add-AzureRmVMNetworkInterface</code> to add the NIC that you previously created to the VM configuration.
Azure CLI	Create and connect a VM to a Vnet, subnet, and NIC that build as individual steps.
Template	Use Very simple deployment of a Windows VM as a guide for deploying a VM using a template.

Next steps

For VM-specific steps on how to manage Azure virtual networks for VMs, see the [Windows](#) or [Linux](#) tutorials.

There are also tutorials on how to load balance VMs and create highly available applications for [Windows](#) or [Linux](#).

- Learn how to configure [user-defined routes and IP forwarding](#).
- Learn how to configure [VNet to VNet connections](#).
- Learn how to [Troubleshoot routes](#).

Automatically scale virtual machines in Azure

9/14/2017 • 4 min to read • [Edit Online](#)

You can easily [automatically scale](#) your [virtual machines \(VMs\)](#) when you use [virtual machine scale sets](#) and the [autoscaling feature of Azure Monitor](#). Your VMs need to be members of a scale set to be automatically scaled. This article provides information that enables you to better understand how to scale your VMs both vertically and horizontally using automatic and manual methods.

Horizontal or vertical scaling

The autoscale feature of Azure Monitor only scales horizontally, which is an increase ("out") or decrease ("in") of the number of VMs. Horizontal scaling is more flexible in a cloud situation as it allows you to run potentially thousands of VMs to handle load. You scale horizontally by either automatically or manually changing the capacity (or instance count) of the the scale set.

Vertical scaling keeps the same number of VMs, but makes the VMs more ("up") or less ("down") powerful. Power is measured in attributes such as memory, CPU speed, or disk space. Vertical scaling is dependent on the availability of larger hardware, which quickly hits an upper limit and can vary by region. Vertical scaling also usually requires a VM to stop and restart. You scale vertically by setting a new size in the configuration of the VMs in the scale set.

Using runbooks in [Azure Automation](#), you can easily [scale VMs in a scale set](#) up or down.

Create a virtual machine scale set

Scale sets make it easy for you to deploy and manage identical VMs as a set. You can create Linux or Windows scale sets using the [Azure portal](#), [Azure PowerShell](#), or the [Azure CLI](#). You can also create and manage scale sets with SDKs such as [Python](#) or [Node.js](#), or directly with the [REST APIs](#). Automatic scaling of VMs is accomplished by applying metrics and rules to the scale set.

Configure autoscale for a scale set

Automatic scaling provides the right number of VMs to handle the load on your application. It enables you to add VMs to handle increases in load and save money by removing VMs that are sitting idle. You specify a minimum and maximum number of VMs to run based on a set of rules. Having a minimum makes sure your application is always running even under no load. Having a maximum value limits your total possible hourly cost.

You can enable autoscale when you create the scale set using [Azure PowerShell](#) or [Azure CLI](#). You can also enable it after the scale set is created. You can create a scale set, install the extension, and configure autoscale using an [Azure Resource Manager template](#). In the Azure portal, enable autoscale from Azure Monitor, or enable autoscale from the scale set settings.

The screenshot shows the Azure portal interface for managing a virtual machine scale set. The left sidebar has a red box around the 'Scaling' tab. The main content area shows the 'Configure' tab selected. There is an 'Override condition' section with a slider for 'Instance count' set to 3. Below it, a message says 'Your autoscale configuration is disabled. To reinstate your configuration, enable autoscale.' A red box highlights the 'Enable autoscale' button.

Metrics

The autoscale feature of Azure Monitor enables you to scale the number of running VMs up or down based on **metrics**. By default, VMs provide basic host-level metrics for disk, network, and CPU usage. When you configure the collection of diagnostics data using the diagnostic extension, additional guest OS performance counters become available for disk, CPU, and memory.

The screenshot shows the 'Criteria' configuration page. It includes fields for 'Time aggregation' (set to 'Average'), 'Metric name' (set to 'Percentage CPU' with a red box around it), 'Time grain statistic' (set to 'Average'), 'Operator' (set to 'Greater than'), 'Threshold' (set to '85'), and 'Duration (in minutes)' (set to '30').

If your application needs to scale based on metrics that are not available through the host, then the VMs in the scale set need to have either the [Linux diagnostic extension](#) or [Windows diagnostics extension](#) installed. If you create a scale set using the Azure portal, you need to also use Azure PowerShell or the Azure CLI to install the extension with the diagnostics configuration that you need.

Rules

Rules combine a metric with an action to be performed. When rule conditions are met, one or more autoscale actions are triggered. For example, you might have a rule defined that increases the number of VMs by 1 if the average CPU usage goes above 85 percent.

The screenshot shows the 'Action' configuration page. It includes fields for 'Operation' (set to 'Increase count by'), 'Instance count' (set to '1' with a red box around it), and 'Cool down (minutes)' (set to '5').

Notifications

You can set up [triggers](#) so that specific web URLs are called or emails are sent based on the autoscale rules that you create. Webhooks allow you to route the Azure alert notifications to other systems for post-processing or custom notifications.

Manually scale VMs in a scale set

Horizontal

You can add or remove VMs by changing the capacity of the scale set. In the Azure portal, you can decrease or increase the number of VMs (shown as **instance count**) in the scale set by sliding the Override condition bar on the Scaling screen left or right.

Using Azure PowerShell, you need to get the scale set object using [Get-AzureRmVmss](#). You then set the **sku.capacity** property to the number of VMs that you want and update the scale set with [Update-AzureRmVmss](#). Using Azure CLI, you change the capacity with the **--new-capacity** parameter for the [az vmss scale](#) command.

Vertical

You can manually change the size of the VMs in the Azure portal on the Size screen for the scale set. You can use Azure PowerShell with [Get-AzureRmVmss](#), setting the image reference sku property, and then using [Update-AzureRmVmss](#) and [Update-AzureRmVmssInstance](#).

Next steps

- Learn more about scale sets in [Design Considerations for Scale Sets](#).

Use infrastructure automation tools with virtual machines in Azure

12/13/2017 • 7 min to read • [Edit Online](#)

To create and manage Azure virtual machines (VMs) in a consistent manner at scale, some form of automation is typically desired. There are many tools and solutions that allow you to automate the complete Azure infrastructure deployment and management lifecycle. This article introduces some of the infrastructure automation tools that you can use in Azure. These tools commonly fit in to one of the following approaches:

- Automate the configuration of VMs
 - Tools include [Ansible](#), [Chef](#), and [Puppet](#).
 - Tools specific to VM customization include [cloud-init](#) for Linux VMs, [PowerShell Desired State Configuration \(DSC\)](#), and the [Azure Custom Script Extension](#) for all Azure VMs.
- Automate infrastructure management
 - Tools include [Packer](#) to automate custom VM image builds, and [Terraform](#) to automate the infrastructure build process.
 - [Azure Automation](#) can perform actions across your Azure and on-prem infrastructure.
- Automate application deployment and delivery
 - Examples include [Visual Studio Team Services](#) and [Jenkins](#).

Ansible

[Ansible](#) is an automation engine for configuration management, VM creation, or application deployment. Ansible uses an agent-less model, typically with SSH keys, to authenticate and manage target machines. Configuration tasks are defined in runbooks, with a number of Ansible modules available to carry out specific tasks. For more information, see [How Ansible works](#).

Learn how to:

- [Install and configure Ansible on Linux for use with Azure](#).
- [Create a basic VM](#).
- [Create a complete VM environment including supporting resources](#).

Chef

[Chef](#) is an automation platform that helps define how your infrastructure is configured, deployed, and managed. Additional components included Chef Habitat for application lifecycle automation rather than the infrastructure, and Chef InSpec that helps automate compliance with security and policy requirements. Chef Clients are installed on target machines, with one or more central Chef Servers that store and manage the configurations. For more information, see [An Overview of Chef](#).

Learn how to:

- [Deploy Chef Automate from the Azure Marketplace](#).
- [Install Chef on Windows and create Azure VMs](#).

Puppet

Puppet is an enterprise-ready automation platform that handles the application delivery and deployment process. Agents are installed on target machines to allow Puppet Master to run manifests that define the desired configuration of the Azure infrastructure and VMs. Puppet can integrate with other solutions such as Jenkins and GitHub for an improved devops workflow. For more information, see [How Puppet works](#).

Learn how to:

- [Deploy Puppet from the Azure Marketplace.](#)

Cloud-init

[Cloud-init](#) is a widely used approach to customize a Linux VM as it boots for the first time. You can use cloud-init to install packages and write files, or to configure users and security. Because cloud-init is called during the initial boot process, there are no additional steps or required agents to apply your configuration. For more information on how to properly format your `#cloud-config` files, see the [cloud-init documentation site](#). `#cloud-config` files are text files encoded in base64.

Cloud-init also works across distributions. For example, you don't use **apt-get install** or **yum install** to install a package. Instead you can define a list of packages to install. Cloud-init automatically uses the native package management tool for the distro you select.

We are actively working with our endorsed Linux distro partners in order to have cloud-init enabled images available in the Azure marketplace. These images make your cloud-init deployments and configurations work seamlessly with VMs and virtual machine scale sets. The following table outlines the current cloud-init enabled images availability on the Azure platform:

PUBLISHER	OFFER	SKU	VERSION	CLOUD-INIT READY
Canonical	UbuntuServer	16.04-LTS	latest	yes
Canonical	UbuntuServer	14.04.5-LTS	latest	yes
CoreOS	CoreOS	Stable	latest	yes
OpenLogic	CentOS	7-CI	latest	preview
RedHat	RHEL	7-RAW-CI	latest	preview

Learn more details about cloud-init on Azure:

- [Cloud-init support for Linux virtual machines in Azure](#)
- [Try a tutorial on automated VM configuration using cloud-init.](#)

PowerShell DSC

[PowerShell Desired State Configuration \(DSC\)](#) is a management platform to define the configuration of target machines. DSC can also be used on Linux through the [Open Management Infrastructure \(OMI\) server](#).

DSC configurations define what to install on a machine and how to configure the host. A Local Configuration Manager (LCM) engine runs on each target node that processes requested actions based on pushed configurations. A pull server is a web service that runs on a central host to store the DSC configurations and associated resources. The pull server communicates with the LCM engine on each target host to provide the required configurations and report on compliance.

Learn how to:

- [Create a basic DSC configuration.](#)
- [Configure a DSC pull server.](#)
- [Use DSC for Linux.](#)

Azure Custom Script Extension

The Azure Custom Script Extension for [Linux](#) or [Windows](#) downloads and executes scripts on Azure VMs. You can use the extension when you create a VM, or any time after the VM is in use.

Scripts can be downloaded from Azure storage or any public location such as a GitHub repository. With the Custom Script Extension, you can write scripts in any language that runs on the source VM. These scripts can be used to install applications or configure the VM as desired. To secure credentials, sensitive information such as passwords can be stored in a protected configuration. These credentials are only decrypted inside the VM.

Learn how to:

- [Create a Linux VM with the Azure CLI and use the Custom Script Extension.](#)
- [Create a Windows VM with Azure PowerShell and use the Custom Script Extension.](#)

Packer

[Packer](#) automates the build process when you create a custom VM image in Azure. You use Packer to define the OS and run post-configuration scripts that customize the VM for your specific needs. Once configured, the VM is then captured as a Managed Disk image. Packer automates the process to create the source VM, network and storage resources, run configuration scripts, and then create the VM image.

Learn how to:

- [Use Packer to create a Linux VM image in Azure.](#)
- [Use Packer to create a Windows VM image in Azure.](#)

Terraform

[Terraform](#) is an automation tool that allows you to define and create an entire Azure infrastructure with a single template format language - the HashiCorp Configuration Language (HCL). With Terraform, you define templates that automate the process to create network, storage, and VM resources for a given application solution. You can use your existing Terraform templates for other platforms with Azure to ensure consistency and simplify the infrastructure deployment without needing to convert to an Azure Resource Manager template.

Learn how to:

- [Install and configure Terraform with Azure.](#)
- [Create an Azure infrastructure with Terraform.](#)

Azure Automation

[Azure Automation](#) uses runbooks to process a set of tasks on the VMs you target. Azure Automation is used to manage existing VMs rather than to create an infrastructure. Azure Automation can run across both Linux and Windows VMs, as well as on-prem virtual or physical machines with a hybrid runbook worker. Runbooks can be stored in a source control repository, such as GitHub. These runbooks can then run manually or on a defined schedule.

Azure Automation also provides a Desired State Configuration (DSC) service that allows you to create definitions for how a given set of VMs should be configured. DSC then ensures that the required configuration is applied and the VM stays consistent. Azure Automation DSC runs on both Windows and Linux machines.

Learn how to:

- [Create a PowerShell runbook.](#)
- [Use Hybrid Runbook Worker to manage on-prem resources.](#)
- [Use Azure Automation DSC.](#)

Visual Studio Team Services

[Team Services](#) is a suite of tools that help you share and track code, use automated builds, and create a complete continuous integration and development (CI/CD) pipeline. Team Services integrates with Visual Studio and other editors to simplify usage. Team Services can also create and configure Azure VMs and then deploy code to them.

Learn how to:

- [Create a continuous integration pipeline with Team Services.](#)

Jenkins

[Jenkins](#) is a continuous integration server that helps deploy and test applications, and create automated pipelines for code delivery. There are hundreds of plugins to extend the core Jenkins platform, and you can also integrate with many other products and solutions through webhooks. You can manually install Jenkins on an Azure VM, run Jenkins from within a Docker container, or use a pre-built Azure Marketplace image.

Learn how to:

- [Create a development infrastructure on a Linux VM in Azure with Jenkins, GitHub, and Docker.](#)

Next steps

There are many different options to use infrastructure automation tools in Azure. You have the freedom to use the solution that best fits your needs and environment. To get started and try some of the tools built-in to Azure, see how to automate the customization of a [Linux](#) or [Windows](#) VM.

Secure and use policies on virtual machines in Azure

9/14/2017 • 3 min to read • [Edit Online](#)

It's important to keep your virtual machine (VM) secure for the applications that you run. Securing your VMs can include one or more Azure services and features that cover secure access to your VMs and secure storage of your data. This article provides information that enables you to keep your VM and applications secure.

Antimalware

The modern threat landscape for cloud environments is dynamic, increasing the pressure to maintain effective protection in order to meet compliance and security requirements. [Microsoft Antimalware for Azure](#) is a free real-time protection capability that helps identify and remove viruses, spyware, and other malicious software. Alerts can be configured to notify you when known malicious or unwanted software attempts to install itself or run on your VM.

Azure Security Center

[Azure Security Center](#) helps you prevent, detect, and respond to threats to your VMs. Security Center provides integrated security monitoring and policy management across your Azure subscriptions, helps detect threats that might otherwise go unnoticed, and works with a broad ecosystem of security solutions.

Encryption

For enhanced [Windows VM](#) and [Linux VM](#) security and compliance, virtual disks in Azure can be encrypted. Virtual disks on Windows VMs are encrypted at rest using BitLocker. Virtual disks on Linux VMs are encrypted at rest using dm-crypt.

There is no charge for encrypting virtual disks in Azure. Cryptographic keys are stored in Azure Key Vault using software-protection, or you can import or generate your keys in Hardware Security Modules (HSMs) certified to FIPS 140-2 level 2 standards. These cryptographic keys are used to encrypt and decrypt virtual disks attached to your VM. You retain control of these cryptographic keys and can audit their use. An Azure Active Directory service principal provides a secure mechanism for issuing these cryptographic keys as VMs are powered on and off.

Key Vault and SSH Keys

Secrets and certificates can be modeled as resources and provided by [Key Vault](#). You can use Azure PowerShell to create key vaults for [Windows VMs](#) and the Azure CLI for [Linux VMs](#). You can also create keys for encryption.

Key vault access policies grant permissions to keys, secrets, and certificates separately. For example, you can give a user access to only keys, but no permissions for secrets. However, permissions to access keys or secrets or certificates are at the vault level. In other words, [key vault access policy](#) does not support object level permissions.

When you connect to VMs, you should use public-key cryptography to provide a more secure way to log in to them. This process involves a public and private key exchange using the secure shell (SSH) command to authenticate yourself rather than a username and password. Passwords are vulnerable to brute-force attacks, especially on Internet-facing VMs such as web servers. With a secure shell (SSH) key pair, you can create a [Linux VM](#) that uses SSH keys for authentication, eliminating the need for passwords to log in. You can also use SSH keys to connect from a [Windows VM](#) to a Linux VM.

Policies

Azure policies can be used to define the desired behavior for your organization's [Windows VMs](#) and [Linux VMs](#). By using policies, an organization can enforce various conventions and rules throughout the enterprise. Enforcement of the desired behavior can help mitigate risk while contributing to the success of the organization.

Role-based access control

Using [role-based access control \(RBAC\)](#), you can segregate duties within your team and grant only the amount of access to users on your VM that they need to perform their jobs. Instead of giving everybody unrestricted permissions on the VM, you can allow only certain actions. You can configure access control for the VM in the [Azure portal](#), using the [Azure CLI](#), or [Azure PowerShell](#).

Next steps

- Walk through the steps to monitor virtual machine security by using Azure Security Center for [Linux](#) or [Windows](#).

How to monitor virtual machines in Azure

9/14/2017 • 5 min to read • [Edit Online](#)

You can take advantage of many opportunities to monitor your VMs by collecting, viewing, and analyzing diagnostic and log data. To do simple [monitoring](#) of your VM, you can use the Overview screen for the VM in the Azure portal. You can use [extensions](#) to configure diagnostics on your VMs to collect additional metric data. You can also use more advanced monitoring options, such as [Application Insights](#) and [Log Analytics](#).

Diagnostics and metrics

You can set up and monitor the collection of [diagnostics data](#) using [metrics](#) in the Azure portal, the Azure CLI, Azure PowerShell, and programming Applications Programming Interfaces (APIs). For example, you can:

- **Observe basic metrics for the VM.** On the Overview screen of the Azure portal, the basic metrics shown include CPU usage, network usage, total of disk bytes, and disk operations per second.
- **Enable the collection of boot diagnostics and view it using the Azure portal.** When bringing your own image to Azure or even booting one of the platform images, there can be many reasons why a VM gets into a non-bootable state. You can easily enable boot diagnostics when you create a VM by clicking **Enabled** for Boot Diagnostics under the Monitoring section of the Settings screen.

As VMs boot, the boot diagnostic agent captures boot output and stores it in Azure storage. This data can be used to troubleshoot VM boot issues. Boot diagnostics are not automatically enabled when you create a VM from command-line tools. Before enabling boot diagnostics, a storage account needs to be created for storing boot logs. If you enable boot diagnostics in the Azure portal, a storage account is automatically created for you.

If you didn't enable boot diagnostics when the VM was created, you can always enable it later by using [Azure CLI](#), [Azure PowerShell](#), or an [Azure Resource Manager template](#).

- **Enable the collection of guest OS diagnostics data.** When you create a VM, you have the opportunity on the settings screen to enable guest OS diagnostics. When you do enable the collection of diagnostics data, the [IaaS Diagnostics extension for Linux](#) or the [IaaS Diagnostics extension for Windows](#) is added to the VM, which enables you to collect additional disk, CPU, and memory data.

Using the collected diagnostics data, you can configure autoscaling for your VMs. You can also configure logs to store the data and set up alerts to let you know when performance isn't quite right.

Alerts

You can create [alerts](#) based on specific performance metrics. Examples of the issues you can be alerted about include when average CPU usage exceeds a certain threshold, or available free disk space drops below a certain amount. Alerts can be configured in the [Azure portal](#), using [Azure PowerShell](#), or the [Azure CLI](#).

Azure Service Health

[Azure Service Health](#) provides personalized guidance and support when issues in Azure services affect you, and helps you prepare for upcoming planned maintenance. Azure Service Health alerts you and your teams using targeted and flexible notifications.

Azure Resource Health

Azure Resource health helps you diagnose and get support when an Azure issue impacts your resources. It informs you about the current and past health of your resources and helps you mitigate issues. Resource health provides technical support when you need help with Azure service issues.

Logs

The [Azure Activity Log](#) is a subscription log that provides insight into subscription-level events that have occurred in Azure. The log includes a range of data, from Azure Resource Manager operational data to updates on Service Health events. You can click Activity Log in the Azure portal to view the log for your VM.

Some of the things you can do with the activity log include:

- Create an [alert on an Activity Log event](#).
- [Stream it to an Event Hub](#) for ingestion by a third-party service or custom analytics solution such as PowerBI.
- Analyze it in PowerBI using the [PowerBI content pack](#).
- [Save it to a storage account](#) for archival or manual inspection. You can specify the retention time (in days) using the Log Profile.

You can also access activity log data by using [Azure PowerShell](#), the [Azure CLI](#), or [Monitor REST APIs](#).

[Azure Diagnostic Logs](#) are logs emitted by your VM that provide rich, frequent data about its operation. Diagnostic logs differ from the activity log by providing insight about operations that were performed within the VM.

Some of the things you can do with diagnostics logs include:

- [Save them to a storage account](#) for auditing or manual inspection. You can specify the retention time (in days) using Resource Diagnostic Settings.
- [Stream them to Event Hubs](#) for ingestion by a third-party service or custom analytics solution such as PowerBI.
- Analyze them with [OMS Log Analytics](#).

Advanced monitoring

- [Operations Management Suite \(OMS\)](#) provides monitoring, alerting, and alert remediation capabilities across cloud and on-premises assets. You can install an extension on a [Linux VM](#) or a [Windows VM](#) that installs the OMS agent, and enrolls the VM into an existing OMS workspace.
- [Log Analytics](#) is a service in OMS that monitors your cloud and on-premises environments to maintain their availability and performance. It collects data generated by resources in your cloud and on-premises environments and from other monitoring tools to provide analysis across multiple sources.

For Windows and Linux VMs, the recommended method for collecting logs and metrics is by installing the Log Analytics agent. The easiest way to install the Log Analytics agent on a VM is through the [Log Analytics VM Extension](#). Using the extension simplifies the installation process and automatically configures the agent to send data to the Log Analytics workspace that you specify. The agent is also upgraded automatically, ensuring that you have the latest features and fixes.

- [Network Watcher](#) enables you to monitor your VM and its associated resources as they relate to the network that they are in. You can install the Network Watcher Agent extension on a [Linux VM](#) or a [Windows VM](#).

Next steps

- Walk through the steps in [Monitor a Windows Virtual Machine with Azure PowerShell](#) or [Monitor a Linux Virtual Machine with the Azure CLI](#).
- Learn more about the best practices around [Monitoring and diagnostics](#).

Backup and restore options for virtual machines in Azure

9/14/2017 • 1 min to read • [Edit Online](#)

You can protect your data by taking backups at regular intervals. There are several backup options available for VMs, depending on your use-case.

Azure Backup

For backing up Azure VMs running production workloads, use Azure Backup. Azure Backup supports application-consistent backups for both Windows and Linux VMs. Azure Backup creates recovery points that are stored in geo-redundant recovery vaults. When you restore from a recovery point, you can restore the whole VM or just specific files.

For a simple, hands-on introduction to Azure Backup for Azure VMs, see the "Back up Windows virtual machines tutorial" for [Linux](#) or [Windows](#)

For more information on how Azure Backup works, see [Plan your VM backup infrastructure in Azure](#)

Azure Site Recovery

Azure Site Recovery protects your VMs from a major disaster scenario, when a whole region experiences an outage due to major natural disaster or widespread service interruption. You can configure Azure Site Recovery for your VMs so that you can recover your application with a single click in matter of minutes. You can replicate to an Azure region of your choice, it is not restricted to paired regions.

You can run disaster-recovery drills with on-demand test failovers, without affecting your production workloads or ongoing replication. Create recovery plans to orchestrate failover and failback of the entire application running on multiple VMs. The recovery plan feature is integrated with Azure automation runbooks.

You can get started by [replicating your virtual machines](#).

Managed snapshots

In development and test environments, snapshots provide a quick and simple option for backing up VMs that use Managed Disks. A managed snapshot is a read-only full copy of a managed disk. Snapshots exist independent of the source disk and can be used to create new managed disks for rebuilding a VM. They are billed based on the used portion of the disk. For example, if you create a snapshot of a managed disk with provisioned capacity of 64 GB and actual used data size of 10 GB, snapshot will be billed only for the used data size of 10 GB.

For more information on creating snapshots, see:

- [Create copy of VHD stored as a Managed Disk using Snapshots in Windows](#)
- [Create copy of VHD stored as a Managed Disk using Snapshots in Linux](#)

Next steps

You can try out Azure Backup by following the "Back up Windows virtual machines tutorial" for [Linux](#) or [Windows](#).

HPC, Batch, and Big Compute solutions using Azure VMs

9/14/2017 • 4 min to read • [Edit Online](#)

Organizations have large-scale computing needs. These Big Compute workloads include engineering design and analysis, financial risk calculations, image rendering, complex modeling, Monte Carlo simulations, and more.

Use the Azure cloud to efficiently run compute-intensive Linux and Windows workloads, from parallel batch jobs to traditional HPC simulations. Run your HPC and batch workloads on Azure infrastructure, with your choice of compute services, grid managers, Marketplace solutions, and vendor-hosted (SaaS) applications. Azure provides flexible solutions to distribute work and scale to thousands of VMs or cores and then scale down when you need fewer resources.

Solution options

- **Do-it-yourself solutions**

- Set up your own cluster environment in Azure virtual machines or [virtual machine scale sets](#).
- Lift and shift an on-premises cluster, or deploy a new cluster in Azure for additional capacity.
- Use Azure Resource Manager templates to deploy leading [workload managers](#), infrastructure, and [applications](#).
- Choose [HPC and GPU VM sizes](#) that include specialized hardware and network connections for MPI or GPU workloads.
- Add [high performance storage](#) for I/O-intensive workloads.

- **Hybrid solutions**

- Extend your on-premises solution to offload ("burst") peak workloads to Azure infrastructure
- Use cloud compute on-demand with your existing [workload manager](#).
- Take advantage of [HPC and GPU VM sizes](#) for MPI or GPU workloads.

- **Big Compute solutions as a service**

- Develop custom Big Compute solutions and workflows using [Azure Batch](#) and related [Azure services](#).
- Run Azure-enabled engineering and simulation solutions from vendors including [Altair](#), [Rescale](#), and [Cycle Computing](#) (now [joined with Microsoft](#)).

- **Marketplace solutions**

- Use the scale of [HPC applications](#) and [solutions](#) offered in the [Azure Marketplace](#).

The following sections provide more information about the supporting technologies and links to guidance.

Marketplace solutions

Visit the [Azure Marketplace](#) for Linux and Windows VM images and solutions designed for HPC. Examples include:

- [RogueWave CentOS-based HPC](#)
- [SUSE Linux Enterprise Server for HPC](#)
- [TIBCO Grid Server Engine](#)
- [Azure Data Science VM for Windows and Linux](#)
- [D3View](#)
- [UberCloud](#)
- [Intel Cloud Edition for Lustre](#)

HPC applications

Run custom or commercial HPC applications in Azure. Several examples in this section are benchmarked to scale efficiently with additional VMs or compute cores. Visit the [Azure Marketplace](#) for ready-to-deploy solutions.

NOTE

Check with the vendor of any commercial application for licensing or other restrictions for running in the cloud. Not all vendors offer pay-as-you-go licensing. You might need a licensing server in the cloud for your solution, or connect to an on-premises license server.

Engineering applications

- [Altair RADIOSS](#)
- [ANSYS CFD](#)
- [MATLAB Distributed Computing Server](#)
- [StarCCM+](#)
- [OpenFOAM](#)

Graphics and rendering

- [Autodesk Maya, 3ds Max, and Arnold](#) on Azure Batch (preview)

AI and deep learning

- [Batch AI](#) training for deep learning models
- [Microsoft Cognitive Toolkit](#)
- [Deep Learning VM](#)
- [Batch Shipyard recipes for deep learning](#)

HPC and GPU VM sizes

Azure offers a range of sizes for [Linux](#) and [Windows](#) VMs, including sizes designed for compute-intensive workloads. For example, H16r and H16mr VMs can connect to a high throughput back-end RDMA network. This cloud network can improve the performance of tightly coupled parallel applications running under [Microsoft MPI](#) or Intel MPI.

N-series VMs feature NVIDIA GPUs designed for compute-intensive or graphics-intensive applications including artificial intelligence (AI) learning and visualization.

Learn more:

- High performance compute sizes for [Linux](#) and [Windows](#) VMs
- GPU-enabled sizes for [Linux](#) and [Windows](#) VMs

Learn how to:

- [Set up a Linux RDMA cluster to run MPI applications](#)
- [Set up a Windows RDMA cluster with Microsoft HPC Pack to run MPI applications](#)
- [Use compute-intensive VMs in Batch pools](#)

Azure Batch

[Batch](#) is a platform service for running large-scale parallel and high-performance computing (HPC) applications efficiently in the cloud. Azure Batch schedules compute-intensive work to run on a managed pool of virtual machines, and can automatically scale compute resources to meet the needs of your jobs.

SaaS providers or developers can use the Batch SDKs and tools to integrate HPC applications or container workloads with Azure, stage data to Azure, and build job execution pipelines.

Learn how to:

- [Get started developing with Batch](#)
- [Use Azure Batch code samples](#)
- [Use low-priority VMs with Batch](#)
- [Run containerized HPC workloads with Batch Shipyard](#)
- [Use the R language with Batch](#)

Workload managers

The following are examples of cluster and workload managers that can run in Azure infrastructure. Create stand-alone clusters in Azure VMs or burst to Azure VMs from an on-premises cluster.

- [TIBCO DataSynapse GridServer](#)
- [Bright Cluster Manager](#)
- [IBM Spectrum Symphony and Symphony LSF](#)
- [PBS Pro](#)
- [Microsoft HPC Pack](#) - see options to run in [Windows](#) and [Linux](#) VMs

HPC storage

Large-scale Batch and HPC workloads have demands for data storage and access that exceed the capabilities of traditional cloud file systems. Implement parallel file system solutions in Azure such as [Lustre](#) and [BeeGFS](#).

Learn more:

- [Parallel file systems for HPC storage on Azure](#)

Related Azure services

Azure virtual machines, virtual machine scale sets, Batch, and related compute services are the foundation of most Azure HPC solutions. However, your solution can take advantage of many related Azure services. Here is a partial list:

Storage

- [Blob, table, and queue storage](#)
- [File storage](#)

Data and analytics

- [HDInsight](#) for Hadoop clusters on Azure
- [Data Factory](#)
- [Data Lake Store](#)
- [Machine Learning](#)
- [SQL Database](#)

Networking

- [Virtual Network](#)
- [ExpressRoute](#)

Containers

- [Container Service](#)

- [Container Registry](#)

Customer stories

Here are examples of customers that have solved business problems with Azure HPC solutions:

- [ANEO](#)
- [AXA Global P&C](#)
- [Axioma](#)
- [d3View](#)
- [Hymans Robertson](#)
- [MetLife](#)
- [Microsoft Research](#)
- [Milliman](#)
- [Mitsubishi UFJ Securities International](#)
- [Schlumberger](#)
- [Towers Watson](#)

Next steps

- Learn more about Big Compute solutions for [engineering simulation](#), [rendering](#), [banking](#) and [capital markets](#), and [genomics](#).
- For the latest announcements, see the [Microsoft HPC and Batch team blog](#) and the [Azure blog](#).
- Use the managed and scalable Azure [Batch](#) service to run compute-intensive workloads, without managing underlying infrastructure [Learn more](#)

Example Azure infrastructure walkthrough for Windows VMs

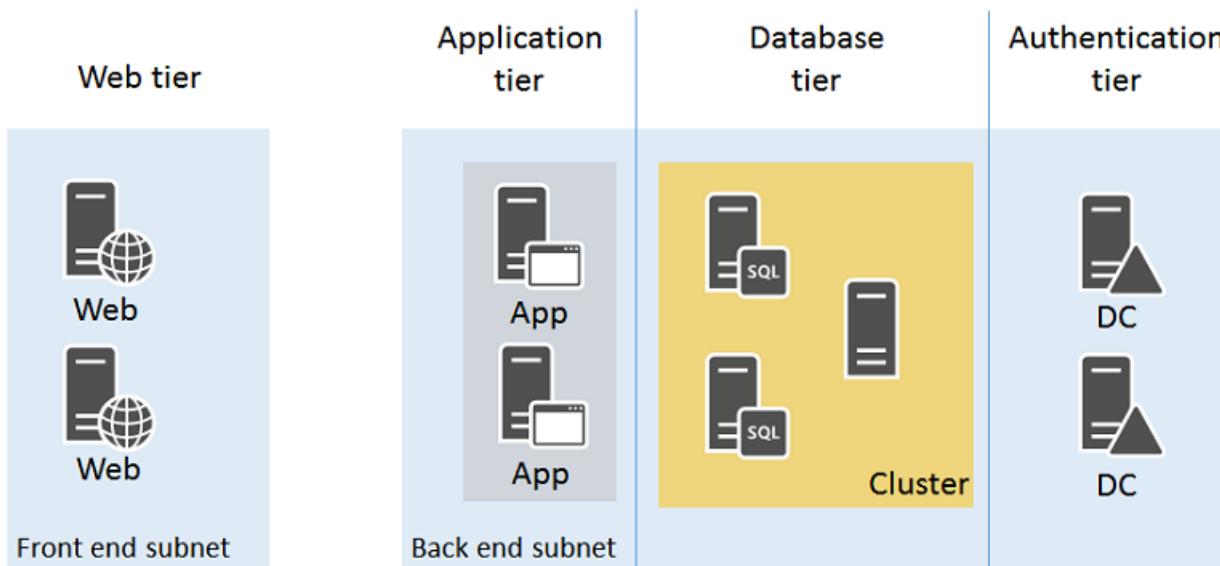
12/15/2017 • 3 min to read • [Edit Online](#)

This article walks through building out an example application infrastructure. We detail designing an infrastructure for a simple online store that brings together all the guidelines and decisions around naming conventions, availability sets, virtual networks and load balancers, and actually deploying your virtual machines (VMs).

Example workload

Adventure Works Cycles wants to build an online store application in Azure that consists of:

- Two IIS servers running the client front-end in a web tier
- Two IIS servers processing data and orders in an application tier
- Two Microsoft SQL Server instances with AlwaysOn availability groups (two SQL Servers and a majority node witness) for storing product data and orders in a database tier
- Two Active Directory domain controllers for customer accounts and suppliers in an authentication tier
- All the servers are located in two subnets:
 - a front-end subnet for the web servers
 - a back-end subnet for the application servers, SQL cluster, and domain controllers



Incoming secure web traffic must be load-balanced among the web servers as customers browse the online store. Order processing traffic in the form of HTTP requests from the web servers must be balanced among the application servers. Additionally, the infrastructure must be designed for high availability.

The resulting design must incorporate:

- An Azure subscription and account
- A single resource group
- Azure Managed Disks
- A virtual network with two subnets
- Availability sets for the VMs with a similar role
- Virtual machines

All the above follow these naming conventions:

- Adventure Works Cycles uses **[IT workload]-[location]-[Azure resource]** as a prefix
 - For this example, "azos" (Azure Online Store) is the IT workload name and "use" (East US 2) is the location
- Virtual networks use AZOS-USE-VN**[number]**
- Availability sets use azos-use-as-**[role]**
- Virtual machine names use azos-use-vm-**[vmname]**

Azure subscriptions and accounts

Adventure Works Cycles is using their Enterprise subscription, named Adventure Works Enterprise Subscription, to provide billing for this IT workload.

Storage

Adventure Works Cycles determined that they should use Azure Managed Disks. When creating VMs, both storage available storage tiers are used:

- **Standard storage** for the web servers, application servers, and domain controllers and their data disks.
- **Premium storage** for the SQL Server VMs and their data disks.

Virtual network and subnets

Because the virtual network does not need ongoing connectivity to the Adventure Work Cycles on-premises network, they decided on a cloud-only virtual network.

They created a cloud-only virtual network with the following settings using the Azure portal:

- Name: AZOS-USE-VN01
- Location: East US 2
- Virtual network address space: 10.0.0.0/8
- First subnet:
 - Name: FrontEnd
 - Address space: 10.0.1.0/24
- Second subnet:
 - Name: BackEnd
 - Address space: 10.0.2.0/24

Availability sets

To maintain high availability of all four tiers of their online store, Adventure Works Cycles decided on four availability sets:

- **azos-use-as-web** for the web servers
- **azos-use-as-app** for the application servers
- **azos-use-as-sql** for the SQL Servers
- **azos-use-as-dc** for the domain controllers

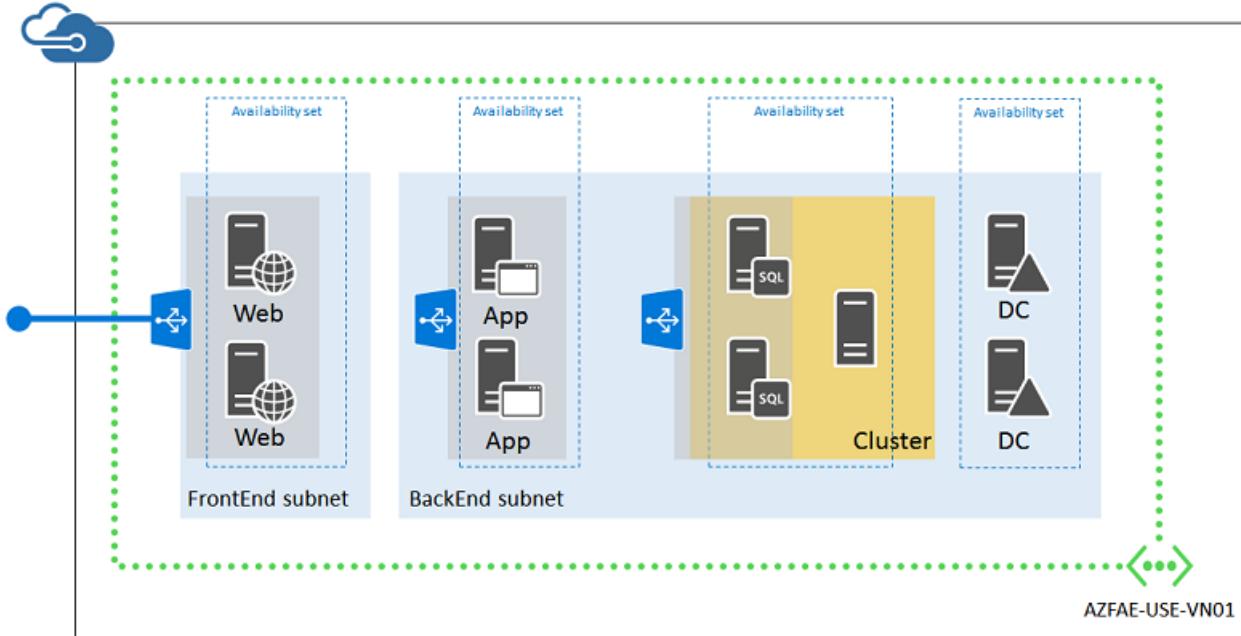
Virtual machines

Adventure Works Cycles decided on the following names for their Azure VMs:

- **azos-use-vm-web01** for the first web server

- **azos-use-vm-web02** for the second web server
- **azos-use-vm-app01** for the first application server
- **azos-use-vm-app02** for the second application server
- **azos-use-vm-sql01** for the first SQL Server server in the cluster
- **azos-use-vm-sql02** for the second SQL Server server in the cluster
- **azos-use-vm-dc01** for the first domain controller
- **azos-use-vm-dc02** for the second domain controller

Here is the resulting configuration.



This configuration incorporates:

- A cloud-only virtual network with two subnets (FrontEnd and BackEnd)
- Azure Managed Disks with both Standard and Premium disks
- Four availability sets, one for each tier of the online store
- The virtual machines for the four tiers
- An external load balanced set for HTTPS-based web traffic from the Internet to the web servers
- An internal load balanced set for unencrypted web traffic from the web servers to the application servers
- A single resource group

Virtual machine vCPU quotas

12/11/2017 • 2 min to read • [Edit Online](#)

The vCPU quotas for virtual machines and virtual machine scale sets are arranged in two tiers for each subscription, in each region. The first tier is the Total Regional vCPUs, and the second tier is the various VM size family cores such as Standard D Family vCPUs. Any time a new VM is deployed the vCPUs for the newly deployed VM must not exceed the vCPU quota for the specific VM size family or the total regional vCPU quota. If either of those quotas are exceeded, then the VM deployment will not be allowed. There is also a quota for the overall number of virtual machines in the region. The details on each of these quotas can be seen in the **Usage + quotas** section of the **Subscription** page in the [Azure portal](#), or you can query for the values using PowerShell.

Check usage

You can use the [Get-AzureRmVMUsage](#) cmdlet to check on your quota usage.

```
Get-AzureRmVMUsage -Location "East US"
```

The output will look similar to this:

Name	Current Value	Limit	Unit
Availability Sets	0	2000	Count
Total Regional vCPUs	4	260	Count
Virtual Machines	4	10000	Count
Virtual Machine Scale Sets	1	2000	Count
Standard B Family vCPUs	1	10	Count
Standard DSV2 Family vCPUs	1	100	Count
Standard Dv2 Family vCPUs	2	100	Count
Basic A Family vCPUs	0	100	Count
Standard A0-A7 Family vCPUs	0	250	Count
Standard A8-A11 Family vCPUs	0	100	Count
Standard D Family vCPUs	0	100	Count
Standard G Family vCPUs	0	100	Count
Standard DS Family vCPUs	0	100	Count
Standard GS Family vCPUs	0	100	Count
Standard F Family vCPUs	0	100	Count
Standard FS Family vCPUs	0	100	Count
Standard NV Family vCPUs	0	24	Count
Standard NC Family vCPUs	0	48	Count
Standard H Family vCPUs	0	8	Count
Standard Av2 Family vCPUs	0	100	Count
Standard LS Family vCPUs	0	100	Count
Standard Dv2 Promo Family vCPUs	0	100	Count
Standard DSV2 Promo Family vCPUs	0	100	Count
Standard MS Family vCPUs	0	0	Count
Standard Dv3 Family vCPUs	0	100	Count
Standard DSV3 Family vCPUs	0	100	Count
Standard Ev3 Family vCPUs	0	100	Count
Standard Esv3 Family vCPUs	0	100	Count
Standard FSv2 Family vCPUs	0	100	Count
Standard ND Family vCPUs	0	0	Count
Standard NCv2 Family vCPUs	0	0	Count
Standard NCv3 Family vCPUs	0	0	Count
Standard LSv2 Family vCPUs	0	0	Count
Standard Storage Managed Disks	2	10000	Count
Premium Storage Managed Disks	1	10000	Count

Reserved VM Instances

Reserved VM Instances, which are scoped to a single subscription, will add a new aspect to the vCPU quotas. These values describe the number of instances of the stated size that must be deployable in the subscription. They work as a placeholder in the quota system to ensure that quota is reserved to ensure reserved instances are deployable in the subscription. For example, if a specific subscription has 10 Standard_D1 reserved instances the usages limit for Standard_D1 Reserved Instances will be 10. This will cause Azure to ensure that there are always at least 10 vCPUs available in the Total Regional vCPUs quota to be used for Standard_D1 instances and there are at least 10 vCPUs available in the Standard D Family vCPU quota to be used for Standard_D1 instances.

If a quota increase is required to either purchase a Single Subscription RI, you can [request a quota increase](#) on your subscription.

Next steps

For more information about billing and quotas, see [Azure subscription and service limits, quotas, and constraints](#).

Create and manage Windows VMs in Azure using C#

7/17/2017 • 7 min to read • [Edit Online](#)

An [Azure Virtual Machine](#) (VM) needs several supporting Azure resources. This article covers creating, managing, and deleting VM resources using C#. You learn how to:

- Create a Visual Studio project
- Install the package
- Create credentials
- Create resources
- Perform management tasks
- Delete resources
- Run the application

It takes about 20 minutes to do these steps.

Create a Visual Studio project

1. If you haven't already, install [Visual Studio](#). Select **.NET desktop development** on the Workloads page, and then click **Install**. In the summary, you can see that **.NET Framework 4 - 4.6 development tools** is automatically selected for you. If you have already installed Visual Studio, you can add the .NET workload using the Visual Studio Launcher.
2. In Visual Studio, click **File > New > Project**.
3. In **Templates > Visual C#**, select **Console App (.NET Framework)**, enter *myDotnetProject* for the name of the project, select the location of the project, and then click **OK**.

Install the package

NuGet packages are the easiest way to install the libraries that you need to finish these steps. To get the libraries that you need in Visual Studio, do these steps:

1. Click **Tools > Nuget Package Manager**, and then click **Package Manager Console**.
2. Type this command in the console:

```
Install-Package Microsoft.Azure.Management.Fluent
```

Create credentials

Before you start this step, make sure that you have access to an [Active Directory service principal](#). You should also record the application ID, the authentication key, and the tenant ID that you need in a later step.

Create the authorization file

1. In Solution Explorer, right-click *myDotnetProject* > **Add > New Item**, and then select **Text File** in *Visual C# Items*. Name the file *azureauth.properties*, and then click **Add**.
2. Add these authorization properties:

```
subscription=<subscription-id>
client=<application-id>
key=<authentication-key>
tenant=<tenant-id>
managementURI=https://management.core.windows.net/
baseURL=https://management.azure.com/
authURL=https://login.windows.net/
graphURL=https://graph.windows.net/
```

Replace <**subscription-id**> with your subscription identifier, <**application-id**> with the Active Directory application identifier, <**authentication-key**> with the application key, and <**tenant-id**> with the tenant identifier.

3. Save the azureauth.properties file.
4. Set an environment variable in Windows named AZURE_AUTH_LOCATION with the full path to authorization file that you created. For example, the following PowerShell command can be used:

```
[Environment]::SetEnvironmentVariable("AZURE_AUTH_LOCATION", "C:\Visual Studio 2017\Projects\myDotnetProject\myDotnetProject\azureauth.properties", "User")
```

Create the management client

1. Open the Program.cs file for the project that you created, and then add these using statements to the existing statements at top of the file:

```
using Microsoft.Azure.Management.Compute.Fluent;
using Microsoft.Azure.Management.Compute.Fluent.Models;
using Microsoft.Azure.Management.Fluent;
using Microsoft.Azure.Management.ResourceManager.Fluent;
using Microsoft.Azure.Management.ResourceManager.Fluent.Core;
```

2. To create the management client, add this code to the Main method:

```
var credentials = SdkContext.AzureCredentialsFactory
    .FromFile(Environment.GetEnvironmentVariable("AZURE_AUTH_LOCATION"));

var azure = Azure
    .Configure()
    .WithLogLevel(HttpLoggingDelegatingHandler.Level.Basic)
    .Authenticate(credentials)
    .WithDefaultSubscription();
```

Create resources

Create the resource group

All resources must be contained in a [Resource group](#).

To specify values for the application and create the resource group, add this code to the Main method:

```
var groupName = "myResourceGroup";
var vmName = "myVM";
var location = Region.USWest;

Console.WriteLine("Creating resource group...");
var resourceGroup = azure.ResourceGroups.Define(groupName)
    .WithRegion(location)
    .Create();
```

Create the availability set

[Availability sets](#) make it easier for you to maintain the virtual machines used by your application.

To create the availability set, add this code to the Main method:

```
Console.WriteLine("Creating availability set...");
var availabilitySet = azure.AvailabilitySets.Define("myAVSet")
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithSku(AvailabilitySetSkuTypes.Managed)
    .Create();
```

Create the public IP address

A [Public IP address](#) is needed to communicate with the virtual machine.

To create the public IP address for the virtual machine, add this code to the Main method:

```
Console.WriteLine("Creating public IP address...");
var publicIPAddress = azure.PublicIPAddresses.Define("myPublicIP")
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithDynamicIP()
    .Create();
```

Create the virtual network

A virtual machine must be in a subnet of a [Virtual network](#).

To create a subnet and a virtual network, add this code to the Main method:

```
Console.WriteLine("Creating virtual network...");
var network = azure.Networks.Define("myVNet")
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithAddressSpace("10.0.0.0/16")
    .WithSubnet("mySubnet", "10.0.0.0/24")
    .Create();
```

Create the network interface

A virtual machine needs a network interface to communicate on the virtual network.

To create a network interface, add this code to the Main method:

```
Console.WriteLine("Creating network interface...");
var networkInterface = azure.NetworkInterfaces.Define("myNIC")
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithExistingPrimaryNetwork(network)
    .WithSubnet("mySubnet")
    .WithPrimaryPrivateIPAddressDynamic()
    .WithExistingPrimaryPublicIPAddress(publicIPAddress)
    .Create();
```

Create the virtual machine

Now that you created all the supporting resources, you can create a virtual machine.

To create the virtual machine, add this code to the Main method:

```
Console.WriteLine("Creating virtual machine...");
azure.VirtualMachines.Define(vmName)
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithExistingPrimaryNetworkInterface(networkInterface)
    .WithLatestWindowsImage("MicrosoftWindowsServer", "WindowsServer", "2012-R2-Datacenter")
    .WithAdminUsername("azureuser")
    .WithAdminPassword("Azure12345678")
    .WithComputerName(vmName)
    .WithExistingAvailabilitySet(availabilitySet)
    .WithSize(VirtualMachineSizeTypes.StandardDS1)
    .Create();
```

NOTE

This tutorial creates a virtual machine running a version of the Windows Server operating system. To learn more about selecting other images, see [Navigate and select Azure virtual machine images with Windows PowerShell and the Azure CLI](#).

If you want to use an existing disk instead of a marketplace image, use this code:

```
var managedDisk = azure.Disks.Define("myosdisk")
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithWindowsFromVhd("https://mystorage.blob.core.windows.net/vhds/myosdisk.vhd")
    .WithSizeInGB(128)
    .WithSku(DiskSkuTypes.PremiumLRS)
    .Create();

azure.VirtualMachines.Define("myVM")
    .WithRegion(location)
    .WithExistingResourceGroup(groupName)
    .WithExistingPrimaryNetworkInterface(networkInterface)
    .WithSpecializedOSDisk(managedDisk, OperatingSystemTypes.Windows)
    .WithExistingAvailabilitySet(availabilitySet)
    .WithSize(VirtualMachineSizeTypes.StandardDS1)
    .Create();
```

Perform management tasks

During the lifecycle of a virtual machine, you may want to run management tasks such as starting, stopping, or deleting a virtual machine. Additionally, you may want to create code to automate repetitive or complex tasks.

When you need to do anything with the VM, you need to get an instance of it:

```
var vm = azure.VirtualMachines.GetByResourceGroup(groupName, vmName);
```

Get information about the VM

To get information about the virtual machine, add this code to the Main method:

```

Console.WriteLine("Getting information about the virtual machine...");
Console.WriteLine("hardwareProfile");
Console.WriteLine("  vmSize: " + vm.Size);
Console.WriteLine("storageProfile");
Console.WriteLine("  imageReference");
Console.WriteLine("    publisher: " + vm.StorageProfile.ImageReference.Publisher);
Console.WriteLine("    offer: " + vm.StorageProfile.ImageReference.Offer);
Console.WriteLine("    sku: " + vm.StorageProfile.ImageReference.Sku);
Console.WriteLine("    version: " + vm.StorageProfile.ImageReference.Version);
Console.WriteLine("osDisk");
Console.WriteLine("  osType: " + vm.StorageProfile.OsDisk.OsType);
Console.WriteLine("  name: " + vm.StorageProfile.OsDisk.Name);
Console.WriteLine("  createOption: " + vm.StorageProfile.OsDisk.CreateOption);
Console.WriteLine("  caching: " + vm.StorageProfile.OsDisk.Caching);
Console.WriteLine("osProfile");
Console.WriteLine("  computerName: " + vm.OSProfile.ComputerName);
Console.WriteLine("  adminUsername: " + vm.OSProfile.AdminUsername);
Console.WriteLine("  provisionVMAgent: " + vm.OSProfile.WindowsConfiguration.ProvisionVMAgent.Value);
Console.WriteLine("  enableAutomaticUpdates: " +
vm.OSProfile.WindowsConfiguration.EnableAutomaticUpdates.Value);
Console.WriteLine("networkProfile");
foreach (string nicId in vm.NetworkInterfaceIds)
{
    Console.WriteLine("  networkInterface id: " + nicId);
}
Console.WriteLine("vmAgent");
Console.WriteLine("  vmAgentVersion" + vm.InstanceView.VmAgent.VmAgentVersion);
Console.WriteLine("  statuses");
foreach (InstanceViewStatus stat in vm.InstanceView.VmAgent.Statuses)
{
    Console.WriteLine("    code: " + stat.Code);
    Console.WriteLine("    level: " + stat.Level);
    Console.WriteLine("    displayStatus: " + stat.DisplayStatus);
    Console.WriteLine("    message: " + stat.Message);
    Console.WriteLine("    time: " + stat.Time);
}
Console.WriteLine("disks");
foreach (DiskInstanceView disk in vm.InstanceView.Disks)
{
    Console.WriteLine("  name: " + disk.Name);
    Console.WriteLine("  statuses");
    foreach (InstanceViewStatus stat in disk.Statuses)
    {
        Console.WriteLine("    code: " + stat.Code);
        Console.WriteLine("    level: " + stat.Level);
        Console.WriteLine("    displayStatus: " + stat.DisplayStatus);
        Console.WriteLine("    time: " + stat.Time);
    }
}
Console.WriteLine("VM general status");
Console.WriteLine("  provisioningStatus: " + vm.ProvisioningState);
Console.WriteLine("  id: " + vm.Id);
Console.WriteLine("  name: " + vm.Name);
Console.WriteLine("  type: " + vm.Type);
Console.WriteLine("  location: " + vm.Region);
Console.WriteLine("VM instance status");
foreach (InstanceViewStatus stat in vm.InstanceView.Statuses)
{
    Console.WriteLine("  code: " + stat.Code);
    Console.WriteLine("  level: " + stat.Level);
    Console.WriteLine("  displayStatus: " + stat.DisplayStatus);
}
Console.WriteLine("Press enter to continue...");
Console.ReadLine();

```

Stop the VM

You can stop a virtual machine and keep all its settings, but continue to be charged for it, or you can stop a virtual machine and deallocate it. When a virtual machine is deallocated, all resources associated with it are also deallocated and billing ends for it.

To stop the virtual machine without deallocating it, add this code to the Main method:

```
Console.WriteLine("Stopping vm...");
vm.PowerOff();
Console.WriteLine("Press enter to continue...");
Console.ReadLine();
```

If you want to deallocate the virtual machine, change the PowerOff call to this code:

```
vm.Deallocate();
```

Start the VM

To start the virtual machine, add this code to the Main method:

```
Console.WriteLine("Starting vm...");
vm.Start();
Console.WriteLine("Press enter to continue...");
Console.ReadLine();
```

Resize the VM

Many aspects of deployment should be considered when deciding on a size for your virtual machine. For more information, see [VM sizes](#).

To change size of the virtual machine, add this code to the Main method:

```
Console.WriteLine("Resizing vm...");
vm.Update()
    .WithSize(VirtualMachineSizeTypes.StandardDS2)
    .Apply();
Console.WriteLine("Press enter to continue...");
Console.ReadLine();
```

Add a data disk to the VM

To add a data disk to the virtual machine, add this code to the Main method to add a data disk that is 2 GB in size, has a LUN of 0 and a caching type of ReadWrite:

```
Console.WriteLine("Adding data disk to vm...");
vm.Update()
    .WithNewDataDisk(2, 0, CachingTypes.ReadWrite)
    .Apply();
Console.WriteLine("Press enter to delete resources...");
Console.ReadLine();
```

Delete resources

Because you are charged for resources used in Azure, it is always good practice to delete resources that are no longer needed. If you want to delete the virtual machines and all the supporting resources, all you have to do is delete the resource group.

To delete the resource group, add this code to the Main method:

```
azure.ResourceGroups.DeleteByName(groupName);
```

Run the application

It should take about five minutes for this console application to run completely from start to finish.

1. To run the console application, click **Start**.
2. Before you press **Enter** to start deleting resources, you could take a few minutes to verify the creation of the resources in the Azure portal. Click the deployment status to see information about the deployment.

Next steps

- Take advantage of using a template to create a virtual machine by using the information in [Deploy an Azure Virtual Machine using C# and a Resource Manager template](#).
- Learn more about using the [Azure libraries for .NET](#).

Create a Windows VM from a specialized disk

10/3/2017 • 7 min to read • [Edit Online](#)

Create a new VM by attaching a specialized managed disk as the OS disk using Powershell. A specialized disk is a copy of virtual hard disk (VHD) from an existing VM that maintains the user accounts, applications, and other state data from your original VM.

When you use a specialized VHD to create a new VM, the new VM retains the computer name of the original VM. Other computer-specific information is also kept and, in some cases, this duplicate information could cause issues. Be aware of what types of computer-specific information your applications rely on when copying a VM.

You have two options:

- [Upload a VHD](#)
- [Copy an existing Azure VM](#)

This topic shows you how to use managed disks. If you have a legacy deployment that requires using a storage account, see [Create a VM from a specialized VHD in a storage account](#)

Before you begin

If you use PowerShell, make sure that you have the latest version of the AzureRM.Compute PowerShell module.

```
Install-Module AzureRM.Compute -RequiredVersion 2.6.0
```

For more information, see [Azure PowerShell Versioning](#).

Option 1: Upload a specialized VHD

You can upload the VHD from a specialized VM created with an on-premises virtualization tool, like Hyper-V, or a VM exported from another cloud.

Prepare the VM

If you intend to use the VHD as-is to create a new VM, ensure the following steps are completed.

- [Prepare a Windows VHD to upload to Azure](#). **Do not** generalize the VM using Sysprep.
- Remove any guest virtualization tools and agents that are installed on the VM (like VMware tools).
- Ensure the VM is configured to pull its IP address and DNS settings via DHCP. This ensures that the server obtains an IP address within the VNet when it starts up.

Get the storage account

You need a storage account in Azure to store the uploaded VHD. You can either use an existing storage account or create a new one.

To show the available storage accounts, type:

```
Get-AzureRmStorageAccount
```

If you want to use an existing storage account, proceed to the [Upload the VHD](#) section.

If you need to create a storage account, follow these steps:

1. You need the name of the resource group where the storage account should be created. To find out all the resource groups that are in your subscription, type:

```
Get-AzureRmResourceGroup
```

To create a resource group named *myResourceGroup* in the *West US* region, type:

```
New-AzureRmResourceGroup -Name myResourceGroup -Location "West US"
```

2. Create a storage account named *mystorageaccount* in this resource group by using the [New-AzureRmStorageAccount](#) cmdlet:

```
New-AzureRmStorageAccount -ResourceGroupName myResourceGroup -Name mystorageaccount -Location "West US"  
-SkuName "Standard_LRS" -Kind "Storage"
```

Upload the VHD to your storage account

Use the [Add-AzureRmVhd](#) cmdlet to upload the VHD to a container in your storage account. This example uploads the file *myVHD.vhd* from `"C:\Users\Public\Documents\Virtual hard disks\"` to a storage account named *mystorageaccount* in the *myResourceGroup* resource group. The file is stored in the container named *mycontainer* and the new file name will be *myUploadedVHD.vhd*.

```
$resourceGroupName = "myResourceGroup"  
$urlOfUploadedVhd = "https://mystorageaccount.blob.core.windows.net/mycontainer/myUploadedVHD.vhd"  
Add-AzureRmVhd -ResourceGroupName $resourceGroupName -Destination $urlOfUploadedVhd  
-LocalFilePath "C:\Users\Public\Documents\Virtual hard disks\myVHD.vhd"
```

If successful, you get a response that looks similar to this:

```
MD5 hash is being calculated for the file C:\Users\Public\Documents\Virtual hard disks\myVHD.vhd.  
MD5 hash calculation is completed.  
Elapsed time for the operation: 00:03:35  
Creating new page blob of size 53687091712...  
Elapsed time for upload: 01:12:49  
  
LocalFilePath          DestinationUri  
-----              -----  
C:\Users\Public\Doc... https://mystorageaccount.blob.core.windows.net/mycontainer/myUploadedVHD.vhd
```

Depending on your network connection and the size of your VHD file, this command may take a while to complete.

Create a managed disk from the VHD

Create a managed disk from the specialized VHD in your storage account using [New-AzureRMDisk](#). This example uses **myOSDisk1** for the disk name, puts the disk in *StandardLRS* storage, and uses <https://storageaccount.blob.core.windows.net/vhdcontainer/osdisk.vhd> as the URI for the source VHD.

Create a new resource group for the new VM.

```
$destinationResourceGroup = 'myDestinationResourceGroup'  
New-AzureRmResourceGroup -Location $location -Name $destinationResourceGroup
```

Create the new OS disk from the uploaded VHD.

```
$sourceUri = (https://storageaccount.blob.core.windows.net/vhdcontainer/osdisk.vhd)
$osDiskName = 'myOsDisk'
$osDisk = New-AzureRmDisk -DiskName $osDiskName -Disk ` 
(New-AzureRmDiskConfig -AccountType StandardLRS -Location $location -CreateOption Import ` 
-SourceUri $sourceUri) ` 
-ResourceGroupName $destinationResourceGroup
```

Option 2: Copy an existing Azure VM

You can create a copy of a VM that uses managed disks by taking a snapshot of the VM, then using that snapshot to create a new managed disk and a new VM.

Take a snapshot of the OS disk

You can take a snapshot of an entire VM (including all disks) or of just a single disk. The following steps show you how to take a snapshot of just the OS disk of your VM using the [New-AzureRmSnapshot](#) cmdlet.

Set some parameters.

```
$resourceGroupName = 'myResourceGroup'
$vmName = 'myVM'
.setLocation = 'westus'
$snapshotName = 'mySnapshot'
```

Get the VM object.

```
$vm = Get-AzureRmVM -Name $vmName -ResourceGroupName $resourceGroupName
```

Get the OS disk name.

```
$disk = Get-AzureRmDisk -ResourceGroupName $resourceGroupName -DiskName $vm.StorageProfile.OsDisk.Name
```

Create the snapshot configuration.

```
$snapshotConfig = New-AzureRmSnapshotConfig -SourceUri $disk.Id -OsType Windows -CreateOption Copy -Location
.setLocation
```

Take the snapshot.

```
$snapShot = New-AzureRmSnapshot -Snapshot $snapshotConfig -SnapshotName $snapshotName -ResourceGroupName
$resourceGroupName
```

If you plan to use the snapshot to create a VM that needs to be high performing, use the parameter `-AccountType Premium_LRS` with the [New-AzureRmSnapshot](#) command. The parameter creates the snapshot so that it's stored as a Premium Managed Disk. Premium Managed Disks are more expensive than Standard. So be sure you really need Premium before using the parameter.

Create a new disk from the snapshot

Create a managed disk from the snapshot using [New-AzureRMDisk](#). This example uses *myOSDisk* for the disk name.

Create a new resource group for the new VM.

```
$destinationResourceGroup = 'myDestinationResourceGroup'  
New-AzureRmResourceGroup -Location $location -Name $destinationResourceGroup
```

Set the OS disk name.

```
$osDiskName = 'myOsDisk'
```

Create the managed disk.

```
$osDisk = New-AzureRmDisk -DiskName $osDiskName -Disk `  
    (New-AzureRmDiskConfig -Location $location -CreateOption Copy `  
        -SourceResourceId $snapshot.Id) `  
    -ResourceGroupName $destinationResourceGroup
```

Create the new VM

Create networking and other VM resources to be used by the new VM.

Create the subNet and vNet

Create the vNet and subNet of the [virtual network](#).

Create the subNet. This example creates a subnet named **mySubNet**, in the resource group **myDestinationResourceGroup**, and sets the subnet address prefix to **10.0.0.0/24**.

```
$subnetName = 'mySubNet'  
$singleSubnet = New-AzureRmVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix 10.0.0.0/24
```

Create the vNet. This example sets the virtual network name to be **myVnetName**, the location to **West US**, and the address prefix for the virtual network to **10.0.0.0/16**.

```
$vnetName = "myVnetName"  
$vnet = New-AzureRmVirtualNetwork -Name $vnetName -ResourceGroupName $destinationResourceGroup -Location  
    $location `  
    -AddressPrefix 10.0.0.0/16 -Subnet $singleSubnet
```

Create the network security group and an RDP rule

To be able to log in to your VM using RDP, you need to have a security rule that allows RDP access on port 3389. Because the VHD for the new VM was created from an existing specialized VM, you can use an account from the source virtual machine for RDP.

This example sets the NSG name to **myNsg** and the RDP rule name to **myRdpRule**.

```
$nsgName = "myNsg"  
  
$rdpRule = New-AzureRmNetworkSecurityRuleConfig -Name myRdpRule -Description "Allow RDP" `  
    -Access Allow -Protocol Tcp -Direction Inbound -Priority 110 `  
    -SourceAddressPrefix Internet -SourcePortRange * `  
    -DestinationAddressPrefix * -DestinationPortRange 3389  
$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $destinationResourceGroup -Location $location `  
    -Name $nsgName -SecurityRules $rdpRule
```

For more information about endpoints and NSG rules, see [Opening ports to a VM in Azure using PowerShell](#).

Create a public IP address and NIC

To enable communication with the virtual machine in the virtual network, you need a [public IP address](#) and a network interface.

Create the public IP. In this example, the public IP address name is set to **myIP**.

```
$ipName = "myIP"  
$pip = New-AzureRmPublicIpAddress -Name $ipName -ResourceGroupName $destinationResourceGroup -Location  
$location `  
-AllocationMethod Dynamic
```

Create the NIC. In this example, the NIC name is set to **myNicName**.

```
$nicName = "myNicName"  
$nic = New-AzureRmNetworkInterface -Name $nicName -ResourceGroupName $destinationResourceGroup `  
-Location $location -SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $pip.Id -NetworkSecurityGroupId  
$nsg.Id
```

Set the VM name and size

This example sets the VM name to *myVM* and the VM size to *Standard_A2*.

```
$vmName = "myVM"  
$vmConfig = New-AzureRmVMConfig -VMName $vmName -VMSize "Standard_A2"
```

Add the NIC

```
$vm = Add-AzureRmVMNetworkInterface -VM $vmConfig -Id $nic.Id
```

Add the OS disk

Add the OS disk to the configuration using [Set-AzureRmVMOSDisk](#). This example sets the size of the disk to *128 GB* and attaches the managed disk as a *Windows* OS disk.

```
$vm = Set-AzureRmVMOSDisk -VM $vm -ManagedDiskId $osDisk.Id -StorageAccountType StandardLRS `  
-DiskSizeInGB 128 -CreateOption Attach -Windows
```

Complete the VM

Create the VM using [New-AzureRMVM](#) the configurations that we just created.

```
New-AzureRmVM -ResourceGroupName $destinationResourceGroup -Location $location -VM $vm
```

If this command was successful, you'll see output like this:

RequestId	IsSuccessStatusCode	StatusCode	ReasonPhrase
-----	-----	-----	-----
True	OK	OK	

Verify that the VM was created

You should see the newly created VM either in the [Azure portal](#), under **Browse > Virtual machines**, or by using the following PowerShell commands:

```
$vmList = Get-AzureRmVM -ResourceGroupName $destinationResourceGroup  
$vmList.Name
```

Next steps

To sign in to your new virtual machine, browse to the VM in the [portal](#), click **Connect**, and open the Remote Desktop RDP file. Use the account credentials of your original virtual machine to sign in to your new virtual machine. For more information, see [How to connect and log on to an Azure virtual machine running Windows](#).

Deploy an Azure Virtual Machine using C# and a Resource Manager template

7/17/2017 • 5 min to read • [Edit Online](#)

This article shows you how to deploy an Azure Resource Manager template using C#. The template that you create deploys a single virtual machine running Windows Server in a new virtual network with a single subnet.

For a detailed description of the virtual machine resource, see [Virtual machines in an Azure Resource Manager template](#). For more information about all the resources in a template, see [Azure Resource Manager template walkthrough](#).

It takes about 10 minutes to do these steps.

Create a Visual Studio project

In this step, you make sure that Visual Studio is installed and you create a console application used to deploy the template.

1. If you haven't already, install [Visual Studio](#). Select **.NET desktop development** on the Workloads page, and then click **Install**. In the summary, you can see that **.NET Framework 4 - 4.6 development tools** is automatically selected for you. If you have already installed Visual Studio, you can add the .NET workload using the Visual Studio Launcher.
2. In Visual Studio, click **File > New > Project**.
3. In **Templates > Visual C#**, select **Console App (.NET Framework)**, enter *myDotnetProject* for the name of the project, select the location of the project, and then click **OK**.

Install the packages

NuGet packages are the easiest way to install the libraries that you need to finish these steps. To get the libraries that you need in Visual Studio, do these steps:

1. Click **Tools > Nuget Package Manager**, and then click **Package Manager Console**.
2. Type these commands in the console:

```
Install-Package Microsoft.Azure.Management.Fluent  
Install-Package WindowsAzure.Storage
```

Create the files

In this step, you create a template file that deploys the resources and a parameters file that supplies parameter values to the template. You also create an authorization file that is used to perform Azure Resource Manager operations.

Create the template file

1. In Solution Explorer, right-click *myDotnetProject* > **Add > New Item**, and then select **Text File** in *Visual C# Items*. Name the file *CreateVMTemplate.json*, and then click **Add**.
2. Add this JSON code to the file that you created:

```
{
```

```

"$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
"contentVersion": "1.0.0.0",
"parameters": {
    "adminUsername": { "type": "string" },
    "adminPassword": { "type": "securestring" }
},
"variables": {
    "vnetID": "[resourceId('Microsoft.Network/virtualNetworks', 'myVNet')]",
    "subnetRef": "[concat(variables('vnetID'), '/subnets/mySubnet')]"
},
"resources": [
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/publicIPAddresses",
    "name": "myPublicIPAddress",
    "location": "[resourceGroup().location]",
    "properties": {
        "publicIPAllocationMethod": "Dynamic",
        "dnsSettings": {
            "domainNameLabel": "myresourcegroupdns1"
        }
    }
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/virtualNetworks",
    "name": "myVNet",
    "location": "[resourceGroup().location]",
    "properties": {
        "addressSpace": { "addressPrefixes": [ "10.0.0.0/16" ] },
        "subnets": [
            {
                "name": "mySubnet",
                "properties": { "addressPrefix": "10.0.0.0/24" }
            }
        ]
    }
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/networkInterfaces",
    "name": "myNic",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Network/publicIPAddresses/', 'myPublicIPAddress')]",
        "[resourceId('Microsoft.Network/virtualNetworks/', 'myVNet')]"
    ],
    "properties": {
        "ipConfigurations": [
            {
                "name": "ipconfig1",
                "properties": {
                    "privateIPAllocationMethod": "Dynamic",
                    "publicIPAddress": { "id": "[resourceId('Microsoft.Network/publicIPAddresses', 'myPublicIPAddress')]",
                        "subnet": { "id": "[variables('subnetRef')]" }
                    }
                }
            }
        ]
    }
},
{
    "apiVersion": "2016-04-30-preview",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "myVM",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Network/networkInterfaces/', 'myNic')]"
    ],

```

```

"properties": {
    "hardwareProfile": { "vmSize": "Standard_DS1" },
    "osProfile": {
        "computerName": "myVM",
        "adminUsername": "[parameters('adminUsername')]",
        "adminPassword": "[parameters('adminPassword')]"
    },
    "storageProfile": {
        "imageReference": {
            "publisher": "MicrosoftWindowsServer",
            "offer": "WindowsServer",
            "sku": "2012-R2-Datacenter",
            "version": "latest"
        },
        "osDisk": {
            "name": "myManagedOSDisk",
            "caching": "ReadWrite",
            "createOption": "FromImage"
        }
    },
    "networkProfile": {
        "networkInterfaces": [
            {
                "id": "[resourceId('Microsoft.Network/networkInterfaces', 'myNic')]"
            }
        ]
    }
}
]
}

```

- Save the CreateVMTemplate.json file.

Create the parameters file

To specify values for the resource parameters that are defined in the template, you create a parameters file that contains the values.

- In Solution Explorer, right-click *myDotnetProject* > **Add** > **New Item**, and then select **Text File** in *Visual C# Items*. Name the file *Parameters.json*, and then click **Add**.
- Add this JSON code to the file that you created:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "adminUserName": { "value": "azureuser" },
        "adminPassword": { "value": "Azure12345678" }
    }
}
```

- Save the Parameters.json file.

Create the authorization file

Before you can deploy a template, make sure that you have access to an [Active Directory service principal](#). From the service principal, you acquire a token for authenticating requests to Azure Resource Manager. You should also record the application ID, the authentication key, and the tenant ID that you need in the authorization file.

- In Solution Explorer, right-click *myDotnetProject* > **Add** > **New Item**, and then select **Text File** in *Visual C# Items*. Name the file *azureauth.properties*, and then click **Add**.
- Add these authorization properties:

```
subscription=<subscription-id>
client=<application-id>
key=<authentication-key>
tenant=<tenant-id>
managementURI=https://management.core.windows.net/
baseURL=https://management.azure.com/
authURL=https://login.windows.net/
graphURL=https://graph.windows.net/
```

Replace <**subscription-id**> with your subscription identifier, <**application-id**> with the Active Directory application identifier, <**authentication-key**> with the application key, and <**tenant-id**> with the tenant identifier.

3. Save the azureauth.properties file.
4. Set an environment variable in Windows named AZURE_AUTH_LOCATION with the full path to authorization file that you created, for example the following PowerShell command can be used:

```
[Environment]::SetEnvironmentVariable("AZURE_AUTH_LOCATION", "C:\Visual Studio 2017\Projects\myDotnetProject\myDotnetProject\azureauth.properties", "User")
```

Create the management client

1. Open the Program.cs file for the project that you created, and then add these using statements to the existing statements at top of the file:

```
using Microsoft.Azure.Management.Compute.Fluent;
using Microsoft.Azure.Management.Compute.Fluent.Models;
using Microsoft.Azure.Management.Fluent;
using Microsoft.Azure.Management.ResourceManager.Fluent;
using Microsoft.Azure.Management.ResourceManager.Fluent.Core;
using Microsoft.WindowsAzure.Storage;
using Microsoft.WindowsAzure.Storage.Blob;
```

2. To create the management client, add this code to the Main method:

```
var credentials = SdkContext.AzureCredentialsFactory
    .FromFile(Environment.GetEnvironmentVariable("AZURE_AUTH_LOCATION"));

var azure = Azure
    .Configure()
    .WithLogLevel(HttpLoggingDelegatingHandler.Level.Basic)
    .Authenticate(credentials)
    .WithDefaultSubscription();
```

Create a resource group

To specify values for the application, add code to the Main method:

```
var groupName = "myResourceGroup";
var location = Region.USWest;

var resourceGroup = azure.ResourceGroups.Define(groupName)
    .WithRegion(location)
    .Create();
```

Create a storage account

The template and parameters are deployed from a storage account in Azure. In this step, you create the account and upload the files.

To create the account, add this code to the Main method:

```
string storageAccountName = SdkContext.RandomResourceName("st", 10);

Console.WriteLine("Creating storage account...");
var storage = azure.StorageAccounts.Define(storageAccountName)
    .WithRegion(Region.USWest)
    .WithExistingResourceGroup(resourceGroup)
    .Create();

var storageKeys = storage.GetKeys();
string storageConnectionString = "DefaultEndpointsProtocol=https;" +
    + "AccountName=" + storage.Name
    + ";AccountKey=" + storageKeys[0].Value
    + ";EndpointSuffix=core.windows.net";

var account = CloudStorageAccount.Parse(storageConnectionString);
var serviceClient = account.CreateCloudBlobClient();

Console.WriteLine("Creating container...");
var container = serviceClient.GetContainerReference("templates");
container.CreateIfNotExistsAsync().Wait();
var containerPermissions = new BlobContainerPermissions()
    { PublicAccess = BlobContainerPublicAccessType.Container };
container.SetPermissionsAsync(containerPermissions).Wait();

Console.WriteLine("Uploading template file...");
var templateblob = container.GetBlockBlobReference("CreateVMTemplate.json");
templateblob.UploadFromFile("../..\\..\\CreateVMTemplate.json");

Console.WriteLine("Uploading parameters file...");
var paramblob = container.GetBlockBlobReference("Parameters.json");
paramblob.UploadFromFile("../..\\..\\Parameters.json");
```

Deploy the template

Deploy the template and parameters from the storage account that was created.

To deploy the template, add this code to the Main method:

```
var templatePath = "https://" + storageAccountName + ".blob.core.windows.net/templates/CreateVMTemplate.json";
var paramPath = "https://" + storageAccountName + ".blob.core.windows.net/templates/Parameters.json";
var deployment = azure.Deployments.Define("myDeployment")
    .WithExistingResourceGroup(groupName)
    .WithTemplateLink(templatePath, "1.0.0.0")
    .WithParametersLink(paramPath, "1.0.0.0")
    .WithMode(Microsoft.Azure.Management.ResourceManager.Fluent.Models.DeploymentMode.Incremental)
    .Create();
Console.WriteLine("Press enter to delete the resource group...");
Console.ReadLine();
```

Delete the resources

Because you are charged for resources used in Azure, it is always good practice to delete resources that are no longer needed. You don't need to delete each resource separately from a resource group. Delete the resource group and all its resources are automatically deleted.

To delete the resource group, add this code to the Main method:

```
azure.ResourceGroups.DeleteByName(groupName);
```

Run the application

It should take about five minutes for this console application to run completely from start to finish.

1. To run the console application, click **Start**.
2. Before you press **Enter** to start deleting resources, you could take a few minutes to verify the creation of the resources in the Azure portal. Click the deployment status to see information about the deployment.

Next steps

- If there were issues with the deployment, a next step would be to look at [Troubleshoot common Azure deployment errors with Azure Resource Manager](#).
- Learn how to deploy a virtual machine and its supporting resources by reviewing [Deploy an Azure Virtual Machine Using C#](#).

Automating Azure virtual machine deployment with Chef

12/11/2017 • 6 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

Chef is a great tool for delivering automation and desired state configurations.

With the latest cloud api release, Chef provides seamless integration with Azure, giving you the ability to provision and deploy configuration states through a single command.

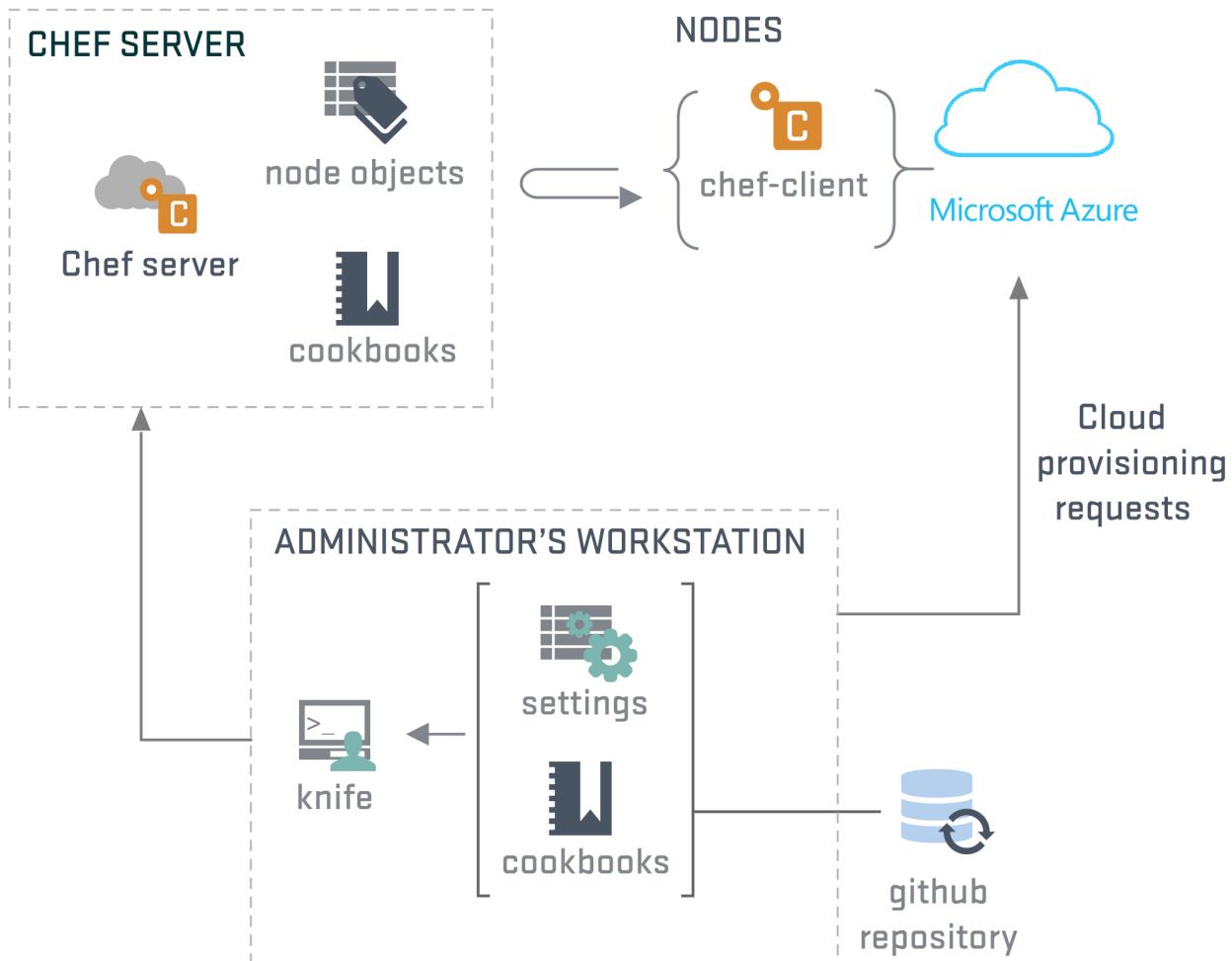
In this article, you set up your Chef environment to provision Azure virtual machines and walk through creating a policy or "CookBook" and then deploying this cookbook to an Azure virtual machine.

Let's begin!

Chef basics

Before you begin, [review the basic concepts of Chef](#).

The following diagram depicts the high-level Chef architecture.



Chef has three main architectural components: Chef Server, Chef Client (node), and Chef Workstation.

The Chef Server is the management point and there are two options for the Chef Server: a hosted solution or an on-premises solution. We will be using a hosted solution.

The Chef Client (node) is the agent that sits on the servers you are managing.

The Chef Workstation is the admin workstation where we create policies and execute management commands. We run the **knife** command from the Chef Workstation to manage the infrastructure.

There is also the concept of "Cookbooks" and "Recipes". These are effectively the policies we define and apply to the servers.

Preparing the workstation

First, let's prep the workstation. I'm using a standard Windows workstation. We need to create a directory to store the config files and cookbooks.

First create a directory called C:\chef.

Then create a second directory called c:\chef\cookbooks.

We now need to download the Azure settings file so Chef can communicate with the Azure subscription.

Download your publish settings using the PowerShell Azure [Get-AzurePublishSettingsFile](#) command.

Save the publish settings file in C:\chef.

Creating a managed Chef account

Sign up for a hosted Chef account [here](#).

During the signup process, you will be asked to create a new organization.

Create Organization

Full Name (example: Chef, Inc.)

Short Name (example: chef)

Short name is required

[Cancel](#) [Create Organization](#)

Once your organization is created, download the starter kit.

CHEF
MANAGE

Nodes Reports Policy Administration

➤ Organizations

- Create
- Reset Validation Key
- Generate Knife Config
- Invite User
- Leave Organization
- Starter Kit

Users

Groups

Global Permissions

Thank you for choosing hosted Chef!

Follow these steps to be on your way to using hosted Chef

[Download Starter Kit](#) [Learn Chef](#)

What's next?

[Chef Documentation](#) [Browse Community Cookbooks](#)

The best place to start learning about Chef in general.

Hundreds of members of the Chef community have contributed cookbooks you can use or draw inspiration from.

NOTE

If you receive a prompt warning you that your keys will be reset, it's ok to proceed as we have no existing infrastructure configured as yet.

This starter kit zip file contains your organization config files and keys.

Configuring the Chef workstation

Extract the content of the chef-starter.zip to C:\chef.

Copy all files under chef-starter\chef-repo.chef to your c:\chef directory.

Your directory should now look something like the following example.

Name	Date modified	Type
chef-starter	11/12/2014 11:29 A...	File folder
cookbooks	11/12/2014 7:12 AM	File folder
a[REDACTED].validator.pem	11/12/2014 11:29 A...	PEM File
diego.publishsettings	11/12/2014 7:46 PM	PUBLISHSETTINGS F...
[REDACTED].pem	11/12/2014 11:29 A...	PEM File
knife.rb	11/12/2014 7:54 PM	RB File

You should now have four files including the Azure publishing file in the root of c:\chef.

The PEM files contain your organization and admin private keys for communication while the knife.rb file contains your knife configuration. We will need to edit the knife.rb file.

Open the file in your editor of choice and modify the "cookbook_path" by removing the ./ from the path so it appears as shown next.

```
cookbook_path ["#{current_dir}/cookbooks"]
```

Also add the following line reflecting the name of your Azure publish settings file.

```
knife[:azure_publish_settings_file] = "yourfilename.publishsettings"
```

Your knife.rb file should now look similar to the following example.

```
current_dir = File.dirname(__FILE__)
log_level           :info
log_location        STDOUT
node_name          [REDACTED]
client_key         "#{current_dir}/[REDACTED].pem"
validation_client_name "a[REDACTED]-validator"
validation_key     "#{current_dir}/a[REDACTED]-validator.pem"
chef_server_url   "https://api.opscode.com/organizations/[REDACTED]"
cache_type         'BasicFile'
cache_options( :path => "#{ENV['HOME']}/.chef/checksums" )
cookbook_path      ["#{current_dir}/cookbooks"]
knife[:azure_publish_settings_file] = "[REDACTED].publishsettings"
```

These lines will ensure that Knife references the cookbooks directory under c:\chef\cookbooks, and also uses our Azure Publish Settings file during Azure operations.

Installing the Chef Development Kit

Next [download and install](#) the ChefDK (Chef Development Kit) to set up your Chef Workstation.

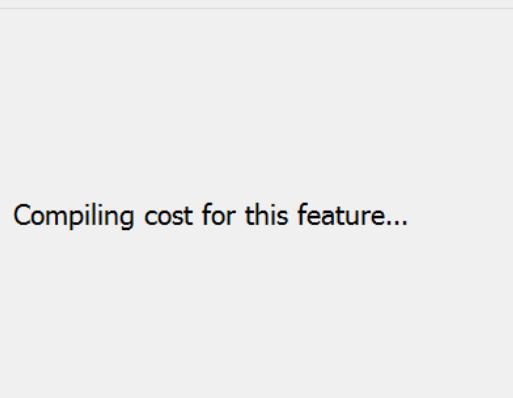


Custom Setup

Select the way you want features to be installed.



Click the icons in the tree below to change the way features will be installed.



Location: C:\opscode\

[Browse...](#)

[Reset](#)

[Disk Usage](#)

[Back](#)

[Next](#)

[Cancel](#)

Install in the default location of c:\opscode. This install will take around 10 minutes.

Confirm your PATH variable contains entries for

C:\opscode\chefdk\bin;C:\opscode\chefdk\embedded\bin;c:\users\yourusername.chefdk\gem\ruby\2.0.0\bin

If they are not there, make sure you add these paths!

NOTE THE ORDER OF THE PATH IS IMPORTANT! If your opscode paths are not in the correct order you will have issues.

Reboot your workstation before you continue.

Next, we will install the Knife Azure extension. This provides Knife with the "Azure Plugin".

Run the following command.

```
chef gem install knife-azure --pre
```

NOTE

The --pre argument ensures you are receiving the latest RC version of the Knife Azure Plugin which provides access to the latest set of APIs.

It's likely that a number of dependencies will also be installed at the same time.

C:\ Command Prompt

```
c:\Chef>chef gem install knife-azure --pre

Temporarily enhancing PATH to include DevKit...
Building native extensions. This could take a while...
Successfully installed eventmachine-1.0.4
Successfully installed em-winrm-0.6.0
Successfully installed winrm-s-0.2.2
Successfully installed knife-windows-0.8.2
Fetching: knife-azure-1.4.0.rc.0.gem (100%)
Successfully installed knife-azure-1.4.0.rc.0
Parsing documentation for eventmachine-1.0.4
Installing ri documentation for eventmachine-1.0.4
Parsing documentation for em-winrm-0.6.0
Installing ri documentation for em-winrm-0.6.0
Parsing documentation for winrm-s-0.2.2
Installing ri documentation for winrm-s-0.2.2
Parsing documentation for knife-windows-0.8.2
Installing ri documentation for knife-windows-0.8.2
Parsing documentation for knife-azure-1.4.0.rc.0
Installing ri documentation for knife-azure-1.4.0.rc.0
Done installing documentation for eventmachine, em-winrm, winrm-s, knife-windows, knife-azure after 16 seconds
5 gems installed
```

To ensure everything is configured correctly, run the following command.

```
knife azure image list
```

If everything is configured correctly, you will see a list of available Azure images scroll through.

Congratulations. The workstation is set up!

Creating a Cookbook

A Cookbook is used by Chef to define a set of commands that you wish to execute on your managed client. Creating a Cookbook is straightforward and we use the **chef generate cookbook** command to generate the Cookbook template. I will be calling my Cookbook web server as I would like a policy that automatically deploys IIS.

Under your C:\Chef directory run the following command.

```
chef generate cookbook webserver
```

This will generate a set of files under the directory C:\Chef\cookbooks\webserver. We now need to define the set of commands we would like the Chef client to execute on the managed virtual machine.

The commands are stored in the file default.rb. In this file, I'll be defining a set of commands that installs IIS, starts IIS and copies a template file to the wwwroot folder.

Modify the C:\chef\cookbooks\webserver\recipes\default.rb file and add the following lines.

```
powershell_script 'Install IIS' do
  action :run
  code 'add-windowsfeature Web-Server'
end

service 'w3svc' do
  action [ :enable, :start ]
end

template 'c:\inetpub\wwwroot\Default.htm' do
  source 'Default.htm.erb'
  rights :read, 'Everyone'
end
```

Save the file once you are done.

Creating a template

As we mentioned previously, we need to generate a template file which will be used as the default.html page.

Run the following command to generate the template.

```
chef generate template webserver Default.htm
```

Now navigate to the C:\chef\cookbooks\webserver\templates\default\Default.htm.erb file. Edit the file by adding some simple "Hello World" HTML code, and then save the file.

Upload the Cookbook to the Chef Server

In this step, we are taking a copy of the Cookbook that we have created on the local machine and uploading it to the Chef Hosted Server. Once uploaded, the Cookbook will appear under the **Policy** tab.

```
knife cookbook upload webserver
```

Nodes	Reports	Policy	Administration
Showing All Cookbooks			
Cookbook			Current Version
webserver			0.1.2

Deploy a virtual machine with Knife Azure

We will now deploy an Azure virtual machine and apply the "Webserver" Cookbook which will install the IIS web service and default web page.

In order to do this, use the **knife azure server create** command.

An example of the command appears next.

```
knife azure server create --azure-dns-name 'diegotest01' --azure-vm-name 'testserver01' --azure-vm-size 'Small'  
--azure-storage-account 'portalvhdsxxxx' --bootstrap-protocol 'cloud-api' --azure-source-image  
'a699494373c04fc0bc8f2bb1389d6106__Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd' --azure-service-  
location 'Southeast Asia' --winrm-user azureuser --winrm-password 'myPassword123' --tcp-endpoints 80,3389 --  
'recipe[webserver]'
```

The parameters are self-explanatory. Substitute your particular variables and run.

NOTE

Through the the command line, I'm also automating my endpoint network filter rules by using the -tcp-endpoints parameter. I've opened up ports 80 and 3389 to provide access to my web page and RDP session.

Once you run the command, go to the Azure portal and you will see your machine begin to provision.

testserver01 ** Starting Visual Studio Ultimate with MSDN Southeast Asia diegotest01.cloudapp.net

The command prompt appears next.

```

C:\ Command Prompt

c:\>Chef>knife azure server create --azure-dns-name 'diegotest01' --azure-vm-name 'testserver01' --azure-vm-size 'Small' --azure-storage-account 'portals' --bootstrap-protocol 'cloud-api' --azure-source-image 'a699494373c04fc0bc8f2bb1389d6106_Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd' --azure-service-location 'Southeast Asia' --winrm-user azureuser --winrm-password 'myPassword123' --tcp-endpoints 80,3389 --r 'recipie\webservice01'

Waiting for virtual machine to reach status 'provisioning' .....vm state 'provisioning' reached after 2.79 minutes.
Waiting for virtual machine to reach status 'ready' .....vm state 'ready' reached after 2.23 minutes.

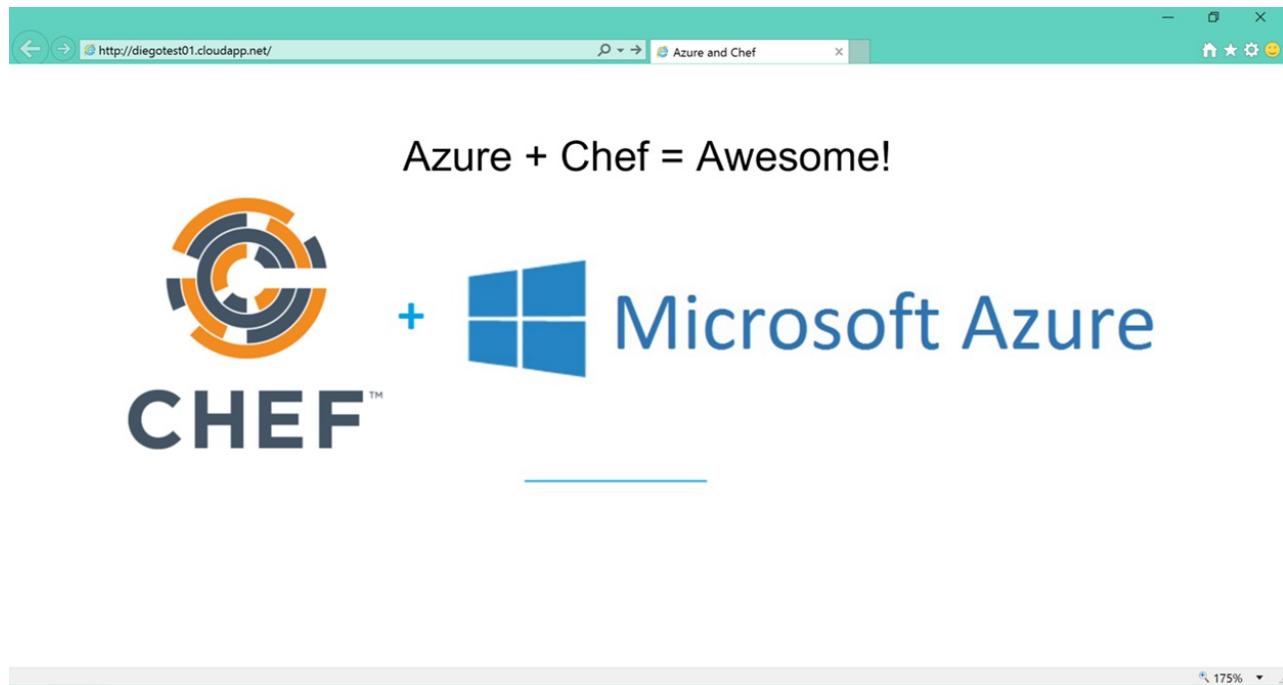
DNS Name: diegotest01.cloudapp.net
VM Name: testserver01
Size: Small
Azure Source Image: a699494373c04fc0bc8f2bb1389d6106_Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd
Azure Service Location: Southeast Asia
Public Ip Address: 104.43.9.88
Private Ip Address: 100.72.52.27
WinRM Port: 5985
TCP Ports: [{"Name": "tcpport_3389_testserver01", "Vip": "104.43.9.88", "PublicPort": "3389", "LocalPort": "3389"}, {"Name": "tcpport_80_testserver01", "Vip": "104.43.9.88", "PublicPort": "80", "LocalPort": "80"}]
Environment: _default
Runlist: ["recipie\webservice01"]

Waiting for Resource Extension to reach status 'wagent provisioning' ....Resource extension state 'wagent provisioning' reached after 0.09 minutes.
Waiting for Resource Extension to reach status 'provisioning' .....Resource extension state 'provisioning' reached after 2.02 minutes.
Waiting for Resource Extension to reach status 'ready' .....Resource extension state 'ready' reached after 4.86 minutes.

DNS Name: diegotest01.cloudapp.net
VM Name: testserver01
Size: Small
Azure Source Image: a699494373c04fc0bc8f2bb1389d6106_Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd
Azure Service Location: Southeast Asia
Public Ip Address: 104.43.9.88
Private Ip Address: 100.72.52.27
WinRM Port: 5985
TCP Ports: [{"Name": "tcpport_3389_testserver01", "Vip": "104.43.9.88", "PublicPort": "3389", "LocalPort": "3389"}, {"Name": "tcpport_80_testserver01", "Vip": "104.43.9.88", "PublicPort": "80", "LocalPort": "80"}]
Environment: _default
Runlist: ["recipie\webservice01"]

```

Once the deployment is complete, we should be able to connect to the web service over port 80 as we had opened the port when we provisioned the virtual machine with the Knife Azure command. As this virtual machine is the only virtual machine in my cloud service, I'll connect it with the cloud service url.



As you can see, I got creative with my HTML code.

Don't forget we can also connect through an RDP session from the Azure portal via port 3389.

I hope this has been helpful! Go and start your infrastructure as code journey with Azure today!

Create and manage Windows VMs in Azure using Java

12/7/2017 • 7 min to read • [Edit Online](#)

An [Azure Virtual Machine](#) (VM) needs several supporting Azure resources. This article covers creating, managing, and deleting VM resources using Java. You learn how to:

- Create a Maven project
- Add dependencies
- Create credentials
- Create resources
- Perform management tasks
- Delete resources
- Run the application

It takes about 20 minutes to do these steps.

Create a Maven project

1. If you haven't already done so, install [Java](#).
2. Install [Maven](#).
3. Create a new folder and the project:

```
mkdir java-azure-test
cd java-azure-test

mvn archetype:generate -DgroupId=com.fabrikam -DartifactId=testAzureApp -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

Add dependencies

1. Under the `testAzureApp` folder, open the `pom.xml` file and add the build configuration to `<project>` to enable the building of your application:

```
<build>
  <plugins>
    <plugin>
      <groupId>org.codehaus.mojo</groupId>
      <artifactId>exec-maven-plugin</artifactId>
      <configuration>
        <mainClass>com.fabrikam.testAzureApp.App</mainClass>
      </configuration>
    </plugin>
  </plugins>
</build>
```

2. Add the dependencies that are needed to access the Azure Java SDK.

```

<dependency>
    <groupId>com.microsoft.azure</groupId>
    <artifactId>azure</artifactId>
    <version>1.1.0</version>
</dependency>
<dependency>
    <groupId>com.microsoft.azure</groupId>
    <artifactId>azure-mgmt-compute</artifactId>
    <version>1.1.0</version>
</dependency>
<dependency>
    <groupId>com.microsoft.azure</groupId>
    <artifactId>azure-mgmt-resources</artifactId>
    <version>1.1.0</version>
</dependency>
<dependency>
    <groupId>com.microsoft.azure</groupId>
    <artifactId>azure-mgmt-network</artifactId>
    <version>1.1.0</version>
</dependency>
<dependency>
    <groupId>com.squareup.okio</groupId>
    <artifactId>okio</artifactId>
    <version>1.13.0</version>
</dependency>
<dependency>
    <groupId>com.nimbusds</groupId>
    <artifactId>nimbus-jose-jwt</artifactId>
    <version>3.6</version>
</dependency>
<dependency>
    <groupId>net.minidev</groupId>
    <artifactId>json-smart</artifactId>
    <version>1.0.6.3</version>
</dependency>
<dependency>
    <groupId>javax.mail</groupId>
    <artifactId>mail</artifactId>
    <version>1.4.5</version>
</dependency>

```

- Save the file.

Create credentials

Before you start this step, make sure that you have access to an [Active Directory service principal](#). You should also record the application ID, the authentication key, and the tenant ID that you need in a later step.

Create the authorization file

- Create a file named `azureauth.properties` and add these properties to it:

```

subscription=<subscription-id>
client=<application-id>
key=<authentication-key>
tenant=<tenant-id>
managementURI=https://management.core.windows.net/
baseURL=https://management.azure.com/
authURL=https://login.windows.net/
graphURL=https://graph.windows.net/

```

Replace **<subscription-id>** with your subscription identifier, **<application-id>** with the Active Directory application identifier, **<authentication-key>** with the application key, and **<tenant-id>** with the tenant

identifier.

2. Save the file.
3. Set an environment variable named AZURE_AUTH_LOCATION in your shell with the full path to the authentication file.

Create the management client

1. Open the `App.java` file under `src\main\java\com\fabrikam` and make sure this package statement is at the top:

```
package com.fabrikam.testAzureApp;
```

2. Under the package statement, add these import statements:

```
import com.microsoft.azure.management.Azure;
import com.microsoft.azure.management.compute.AvailabilitySet;
import com.microsoft.azure.management.compute.AvailabilitySetSkuTypes;
import com.microsoft.azure.management.compute.CachingTypes;
import com.microsoft.azure.management.compute.InstanceViewStatus;
import com.microsoft.azure.management.compute.DiskInstanceView;
import com.microsoft.azure.management.compute.VirtualMachine;
import com.microsoft.azure.management.compute.VirtualMachineSizeTypes;
import com.microsoft.azure.management.network.PublicIPAddress;
import com.microsoft.azure.management.network.Network;
import com.microsoft.azure.management.network.NetworkInterface;
import com.microsoft.azure.management.resources.ResourceGroup;
import com.microsoft.azure.management.resources.fluentcore.arm.Region;
import com.microsoft.azure.management.resources.fluentcore.model.Creatable;
import com.microsoft.rest.LogLevel;
import java.io.File;
import java.util.Scanner;
```

3. To create the Active Directory credentials that you need to make requests, add this code to the main method of the App class:

```
try {
    final File credFile = new File(System.getenv("AZURE_AUTH_LOCATION"));
    Azure azure = Azure.configure()
        .withLogLevel(LogLevel.BASIC)
        .authenticate(credFile)
        .withDefaultSubscription();
} catch (Exception e) {
    System.out.println(e.getMessage());
    e.printStackTrace();
}
```

Create resources

Create the resource group

All resources must be contained in a [Resource group](#).

To specify values for the application and create the resource group, add this code to the try block in the main method:

```
System.out.println("Creating resource group...");  
ResourceGroup resourceGroup = azure.resourceGroups()  
    .define("myResourceGroup")  
    .withRegion(Region.US_EAST)  
    .create();
```

Create the availability set

[Availability sets](#) make it easier for you to maintain the virtual machines used by your application.

To create the availability set, add this code to the try block in the main method:

```
System.out.println("Creating availability set...");  
AvailabilitySet availabilitySet = azure.availabilitySets()  
    .define("myAvailabilitySet")  
    .withRegion(Region.US_EAST)  
    .withExistingResourceGroup("myResourceGroup")  
    .withSku(AvailabilitySetSkuTypes.MANAGED)  
    .create();
```

Create the public IP address

A [Public IP address](#) is needed to communicate with the virtual machine.

To create the public IP address for the virtual machine, add this code to the try block in the main method:

```
System.out.println("Creating public IP address...");  
PublicIPAddress publicIPAddress = azure.publicIPAddresses()  
    .define("myPublicIP")  
    .withRegion(Region.US_EAST)  
    .withExistingResourceGroup("myResourceGroup")  
    .withDynamicIP()  
    .create();
```

Create the virtual network

A virtual machine must be in a subnet of a [Virtual network](#).

To create a subnet and a virtual network, add this code to the try block in the main method:

```
System.out.println("Creating virtual network...");  
Network network = azure.networks()  
    .define("myVN")  
    .withRegion(Region.US_EAST)  
    .withExistingResourceGroup("myResourceGroup")  
    .withAddressSpace("10.0.0.0/16")  
    .withSubnet("mySubnet", "10.0.0.0/24")  
    .create();
```

Create the network interface

A virtual machine needs a network interface to communicate on the virtual network.

To create a network interface, add this code to the try block in the main method:

```

System.out.println("Creating network interface...");
NetworkInterface networkInterface = azure.networkInterfaces()
    .define("myNIC")
    .withRegion(Region.US_EAST)
    .withExistingResourceGroup("myResourceGroup")
    .withExistingPrimaryNetwork(network)
    .withSubnet("mySubnet")
    .withPrimaryPrivateIPAddressDynamic()
    .withExistingPrimaryPublicIPAddress(publicIPAddress)
    .create();

```

Create the virtual machine

Now that you created all the supporting resources, you can create a virtual machine.

To create the virtual machine, add this code to the try block in the main method:

```

System.out.println("Creating virtual machine...");
VirtualMachine virtualMachine = azure.virtualMachines()
    .define("myVM")
    .withRegion(Region.US_EAST)
    .withExistingResourceGroup("myResourceGroup")
    .withExistingPrimaryNetworkInterface(networkInterface)
    .withLatestWindowsImage("MicrosoftWindowsServer", "WindowsServer", "2012-R2-Datacenter")
    .withAdminUsername("azureuser")
    .withAdminPassword("Azure12345678")
    .withComputerName("myVM")
    .withExistingAvailabilitySet(availabilitySet)
    .withSize("Standard_DS1")
    .create();

Scanner input = new Scanner(System.in);
System.out.println("Press enter to get information about the VM...");
input.nextLine();

```

NOTE

This tutorial creates a virtual machine running a version of the Windows Server operating system. To learn more about selecting other images, see [Navigate and select Azure virtual machine images with Windows PowerShell and the Azure CLI](#).

If you want to use an existing disk instead of a marketplace image, use this code:

```

ManagedDisk managedDisk = azure.disks.define("myosdisk")
    .withRegion(Region.US_EAST)
    .withExistingResourceGroup("myResourceGroup")
    .withWindowsFromVhd("https://mystorage.blob.core.windows.net/vhds/myosdisk.vhd")
    .withSizeInGB(128)
    .withSku(DiskSkuTypes.PremiumLRS)
    .create();

azure.virtualMachines.define("myVM")
    .withRegion(Region.US_EAST)
    .withExistingResourceGroup("myResourceGroup")
    .withExistingPrimaryNetworkInterface(networkInterface)
    .withSpecializedOSDisk(managedDisk, OperatingSystemTypes.Windows)
    .withExistingAvailabilitySet(availabilitySet)
    .withSize(VirtualMachineSizeTypes.StandardDS1)
    .create();

```

Perform management tasks

During the lifecycle of a virtual machine, you may want to run management tasks such as starting, stopping, or deleting a virtual machine. Additionally, you may want to create code to automate repetitive or complex tasks.

When you need to do anything with the VM, you need to get an instance of it. Add this code to the try block of the main method:

```
VirtualMachine vm = azure.virtualMachines().getByResourceGroup("myResourceGroup", "myVM");
```

Get information about the VM

To get information about the virtual machine, add this code to the try block in the main method:

```
System.out.println("hardwareProfile");
System.out.println("    vmSize: " + vm.size());
System.out.println("storageProfile");
System.out.println("    imageReference");
System.out.println("        publisher: " + vm.storageProfile().imageReference().publisher());
System.out.println("        offer: " + vm.storageProfile().imageReference().offer());
System.out.println("        sku: " + vm.storageProfile().imageReference().sku());
System.out.println("        version: " + vm.storageProfile().imageReference().version());
System.out.println("    osDisk");
System.out.println("        osType: " + vm.storageProfile().osDisk().osType());
System.out.println("        name: " + vm.storageProfile().osDisk().name());
System.out.println("        createOption: " + vm.storageProfile().osDisk().createOption());
System.out.println("        caching: " + vm.storageProfile().osDisk().caching());
System.out.println("osProfile");
System.out.println("    computerName: " + vm.osProfile().computerName());
System.out.println("    adminUserName: " + vm.osProfile().adminUsername());
System.out.println("    provisionVMAgent: " + vm.osProfile().windowsConfiguration().provisionVMAgent());
System.out.println("    enableAutomaticUpdates: " +
vm.osProfile().windowsConfiguration().enableAutomaticUpdates());
System.out.println("networkProfile");
System.out.println("    networkInterface: " + vm.primaryNetworkInterfaceId());
System.out.println("vmAgent");
System.out.println("    vmAgentVersion: " + vm.instanceView().vmAgent().vmAgentVersion());
System.out.println("    statuses");
for(InstanceViewStatus status : vm.instanceView().vmAgent().statuses()) {
    System.out.println("        code: " + status.code());
    System.out.println("        displayStatus: " + status.displayStatus());
    System.out.println("        message: " + status.message());
    System.out.println("        time: " + status.time());
}
System.out.println("disks");
for(DiskInstanceView disk : vm.instanceView().disks()) {
    System.out.println("    name: " + disk.name());
    System.out.println("    statuses");
    for(InstanceViewStatus status : disk.statuses()) {
        System.out.println("        code: " + status.code());
        System.out.println("        displayStatus: " + status.displayStatus());
        System.out.println("        time: " + status.time());
    }
}
System.out.println("VM general status");
System.out.println("    provisioningStatus: " + vm.provisioningState());
System.out.println("    id: " + vm.id());
System.out.println("    name: " + vm.name());
System.out.println("    type: " + vm.type());
System.out.println("VM instance status");
for(InstanceViewStatus status : vm.instanceView().statuses()) {
    System.out.println("    code: " + status.code());
    System.out.println("    displayStatus: " + status.displayStatus());
}
System.out.println("Press enter to continue...");
input.nextLine();
```

Stop the VM

You can stop a virtual machine and keep all its settings, but continue to be charged for it, or you can stop a virtual machine and deallocate it. When a virtual machine is deallocated, all resources associated with it are also deallocated and billing ends for it.

To stop the virtual machine without deallocating it, add this code to the try block in the main method:

```
System.out.println("Stopping vm...");
vm.powerOff();
System.out.println("Press enter to continue...");
input.nextLine();
```

If you want to deallocate the virtual machine, change the PowerOff call to this code:

```
vm.deallocate();
```

Start the VM

To start the virtual machine, add this code to the try block in the main method:

```
System.out.println("Starting vm...");
vm.start();
System.out.println("Press enter to continue...");
input.nextLine();
```

Resize the VM

Many aspects of deployment should be considered when deciding on a size for your virtual machine. For more information, see [VM sizes](#).

To change size of the virtual machine, add this code to the try block in the main method:

```
System.out.println("Resizing vm...");
vm.update()
    .withSize(VirtualMachineSizeTypes.STANDARD_DS2)
    .apply();
System.out.println("Press enter to continue...");
input.nextLine();
```

Add a data disk to the VM

To add a data disk to the virtual machine that is 2 GB in size, has a LUN of 0, and a caching type of ReadWrite, add this code to the try block in the main method:

```
System.out.println("Adding data disk...");
vm.update()
    .withNewDataDisk(2, 0, CachingTypes.READ_WRITE)
    .apply();
System.out.println("Press enter to delete resources...");
input.nextLine();
```

Delete resources

Because you are charged for resources used in Azure, it is always good practice to delete resources that are no longer needed. If you want to delete the virtual machines and all the supporting resources, all you have to do is delete the resource group.

1. To delete the resource group, add this code to the try block in the main method:

```
System.out.println("Deleting resources...");  
azure.resourceGroups().deleteByName("myResourceGroup");
```

1. Save the App.java file.

Run the application

It should take about five minutes for this console application to run completely from start to finish.

1. To run the application, use this Maven command:

```
mvn compile exec:java
```

2. Before you press **Enter** to start deleting resources, you could take a few minutes to verify the creation of the resources in the Azure portal. Click the deployment status to see information about the deployment.

Next steps

- Learn more about using the [Azure libraries for Java](#).

Create and manage Windows VMs in Azure using Python

8/21/2017 • 10 min to read • [Edit Online](#)

An [Azure Virtual Machine](#) (VM) needs several supporting Azure resources. This article covers creating, managing, and deleting VM resources using Python. You learn how to:

- Create a Visual Studio project
- Install packages
- Create credentials
- Create resources
- Perform management tasks
- Delete resources
- Run the application

It takes about 20 minutes to do these steps.

Create a Visual Studio project

1. If you haven't already, install [Visual Studio](#). Select **Python development** on the Workloads page, and then click **Install**. In the summary, you can see that **Python 3 64-bit (3.6.0)** is automatically selected for you. If you have already installed Visual Studio, you can add the Python workload using the Visual Studio Launcher.
2. After installing and starting Visual Studio, click **File > New > Project**.
3. Click **Templates > Python > Python Application**, enter *myPythonProject* for the name of the project, select the location of the project, and then click **OK**.

Install packages

1. In Solution Explorer, under *myPythonProject*, right-click **Python Environments**, and then select **Add virtual environment**.
2. On the Add Virtual Environment screen, accept the default name of *env*, make sure that *Python 3.6 (64-bit)* is selected for the base interpreter, and then click **Create**.
3. Right-click the *env* environment that you created, click **Install Python Package**, enter *azure* in the search box, and then press Enter.

You should see in the output windows that the azure packages were successfully installed.

Create credentials

Before you start this step, make sure that you have an [Active Directory service principal](#). You should also record the application ID, the authentication key, and the tenant ID that you need in a later step.

1. Open *myPythonProject.py* file that was created, and then add this code to enable your application to run:

```
if __name__ == "__main__":
```

2. To import the code that is needed, add these statements to the top of the .py file:

```
from azure.common.credentials import ServicePrincipalCredentials
from azure.mgmt.resource import ResourceManagementClient
from azure.mgmt.compute import ComputeManagementClient
from azure.mgmt.network import NetworkManagementClient
from azure.mgmt.compute.models import DiskCreateOption
```

3. Next in the .py file, add variables after the import statements to specify common values used in the code:

```
SUBSCRIPTION_ID = 'subscription-id'
GROUP_NAME = 'myResourceGroup'
LOCATION = 'westus'
VM_NAME = 'myVM'
```

Replace **subscription-id** with your subscription identifier.

4. To create the Active Directory credentials that you need to make requests, add this function after the variables in the .py file:

```
def get_credentials():
    credentials = ServicePrincipalCredentials(
        client_id = 'application-id',
        secret = 'authentication-key',
        tenant = 'tenant-id'
    )

    return credentials
```

Replace **application-id**, **authentication-key**, and **tenant-id** with the values that you previously collected when you created your Azure Active Directory service principal.

5. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
credentials = get_credentials()
```

Create resources

Initialize management clients

Management clients are needed to create and manage resources using the Python SDK in Azure. To create the management clients, add this code under the **if** statement at then end of the .py file:

```
resource_group_client = ResourceManagementClient(
    credentials,
    SUBSCRIPTION_ID
)
network_client = NetworkManagementClient(
    credentials,
    SUBSCRIPTION_ID
)
compute_client = ComputeManagementClient(
    credentials,
    SUBSCRIPTION_ID
)
```

Create the VM and supporting resources

All resources must be contained in a [Resource group](#).

1. To create a resource group, add this function after the variables in the .py file:

```
def create_resource_group(resource_group_client):
    resource_group_params = { 'location':LOCATION }
    resource_group_result = resource_group_client.resource_groups.create_or_update(
        GROUP_NAME,
        resource_group_params
    )
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
create_resource_group(resource_group_client)
input('Resource group created. Press enter to continue...')
```

A [Availability sets](#) make it easier for you to maintain the virtual machines used by your application.

1. To create an availability set, add this function after the variables in the .py file:

```
def create_availability_set(compute_client):
    avset_params = {
        'location': LOCATION,
        'sku': { 'name': 'Aligned' },
        'platform_fault_domain_count': 3
    }
    availability_set_result = compute_client.availability_sets.create_or_update(
        GROUP_NAME,
        'myAVSet',
        avset_params
    )
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
create_availability_set(compute_client)
print("-----")
input('Availability set created. Press enter to continue...')
```

A [Public IP address](#) is needed to communicate with the virtual machine.

1. To create a public IP address for the virtual machine, add this function after the variables in the .py file:

```
def create_public_ip_address(network_client):
    public_ip_address_params = {
        'location': LOCATION,
        'public_ip_allocation_method': 'Dynamic'
    }
    creation_result = network_client.public_ip_addresses.create_or_update(
        GROUP_NAME,
        'myIPAddress',
        public_ip_address_params
    )

    return creation_result.result()
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
creation_result = create_public_ip_address(network_client)
print("-----")
print(creation_result)
input('Press enter to continue...')
```

A virtual machine must be in a subnet of a [Virtual network](#).

1. To create a virtual network, add this function after the variables in the .py file:

```
def create_vnet(network_client):
    vnet_params = {
        'location': LOCATION,
        'address_space': {
            'address_prefixes': ['10.0.0.0/16']
        }
    }
    creation_result = network_client.virtual_networks.create_or_update(
        GROUP_NAME,
        'myVNet',
        vnet_params
    )
    return creation_result.result()
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
creation_result = create_vnet(network_client)
print("-----")
print(creation_result)
input('Press enter to continue...')
```

3. To add a subnet to the virtual network, add this function after the variables in the .py file:

```
def create_subnet(network_client):
    subnet_params = {
        'address_prefix': '10.0.0.0/24'
    }
    creation_result = network_client.subnets.create_or_update(
        GROUP_NAME,
        'myVNet',
        'mySubnet',
        subnet_params
    )
    return creation_result.result()
```

4. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
creation_result = create_subnet(network_client)
print("-----")
print(creation_result)
input('Press enter to continue...')
```

A virtual machine needs a network interface to communicate on the virtual network.

1. To create a network interface, add this function after the variables in the .py file:

```

def create_nic(network_client):
    subnet_info = network_client.subnets.get(
        GROUP_NAME,
        'myVNet',
        'mySubnet'
    )
    publicIPAddress = network_client.public_ip_addresses.get(
        GROUP_NAME,
        'myIPAddress'
    )
    nic_params = {
        'location': LOCATION,
        'ip_configurations': [
            {
                'name': 'myIPConfig',
                'public_ip_address': publicIPAddress,
                'subnet': {
                    'id': subnet_info.id
                }
            }
        ]
    }
    creation_result = network_client.network_interfaces.create_or_update(
        GROUP_NAME,
        'myNic',
        nic_params
    )

    return creation_result.result()

```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```

creation_result = create_nic(network_client)
print("-----")
print(creation_result)
input('Press enter to continue...')

```

Now that you created all the supporting resources, you can create a virtual machine.

1. To create the virtual machine, add this function after the variables in the .py file:

```

def create_vm(network_client, compute_client):
    nic = network_client.network_interfaces.get(
        GROUP_NAME,
        'myNic'
    )
    avset = compute_client.availability_sets.get(
        GROUP_NAME,
        'myAVSet'
    )
    vm_parameters = {
        'location': LOCATION,
        'os_profile': {
            'computer_name': VM_NAME,
            'admin_username': 'azureuser',
            'admin_password': 'Azure12345678'
        },
        'hardware_profile': {
            'vm_size': 'Standard_DS1'
        },
        'storage_profile': {
            'image_reference': {
                'publisher': 'MicrosoftWindowsServer',
                'offer': 'WindowsServer',
                'sku': '2012-R2-Datacenter',
                'version': 'latest'
            }
        },
        'network_profile': {
            'network_interfaces': [
                {
                    'id': nic.id
                }
            ]
        },
        'availability_set': {
            'id': avset.id
        }
    }
    creation_result = compute_client.virtual_machines.create_or_update(
        GROUP_NAME,
        VM_NAME,
        vm_parameters
    )

    return creation_result.result()

```

NOTE

This tutorial creates a virtual machine running a version of the Windows Server operating system. To learn more about selecting other images, see [Navigate and select Azure virtual machine images with Windows PowerShell and the Azure CLI](#).

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```

creation_result = create_vm(network_client, compute_client)
print("-----")
print(creation_result)
input('Press enter to continue...')

```

Perform management tasks

During the lifecycle of a virtual machine, you may want to run management tasks such as starting, stopping, or deleting a virtual machine. Additionally, you may want to create code to automate repetitive or complex tasks.

Get information about the VM

1. To get information about the virtual machine, add this function after the variables in the .py file:

```
def get_vm(compute_client):
    vm = compute_client.virtual_machines.get(GROUP_NAME, VM_NAME, expand='instanceView')
    print("hardwareProfile")
    print("  vmSize: ", vm.hardware_profile.vm_size)
    print("\nstorageProfile")
    print("  imageReference")
    print("    publisher: ", vm.storage_profile.image_reference.publisher)
    print("    offer: ", vm.storage_profile.image_reference.offer)
    print("    sku: ", vm.storage_profile.image_reference.sku)
    print("    version: ", vm.storage_profile.image_reference.version)
    print("  osDisk")
    print("    osType: ", vm.storage_profile.os_disk.os_type.value)
    print("    name: ", vm.storage_profile.os_disk.name)
    print("    createOption: ", vm.storage_profile.os_disk.create_option.value)
    print("    caching: ", vm.storage_profile.os_disk.caching.value)
    print("\nosProfile")
    print("  computerName: ", vm.os_profile.computer_name)
    print("  adminUsername: ", vm.os_profile.admin_username)
    print("  provisionVMAgent: {0}".format(vm.os_profile.windows_configuration.provision_vm_agent))
    print("  enableAutomaticUpdates:
{0}".format(vm.os_profile.windows_configuration.enable_automatic_updates))
    print("\nnetworkProfile")
    for nic in vm.network_profile.network_interfaces:
        print("  networkInterface id: ", nic.id)
    print("\nvmAgent")
    print("  vmAgentVersion", vm.instance_view.vm_agent.vm_agent_version)
    print("  statuses")
    for stat in vm_result.instance_view.vm_agent.statuses:
        print("    code: ", stat.code)
        print("    displayStatus: ", stat.display_status)
        print("    message: ", stat.message)
        print("    time: ", stat.time)
    print("\ndisks");
    for disk in vm.instance_view.disks:
        print("  name: ", disk.name)
        print("  statuses")
        for stat in disk.statuses:
            print("    code: ", stat.code)
            print("    displayStatus: ", stat.display_status)
            print("    time: ", stat.time)
    print("\nVM general status")
    print("  provisioningStatus: ", vm.provisioning_state)
    print("  id: ", vm.id)
    print("  name: ", vm.name)
    print("  type: ", vm.type)
    print("  location: ", vm.location)
    print("\nVM instance status")
    for stat in vm.instance_view.statuses:
        print("  code: ", stat.code)
        print("  displayStatus: ", stat.display_status)
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
get_vm(compute_client)
print("-----")
input('Press enter to continue...')
```

Stop the VM

You can stop a virtual machine and keep all its settings, but continue to be charged for it, or you can stop a virtual machine and deallocate it. When a virtual machine is deallocated, all resources associated with it are also

deallocated and billing ends for it.

1. To stop the virtual machine without deallocating it, add this function after the variables in the .py file:

```
def stop_vm(compute_client):
    compute_client.virtual_machines.power_off(GROUP_NAME, VM_NAME)
```

If you want to deallocate the virtual machine, change the power_off call to this code:

```
compute_client.virtual_machines.deallocate(GROUP_NAME, VM_NAME)
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
stop_vm(compute_client)
input('Press enter to continue...')
```

Start the VM

1. To start the virtual machine, add this function after the variables in the .py file:

```
def start_vm(compute_client):
    compute_client.virtual_machines.start(GROUP_NAME, VM_NAME)
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
start_vm(compute_client)
input('Press enter to continue...')
```

Resize the VM

Many aspects of deployment should be considered when deciding on a size for your virtual machine. For more information, see [VM sizes](#).

1. To change the size of the virtual machine, add this function after the variables in the .py file:

```
def update_vm(compute_client):
    vm = compute_client.virtual_machines.get(GROUP_NAME, VM_NAME)
    vm.hardware_profile.vm_size = 'Standard_DS3'
    update_result = compute_client.virtual_machines.create_or_update(
        GROUP_NAME,
        VM_NAME,
        vm
    )

    return update_result.result()
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
update_result = update_vm(compute_client)
print("-----")
print(update_result)
input('Press enter to continue...')
```

Add a data disk to the VM

Virtual machines can have one or more [data disks](#) that are stored as VHDS.

1. To add a data disk to the virtual machine, add this function after the variables in the .py file:

```
def add_datadisk(compute_client):
    disk_creation = compute_client.disks.create_or_update(
        GROUP_NAME,
        'myDataDisk1',
        {
            'location': LOCATION,
            'disk_size_gb': 1,
            'creation_data': {
                'create_option': DiskCreateOption.empty
            }
        }
    )
    data_disk = disk_creation.result()
    vm = compute_client.virtual_machines.get(GROUP_NAME, VM_NAME)
    add_result = vm.storage_profile.data_disks.append({
        'lun': 1,
        'name': 'myDataDisk1',
        'create_option': DiskCreateOption.attach,
        'managed_disk': {
            'id': data_disk.id
        }
    })
    add_result = compute_client.virtual_machines.create_or_update(
        GROUP_NAME,
        VM_NAME,
        vm)

return add_result.result()
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
add_result = add_datadisk(compute_client)
print("-----")
print(add_result)
input('Press enter to continue...')
```

Delete resources

Because you are charged for resources used in Azure, it's always a good practice to delete resources that are no longer needed. If you want to delete the virtual machines and all the supporting resources, all you have to do is delete the resource group.

1. To delete the resource group and all resources, add this function after the variables in the .py file:

```
def delete_resources(resource_group_client):
    resource_group_client.resource_groups.delete(GROUP_NAME)
```

2. To call the function that you previously added, add this code under the **if** statement at the end of the .py file:

```
delete_resources(resource_group_client)
```

3. Save *myPythonProject.py*.

Run the application

1. To run the console application, click **Start** in Visual Studio.

2. Press **Enter** after the status of each resource is returned. In the status information, you should see a **Succeeded** provisioning state. After the virtual machine is created, you have the opportunity to delete all the resources that you create. Before you press **Enter** to start deleting resources, you could take a few minutes to verify their creation in the Azure portal. If you have the Azure portal open, you might have to refresh the blade to see new resources.

It should take about five minutes for this console application to run completely from start to finish. It may take several minutes after the application has finished before all the resources and the resource group are deleted.

Next steps

- If there were issues with the deployment, a next step would be to look at [Troubleshooting resource group deployments with Azure portal](#)
- Learn more about the [Azure Python Library](#)

Create a Windows virtual machine from a Resource Manager template

8/21/2017 • 3 min to read • [Edit Online](#)

This article shows you how to deploy an Azure Resource Manager template using PowerShell. The template that you create deploys a single virtual machine running Windows Server in a new virtual network with a single subnet.

For a detailed description of the virtual machine resource, see [Virtual machines in an Azure Resource Manager template](#). For more information about all the resources in a template, see [Azure Resource Manager template walkthrough](#).

It should take about five minutes to do the steps in this article.

Install Azure PowerShell

See [How to install and configure Azure PowerShell](#) for information about installing the latest version of Azure PowerShell, selecting your subscription, and signing in to your account.

Create a resource group

All resources must be deployed in a [resource group](#).

1. Get a list of available locations where resources can be created.

```
Get-AzureRmLocation | sort DisplayName | Select DisplayName
```

2. Create the resource group in the location that you select. This example shows the creation of a resource group named **myResourceGroup** in the **West US** location:

```
New-AzureRmResourceGroup -Name "myResourceGroup" -Location "West US"
```

Create the files

In this step, you create a template file that deploys the resources and a parameters file that supplies parameter values to the template. You also create an authorization file that is used to perform Azure Resource Manager operations.

1. Create a file named *CreateVMTemplate.json* and add this JSON code to it:

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "adminUsername": { "type": "string" },
    "adminPassword": { "type": "securestring" }
  },
  "variables": {
    "vnetID": "[resourceId('Microsoft.Network/virtualNetworks', 'myVNet')]",
    "subnetRef": "[concat(variables('vnetID'), '/subnets/mySubnet')]"
  },
  "resources": [
    {
      "type": "Microsoft.Compute/virtualMachines",
      "name": "myVM",
      "apiVersion": "2015-06-15",
      "location": "West US",
      "properties": {
        "osProfile": {
          "computerName": "myVM",
          "adminUsername": "[parameters('adminUsername')]",
          "adminPassword": "[parameters('adminPassword')]"
        },
        "hardwareProfile": {
          "vmSize": "Standard_DS1_v2"
        },
        "networkProfile": {
          "networkInterfaces": [
            {
              "id": "[variables('vnetID')]",
              "primary": true,
              "properties": {
                "subnet": "[variables('subnetRef')]"
              }
            }
          ]
        }
      }
    }
  ]
}
```

```

    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/publicIPAddresses",
    "name": "myPublicIPAddress",
    "location": "[resourceGroup().location]",
    "properties": {
        "publicIPAllocationMethod": "Dynamic",
        "dnsSettings": {
            "domainNameLabel": "myresourcegroupdns1"
        }
    }
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/virtualNetworks",
    "name": "myVNet",
    "location": "[resourceGroup().location]",
    "properties": {
        "addressSpace": { "addressPrefixes": [ "10.0.0.0/16" ] },
        "subnets": [
            {
                "name": "mySubnet",
                "properties": { "addressPrefix": "10.0.0.0/24" }
            }
        ]
    }
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/networkInterfaces",
    "name": "myNic",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Network/publicIPAddresses/', 'myPublicIPAddress')]",
        "[resourceId('Microsoft.Network/virtualNetworks/', 'myVNet')]"
    ],
    "properties": {
        "ipConfigurations": [
            {
                "name": "ipconfig1",
                "properties": {
                    "privateIPAllocationMethod": "Dynamic",
                    "publicIPAddress": { "id": "[resourceId('Microsoft.Network/publicIPAddresses', 'myPublicIPAddress')]" },
                    "subnet": { "id": "[variables('subnetRef')]" }
                }
            }
        ]
    }
},
{
    "apiVersion": "2016-04-30-preview",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "myVM",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Network/networkInterfaces/', 'myNic')]"
    ],
    "properties": {
        "hardwareProfile": { "vmSize": "Standard_DS1" },
        "osProfile": {
            "computerName": "myVM",
            "adminUsername": "[parameters('adminUsername')]",
            "adminPassword": "[parameters('adminPassword')]"
        },
        "storageProfile": {
            "imageReference": {
                "publisher": "MicrosoftWindowsServer",
                "offer": "WindowsServer",
                "sku": "2012-R2-Datacenter"
            }
        }
    }
}

```

```

    "sku": "2012-R2-Datacenter",
    "version": "latest"
  },
  "osDisk": {
    "name": "myManagedOSDisk",
    "caching": "ReadWrite",
    "createOption": "FromImage"
  }
},
"networkProfile": {
  "networkInterfaces": [
    {
      "id": "[resourceId('Microsoft.Network/networkInterfaces', 'myNic')]"
    }
  ]
}
}
]
}
}

```

2. Create a file named *Parameters.json* and add this JSON code to it:

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "adminUserName": { "value": "azureuser" },
    "adminPassword": { "value": "Azure12345678" }
  }
}
```

3. Create a new storage account and container:

```
$storageName = "st" + (Get-Random)
New-AzureRmStorageAccount -ResourceGroupName "myResourceGroup" -AccountName $storageName -Location
"West US" -SkuName "Standard_LRS" -Kind Storage
$accountKey = (Get-AzureRmStorageAccountKey -ResourceGroupName myResourceGroup -Name
$storageName).Value[0]
$context = New-AzureStorageContext -StorageAccountName $storageName -StorageAccountKey $accountKey
New-AzureStorageContainer -Name "templates" -Context $context -Permission Container
```

4. Upload the files to the storage account:

```
Set-AzureStorageBlobContent -File "C:\templates\CreateVMTemplate.json" -Context $context -Container
"templates"
Set-AzureStorageBlobContent -File "C:\templates\Parameters.json" -Context $context -Container templates
```

Change the -File paths to the location where you stored the files.

Create the resources

Deploy the template using the parameters:

```
$templatePath = "https://" + $storageName + ".blob.core.windows.net/templates/CreateVMTemplate.json"
$parametersPath = "https://" + $storageName + ".blob.core.windows.net/templates/Parameters.json"
New-AzureRmResourceGroupDeployment -ResourceGroupName "myResourceGroup" -Name "myDeployment" -TemplateUri
$templatePath -TemplateParameterUri $parametersPath
```

NOTE

You can also deploy templates and parameters from local files. To learn more, see [Using Azure PowerShell with Azure Storage](#).

Next Steps

- If there were issues with the deployment, you might take a look at [Troubleshoot common Azure deployment errors with Azure Resource Manager](#).
- Learn how to create and manage a virtual machine in [Create and manage Windows VMs with the Azure PowerShell module](#).

How to connect and log on to an Azure virtual machine running Windows

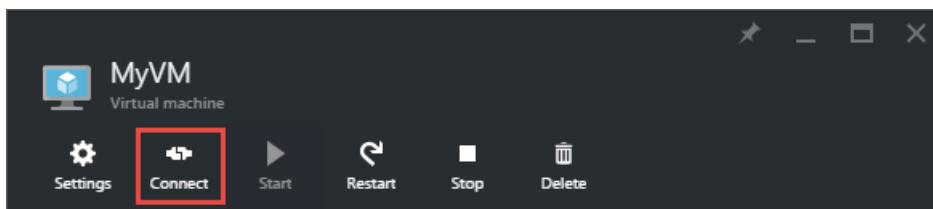
10/23/2017 • 1 min to read • [Edit Online](#)

You'll use the **Connect** button in the Azure portal to start a Remote Desktop (RDP) session from a Windows desktop. First you connect to the virtual machine, then you log on.

If you are trying to connect to a Windows VM from a Mac, you need to install an RDP client for Mac like [Microsoft Remote Desktop](#).

Connect to the virtual machine

1. If you haven't already done so, sign in to the [Azure portal](#).
2. In the left menu, click **Virtual Machines**.
3. Select the virtual machine from the list.
4. On the page for the virtual machine, click **Connect**.

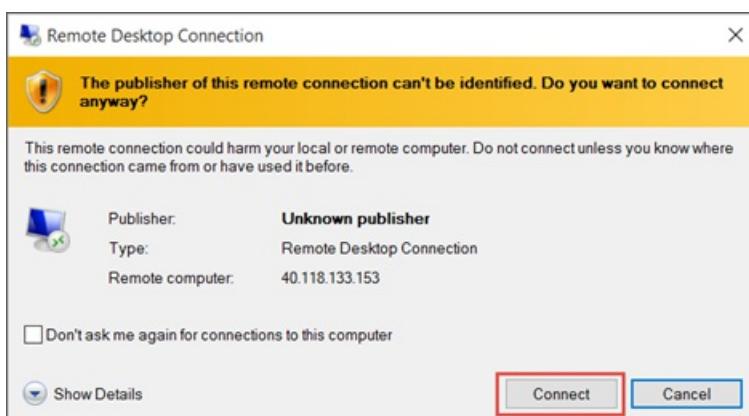


TIP

If the **Connect** button in the portal is greyed out and you are not connected to Azure via an [Express Route](#) or [Site-to-Site VPN](#) connection, you need to create and assign your VM a public IP address before you can use RDP. You can read more about [public IP addresses in Azure](#).

Log on to the virtual machine

1. Clicking **Connect** creates and downloads a Remote Desktop Protocol file (.rdp file). Click **Open** to use this file.
2. You get a warning that the **.rdp** file is from an unknown publisher. This is normal. In the Remote Desktop window, click **Connect** to continue.



3. In the **Windows Security** window, select **More choices** and then **Use a different account**. Type the credentials for an account on the virtual machine and then click **OK**.

Local account - this is usually the local account user name and password that you specified when you created the virtual machine. In this case, the domain is the name of the virtual machine and it is entered as `vmname\username`.

Domain joined VM - if the VM belongs to a domain, enter the user name in the format `Domain\Username`. The account also needs to either be in the Administrators group or have been granted remote access privileges to the VM.

Domain controller - if the VM is a domain controller, type the user name and password of a domain administrator account for that domain.

4. Click **Yes** to verify the identity of the virtual machine and finish logging on.



Next steps

If you run into trouble when you try to connect, see [Troubleshoot Remote Desktop connections](#). This article walks you through diagnosing and resolving common problems.

Azure Hybrid Benefit for Windows Server

12/7/2017 • 6 min to read • [Edit Online](#)

For customers with Software Assurance, Azure Hybrid Benefit for Windows Server allows you to use your on-premises Windows Server licenses and run Windows virtual machines on Azure at a reduced cost. You can use Azure Hybrid Benefit for Windows Server to deploy new virtual machines from any Azure supported platform Windows Server image or Windows custom images. This article goes over the steps on how to deploy new VMs with Azure Hybrid Benefit for Windows Server and how you can update existing running VMs. For more information about Azure Hybrid Benefit for Windows Server licensing and cost savings, see the [Azure Hybrid Benefit for Windows Server licensing page](#).

IMPORTANT

The legacy '[HUB]' Windows Server images that were published for customers with Enterprise Agreement on Azure Marketplace has been retired as of 9/11/2017, use the standard Windows Server with the "Save Money" option on the portal for Azure Hybrid Benefit for Windows Server. For more information, please refer to this [article](#).

NOTE

Using Azure Hybrid Benefit for Windows Server with VMs that are charged for additional software such as SQL Server or any of the third-party marketplace images is being rolled out. If you get a 409 error such as: Changing property 'LicenseType' is not allowed; then you are trying to convert or deploy a new Windows Server VM that has additional software cost, which may not be supported in that region. Same if you try to look for the portal configuration option to do the conversion and you can't see it for that VM.

NOTE

For classic VMs, only deploying new VM from on-prem custom images is supported. To take advantage of the capabilities supported in this article, you must first migrate classic VMs to Resource Manager model.

Ways to use Azure Hybrid Benefit for Windows Server

There are few ways to use Windows virtual machines with the Azure Hybrid Benefit:

1. You can deploy VMs from one of the provided
2. You can [upload a custom VM](#) and [deploy using a Resource Manager template](#) or [Azure PowerShell](#)
3. You can toggle and convert existing VM between running with Azure Hybrid Benefit or pay on-demand cost for Windows Server
4. You can also deploy a new virtual machine scale set with Azure Hybrid Benefit for Windows Server

NOTE

Converting an existing virtual machine scale set to use Azure Hybrid Benefit for Windows Server isn't supported

Deploy a VM from a Windows Server Marketplace Image

All Windows Server images that are available from the Azure Marketplace are enabled with Azure Hybrid Benefit

for Windows Server. For example, Windows Server 2016, Windows Server 2012R2, Windows Server 2012, and Windows Server 2008SP1 and more. You can use these images to deploy VMs directly from the Azure portal, Resource Manager templates, Azure PowerShell, or other SDKs.

You can deploy these images directly from the Azure portal. For use in Resource Manager templates and with Azure PowerShell, view the list of images as follows:

Powershell

```
Get-AzureRmVMImage | Where-Object { $_.Offer -eq "WindowsServer" }
```

You can follow the steps to and pass LicenseType = "Windows_Server". This option allows you to use your existing Windows Server license on Azure.

Portal

You can follow the steps to [Create a Windows virtual machine with the Azure portal](#) and select the option to use your existing Windows Server license.

Convert an existing VM using Azure Hybrid Benefit for Windows Server

If you have an existing VM that you would like to convert to take advantage of Azure Hybrid Benefit for Windows Server, you can update your VM's license type as follows:

Convert to using Azure Hybrid Benefit for Windows Server

```
$vm = Get-AzureRmVM -ResourceGroup "rg-name" -Name "vm-name"
$vm.LicenseType = "Windows_Server"
Update-AzureRmVM -ResourceGroupName rg-name -VM $vm
```

Convert back to pay as you go

```
$vm = Get-AzureRmVM -ResourceGroup "rg-name" -Name "vm-name"
$vm.LicenseType = "None"
Update-AzureRmVM -ResourceGroupName rg-name -VM $vm
```

Portal

From portal VM blade, you can update the VM to use Azure Hybrid Benefit by selecting "Configuration" option and toggle the "Azure hybrid benefit" option

NOTE

If you don't see the option to toggle "Azure hybrid benefit" under "Configuration", it is because the conversion isn't supported yet for the selected VM type (for example a VM built from custom image or from an image that has additional paid software like SQL Server or Azure Marketplace third-party software).

Upload a Windows Server VHD

To deploy a Windows Server VM in Azure, you first need to create a VHD that contains your base Windows build. This VHD must be appropriately prepared via Sysprep before you upload it to Azure. You can [read more about the VHD requirements and Sysprep process](#) and [Sysprep Support for Server Roles](#). Back up the VM before running Sysprep.

Once you have prepared your VHD, upload the VHD to your Azure Storage account as follows:

```
Add-AzureRmVhd -ResourceGroupName "myResourceGroup" -LocalFilePath "C:\Path\To\myvhd.vhd" `  
-Destination "https://mystorageaccount.blob.core.windows.net/vhds/myvhd.vhd"
```

NOTE

Microsoft SQL Server, SharePoint Server, and Dynamics can also utilize your Software Assurance licensing. You shall need to prepare the Windows Server image by installing your application components and providing license keys accordingly, then uploading the disk image to Azure. Review the appropriate documentation for running Sysprep with your application, such as [Considerations for Installing SQL Server using Sysprep](#) or [Build a SharePoint Server 2016 Reference Image \(Sysprep\)](#).

You can also read more about [uploading the VHD to Azure process](#)

Deploy a VM via Resource Manager Template

Within your Resource Manager templates, an additional parameter `licenseType` must be specified. You can read more about [authoring Azure Resource Manager templates](#). Once you have your VHD uploaded to Azure, edit your Resource Manager template to include the license type as part of the compute provider and deploy your template as normal:

```
"properties": {  
    "licenseType": "Windows_Server",  
    "hardwareProfile": {  
        "vmSize": "[variables('vmSize')]"  
    }  
}
```

Deploy a VM via PowerShell quickstart

When deploying your Windows Server VM via PowerShell, you have an additional parameter `-LicenseType`. Once you have your VHD uploaded to Azure, you create a VM using `New-AzureRmVM` and specify the licensing type as follows:

For Windows Server:

```
New-AzureRmVM -ResourceGroupName "myResourceGroup" -Location "West US" -VM $vm -LicenseType "Windows_Server"
```

You can read a more descriptive guide on the different steps to [create a Windows VM using Resource Manager and PowerShell](#).

Verify your VM is utilizing the licensing benefit

Once you have deployed your VM through either PowerShell, Resource Manager template or portal, you can verify the license type with `Get-AzureRmVM` as follows:

```
Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVM"
```

The output is similar to the following example for Windows Server:

Type	:	Microsoft.Compute/virtualMachines
Location	:	westus
LicenseType	:	Windows_Server

This output contrasts with the following VM deployed without Azure Hybrid Benefit for Windows Server licensing:

```
Type          : Microsoft.Compute/virtualMachines
Location     : westus
LicenseType  :
```

List all Azure Hybrid Benefit for Windows Server VMs in a subscription

To see and count all virtual machines deployed with Azure Hybrid Benefit for Windows Server, you can run the following command from your subscription:

```
$vms = Get-AzureRMVM
foreach ($vm in $vms) {"VM Name: " + $vm.Name, "    Azure Hybrid Benefit for Windows Server: "+ $vm.LicenseType}
```

Deploy a virtual machine scale set with Azure Hybrid Benefit for Windows Server

Within your virtual machine scale set Resource Manager templates, an additional parameter `licenseType` must be specified. You can read more about [authoring Azure Resource Manager templates](#). Edit your Resource Manager template to include the `licenseType` property as part of the scale set's `virtualMachineProfile` and deploy your template as normal - see following example using 2016 Windows Server image:

```
"virtualMachineProfile": {
  "storageProfile": {
    "osDisk": {
      "createOption": "FromImage"
    },
    "imageReference": {
      "publisher": "MicrosoftWindowsServer",
      "offer": "WindowsServer",
      "sku": "2016-Datacenter",
      "version": "latest"
    }
  },
  "licenseType": "Windows_Server",
  "osProfile": {
    "computerNamePrefix": "[parameters('vmssName')]",
    "adminUsername": "[parameters('adminUsername')]",
    "adminPassword": "[parameters('adminPassword')]"
  }
}
```

You can also [Create and deploy a virtual machine scale set](#) and set the `LicenseType` property

Next steps

Read more about [How to save money with the Azure Hybrid Benefit](#)

Learn more about [Azure Hybrid Benefit for Windows Server licensing detailed guidance](#)

Learn more about [Using Resource Manager templates](#)

Learn more about [Azure Hybrid Benefit for Windows Server and Azure Site Recovery make migrating applications to Azure even more cost-effective](#)

Learn more about [Windows 10 on Azure with Multitenant Hosting Right](#)

Read more about [Frequently asked questions](#)

How to deploy Windows 10 on Azure with Multitenant Hosting Rights

12/7/2017 • 2 min to read • [Edit Online](#)

For customers with Windows 10 Enterprise E3/E5 per user or Windows Virtual Desktop Access per user (User Subscription Licenses or Add-on User Subscription Licenses), Multitenant Hosting Rights for Windows 10 allows you to bring your Windows 10 Licenses to the cloud and run Windows 10 Virtual Machines on Azure without paying for another license. For more information, please see [Multitenant Hosting for Windows 10](#).

NOTE

This article shows you to implement the licensing benefit for Windows 10 Desktop images. You can refer to the following for [Azure Hybrid use benefits for Windows Server images](#).

Deploying Windows 10 Image from Azure Marketplace

For Powershell, CLI and Azure Resource Manager template deployments, the Windows 10 image can be found with the following publishername, offer, sku.

OS	PUBLISHERNAME	OFFER	SKU
Windows 10 Pro	MicrosoftWindowsDesktop	Windows-10	RS2-Pro
Windows 10 Pro N	MicrosoftWindowsDesktop	Windows-10	RS2-ProN

Uploading Windows 10 VHD to Azure

If you are uploading a generalized Windows 10 VHD, please note Windows 10 does not have built-in administrator account enabled by default. To enable the built-in administrator account, include the following command as part of the Custom Script extension.

```
Net user <username> /active:yes
```

The following powershell snippet is to mark all administrator accounts as active, including the built-in administrator. This example is useful if the built-in administrator username is unknown.

```
$adminAccount = Get-WmiObject Win32_UserAccount -filter "LocalAccount=True" | ? {$_.SID -Like "S-1-5-21-*-500"}  
if($adminAccount.Disabled)  
{  
    $adminAccount.Disabled = $false  
    $adminAccount.Put()  
}
```

For more information:

- [How to upload VHD to Azure](#)
- [How to prepare a Windows VHD to upload to Azure](#)

Deploying Windows 10 with Multitenant Hosting Rights

Make sure you have [installed and configured the latest Azure PowerShell](#). Once you have prepared your VHD, upload the VHD to your Azure Storage account using the `Add-AzureRmVhd` cmdlet as follows:

```
Add-AzureRmVhd -ResourceGroupName "myResourceGroup" -LocalFilePath "C:\Path\To\myvhd.vhd" `  
-Destination "https://mystorageaccount.blob.core.windows.net/vhds/myvhd.vhd"
```

Deploy using Azure Resource Manager Template Deployment Within your Resource Manager templates, an additional parameter for `licenseType` can be specified. You can read more about [authoring Azure Resource Manager templates](#). Once you have your VHD uploaded to Azure, edit your Resource Manager template to include the license type as part of the compute provider and deploy your template as normal:

```
"properties": {  
    "licenseType": "Windows_Client",  
    "hardwareProfile": {  
        "vmSize": "[variables('vmSize')]"  
    }  
}
```

Deploy via PowerShell When deploying your Windows Server VM via PowerShell, you have an additional parameter for `-LicenseType`. Once you have your VHD uploaded to Azure, you create a VM using `New-AzureRmVM` and specify the licensing type as follows:

```
New-AzureRmVM -ResourceGroupName "myResourceGroup" -Location "West US" -VM $vm -LicenseType "Windows_Client"
```

Verify your VM is utilizing the licensing benefit

Once you have deployed your VM through either the PowerShell or Resource Manager deployment method, verify the license type with `Get-AzureRmVM` as follows:

```
Get-AzureRmVM -ResourceGroup "myResourceGroup" -Name "myVM"
```

The output is similar to the following example for Windows 10 with correct license type:

```
Type : Microsoft.Compute/virtualMachines  
Location : westus  
LicenseType : Windows_Client
```

This output contrasts with the following VM deployed without Azure Hybrid Use Benefit licensing, such as a VM deployed straight from the Azure Gallery:

```
Type : Microsoft.Compute/virtualMachines  
Location : westus  
LicenseType :
```

Additional Information about joining Azure AD

NOTE

Azure provisions all Windows VMs with built-in administrator account, which cannot be used to join AAD. For example, *Settings > Account > Access Work or School > +Connect* will not work. You must create and log on as a second administrator account to join Azure AD manually. You can also configure Azure AD using a provisioning package, use the link in the *Next Steps* section to learn more.

Next Steps

- Learn more about [Configuring VDA for Windows 10](#)
- Learn more about [Multitenant Hosting for Windows 10](#)

How to encrypt virtual disks on a Windows VM

7/11/2017 • 7 min to read • [Edit Online](#)

For enhanced virtual machine (VM) security and compliance, virtual disks in Azure can be encrypted. Disks are encrypted using cryptographic keys that are secured in an Azure Key Vault. You control these cryptographic keys and can audit their use. This article details how to encrypt virtual disks on a Windows VM using Azure PowerShell. You can also [encrypt a Linux VM using the Azure CLI 2.0](#).

Overview of disk encryption

Virtual disks on Windows VMs are encrypted at rest using BitLocker. There is no charge for encrypting virtual disks in Azure. Cryptographic keys are stored in Azure Key Vault using software-protection, or you can import or generate your keys in Hardware Security Modules (HSMs) certified to FIPS 140-2 level 2 standards. These cryptographic keys are used to encrypt and decrypt virtual disks attached to your VM. You retain control of these cryptographic keys and can audit their use. An Azure Active Directory service principal provides a secure mechanism for issuing these cryptographic keys as VMs are powered on and off.

The process for encrypting a VM is as follows:

1. Create a cryptographic key in an Azure Key Vault.
2. Configure the cryptographic key to be usable for encrypting disks.
3. To read the cryptographic key from the Azure Key Vault, create an Azure Active Directory service principal with the appropriate permissions.
4. Issue the command to encrypt your virtual disks, specifying the Azure Active Directory service principal and appropriate cryptographic key to be used.
5. The Azure Active Directory service principal requests the required cryptographic key from Azure Key Vault.
6. The virtual disks are encrypted using the provided cryptographic key.

Encryption process

Disk encryption relies on the following additional components:

- **Azure Key Vault** - used to safeguard cryptographic keys and secrets used for the disk encryption/decryption process.
 - If one exists, you can use an existing Azure Key Vault. You do not have to dedicate a Key Vault to encrypting disks.
 - To separate administrative boundaries and key visibility, you can create a dedicated Key Vault.
- **Azure Active Directory** - handles the secure exchanging of required cryptographic keys and authentication for requested actions.
 - You can typically use an existing Azure Active Directory instance for housing your application.
 - The service principal provides a secure mechanism to request and be issued the appropriate cryptographic keys. You are not developing an actual application that integrates with Azure Active Directory.

Requirements and limitations

Supported scenarios and requirements for disk encryption:

- Enabling encryption on new Windows VMs from Azure Marketplace images or custom VHD image.

- Enabling encryption on existing Windows VMs in Azure.
- Enabling encryption on Windows VMs that are configured using Storage Spaces.
- Disabling encryption on OS and data drives for Windows VMs.
- All resources (such as Key Vault, Storage account, and VM) must be in the same Azure region and subscription.
- Standard tier VMs, such as A, D, DS, G, and GS series VMs.

Disk encryption is not currently supported in the following scenarios:

- Basic tier VMs.
- VMs created using the Classic deployment model.
- Updating the cryptographic keys on an already encrypted VM.
- Integration with on-prem Key Management Service.

Create Azure Key Vault and keys

Before you start, make sure that the latest version of the Azure PowerShell module has been installed. For more information, see [How to install and configure Azure PowerShell](#). Throughout the command examples, replace all example parameters with your own names, location, and key values. The following examples use a convention of *myResourceGroup*, *myKeyVault*, *myVM*, etc.

The first step is to create an Azure Key Vault to store your cryptographic keys. Azure Key Vault can store keys, secrets, or passwords that allow you to securely implement them in your applications and services. For virtual disk encryption, you create a Key Vault to store a cryptographic key that is used to encrypt or decrypt your virtual disks.

Enable the Azure Key Vault provider within your Azure subscription with [Register-AzureRmResourceProvider](#), then create a resource group with [New-AzureRmResourceGroup](#). The following example creates a resource group name *myResourceGroup* in the *East US* location:

```
$rgName = "myResourceGroup"
$location = "East US"

Register-AzureRmResourceProvider -ProviderNamespace "Microsoft.KeyVault"
New-AzureRmResourceGroup -Location $location -Name $rgName
```

The Azure Key Vault containing the cryptographic keys and associated compute resources such as storage and the VM itself must reside in the same region. Create an Azure Key Vault with [New-AzureRmKeyVault](#) and enable the Key Vault for use with disk encryption. Specify a unique Key Vault name for *keyVaultName* as follows:

```
$keyVaultName = "myUniqueKeyVaultName"
New-AzureRmKeyVault -Location $location ` 
    -ResourceGroupName $rgName ` 
    -VaultName $keyVaultName ` 
    -EnabledForDiskEncryption
```

You can store cryptographic keys using software or Hardware Security Model (HSM) protection. Using an HSM requires a premium Key Vault. There is an additional cost to creating a premium Key Vault rather than standard Key Vault that stores software-protected keys. To create a premium Key Vault, in the preceding step add the *-Sku "Premium"* parameters. The following example uses software-protected keys since we created a standard Key Vault.

For both protection models, the Azure platform needs to be granted access to request the cryptographic keys when the VM boots to decrypt the virtual disks. Create a cryptographic key in your Key Vault with [Add-AzureKeyVaultKey](#). The following example creates a key named *myKey*:

```
Add-AzureKeyVaultKey -VaultName $keyVaultName `  
-Name "myKey" `  
-Destination "Software"
```

Create the Azure Active Directory service principal

When virtual disks are encrypted or decrypted, you specify an account to handle the authentication and exchanging of cryptographic keys from Key Vault. This account, an Azure Active Directory service principal, allows the Azure platform to request the appropriate cryptographic keys on behalf of the VM. A default Azure Active Directory instance is available in your subscription, though many organizations have dedicated Azure Active Directory directories.

Create a service principal in Azure Active Directory with [New-AzureRmADServicePrincipal](#). To specify a secure password, follow the [Password policies and restrictions in Azure Active Directory](#):

```
$appName = "My App"  
$securePassword = "P@ssword!"  
$app = New-AzureRmADApplication -DisplayName $appName `  
-HomePage "https://myapp.contoso.com" `  
-IdentifierUris "https://contoso.com/myapp" `  
-Password $securePassword  
New-AzureRmADServicePrincipal -ApplicationId $app.ApplicationId
```

To successfully encrypt or decrypt virtual disks, permissions on the cryptographic key stored in Key Vault must be set to permit the Azure Active Directory service principal to read the keys. Set permissions on your Key Vault with [Set-AzureRmKeyVaultAccessPolicy](#):

```
Set-AzureRmKeyVaultAccessPolicy -VaultName $keyvaultName `  
-ServicePrincipalName $app.ApplicationId `  
-PermissionsToKeys "WrapKey" `  
-PermissionsToSecrets "Set"
```

Create virtual machine

To test the encryption process, let's create a VM. The following example creates a VM named *myVM* using a *Windows Server 2016 Datacenter* image:

```

$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig -Name mySubnet -AddressPrefix 192.168.1.0/24

$vnet = New-AzureRmVirtualNetwork -ResourceGroupName $rgName ` 
    -Location $location ` 
    -Name myVnet ` 
    -AddressPrefix 192.168.0.0/16 ` 
    -Subnet $subnetConfig

$pip = New-AzureRmPublicIpAddress -ResourceGroupName $rgName ` 
    -Location $location ` 
    -AllocationMethod Static ` 
    -IdleTimeoutInMinutes 4 ` 
    -Name "mypublicdns$(Get-Random)"

$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig -Name myNetworkSecurityGroupRuleRDP ` 
    -Protocol Tcp ` 
    -Direction Inbound ` 
    -Priority 1000 ` 
    -SourceAddressPrefix * ` 
    -SourcePortRange * ` 
    -DestinationAddressPrefix * ` 
    -DestinationPortRange 3389 ` 
    -Access Allow

$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $rgName ` 
    -Location $location ` 
    -Name myNetworkSecurityGroup ` 
    -SecurityRules $nsgRuleRDP

$nic = New-AzureRmNetworkInterface -Name myNic ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -SubnetId $vnet.Subnets[0].Id ` 
    -PublicIpAddressId $pip.Id ` 
    -NetworkSecurityGroupId $nsg.Id

$cred = Get-Credential

$vmName = "myVM"
$vmConfig = New-AzureRmVMConfig -VMName $vmName -VMSize Standard_D1 | ` 
Set-AzureRmVMOperatingSystem -Windows -ComputerName myVM -Credential $cred | ` 
Set-AzureRmVMSourceImage -PublisherName MicrosoftWindowsServer ` 
    -Offer WindowsServer -Skus 2016-Datacenter -Version latest | ` 
Add-AzureRmVMNetworkInterface -Id $nic.Id

New-AzureRmVM -ResourceGroupName $rgName -Location $location -VM $vmConfig

```

Encrypt virtual machine

To encrypt the virtual disks, you bring together all the previous components:

1. Specify the Azure Active Directory service principal and password.
2. Specify the Key Vault to store the metadata for your encrypted disks.
3. Specify the cryptographic keys to use for the actual encryption and decryption.
4. Specify whether you want to encrypt the OS disk, the data disks, or all.

Encrypt your VM with [Set-AzureRmVMDiskEncryptionExtension](#) using the Azure Key Vault key and Azure Active Directory service principal credentials. The following example retrieves all the key information then encrypts the VM named *myVM*:

```

$keyVault = Get-AzureRmKeyVault -VaultName $keyVaultName -ResourceGroupName $rgName;
$diskEncryptionKeyVaultUrl = $keyVault.VaultUri;
$keyVaultResourceId = $keyVault.ResourceId;
$keyEncryptionKeyUrl = (Get-AzureKeyVaultKey -VaultName $keyVaultName -Name myKey).Key.kid;

Set-AzureRmVMDiskEncryptionExtension -ResourceGroupName $rgName ` 
    -VMName $vmName ` 
    -AadClientID $app.ApplicationId ` 
    -AadClientSecret $securePassword ` 
    -DiskEncryptionKeyVaultUrl $diskEncryptionKeyVaultUrl ` 
    -DiskEncryptionKeyVaultId $keyVaultResourceId ` 
    -KeyEncryptionKeyUrl $keyEncryptionKeyUrl ` 
    -KeyEncryptionKeyVaultId $keyVaultResourceId

```

Accept the prompt to continue with the VM encryption. The VM restarts during the process. Once the encryption process completes and the VM has rebooted, review the encryption status with [Get-AzureRmVmDiskEncryptionStatus](#):

```
Get-AzureRmVmDiskEncryptionStatus -ResourceGroupName $rgName -VMName $vmName
```

The output is similar to the following example:

```

OsVolumeEncrypted      : Encrypted
DataVolumesEncrypted   : Encrypted
OsVolumeEncryptionSettings : Microsoft.Azure.Management.Compute.Models.DiskEncryptionSettings
ProgressMessage         : OsVolume: Encrypted, DataVolumes: Encrypted

```

Next steps

- For more information about managing Azure Key Vault, see [Set up Key Vault for virtual machines](#).
- For more information about disk encryption, such as preparing an encrypted custom VM to upload to Azure, see [Azure Disk Encryption](#).

Setting up WinRM access for Virtual Machines in Azure Resource Manager

6/27/2017 • 3 min to read • [Edit Online](#)

WinRM in Azure Service Management vs Azure Resource Manager

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using the Resource Manager deployment model, which Microsoft recommends for new deployments instead of the classic deployment model.

- For an overview of the Azure Resource Manager, please see this [article](#)
- For differences between Azure Service Management and Azure Resource Manager, please see this [article](#)

The key difference in setting up WinRM configuration between the two stacks is how the certificate gets installed on the VM. In the Azure Resource Manager stack, the certificates are modeled as resources managed by the Key Vault Resource Provider. Therefore, the user needs to provide their own certificate and upload it to a Key Vault before using it in a VM.

Here are the steps you need to take to set up a VM with WinRM connectivity

1. Create a Key Vault
2. Create a self-signed certificate
3. Upload your self-signed certificate to Key Vault
4. Get the URL for your self-signed certificate in the Key Vault
5. Reference your self-signed certificates URL while creating a VM

Step 1: Create a Key Vault

You can use the below command to create the Key Vault

```
New-AzureRmKeyVault -VaultName "<vault-name>" -ResourceGroupName "<rg-name>" -Location "<vault-location>" -EnabledForDeployment -EnabledForTemplateDeployment
```

Step 2: Create a self-signed certificate

You can create a self-signed certificate using this PowerShell script

```
$certificateName = "somename"

$thumbprint = (New-SelfSignedCertificate -DnsName $certificateName -CertStoreLocation Cert:\CurrentUser\My -KeySpec KeyExchange).Thumbprint

$cert = (Get-ChildItem -Path cert:\CurrentUser\My\$thumbprint)

$password = Read-Host -Prompt "Please enter the certificate password." -AsSecureString

Export-PfxCertificate -Cert $cert -FilePath ".\$certificateName.pfx" -Password $password
```

Step 3: Upload your self-signed certificate to the Key Vault

Before uploading the certificate to the Key Vault created in step 1, it needs to converted into a format the Microsoft.Compute resource provider will understand. The below PowerShell script will allow you do that

```
$fileName = "<Path to the .pfx file>"  
$fileContentBytes = Get-Content $fileName -Encoding Byte  
$fileContentEncoded = [System.Convert]::ToBase64String($fileContentBytes)  
  
$jsonObject = @"  
{  
    "data": "$fileContentEncoded",  
    "dataType" : "pfx",  
    "password": "<password>"  
}  
"@  
  
$jsonObjectBytes = [System.Text.Encoding]::UTF8.GetBytes($jsonObject)  
$jsonEncoded = [System.Convert]::ToBase64String($jsonObjectBytes)  
  
$secret = ConvertTo-SecureString -String $jsonEncoded -AsPlainText -Force  
Set-AzureKeyVaultSecret -VaultName "<vault name>" -Name "<secret name>" -SecretValue $secret
```

Step 4: Get the URL for your self-signed certificate in the Key Vault

The Microsoft.Compute resource provider needs a URL to the secret inside the Key Vault while provisioning the VM. This enables the Microsoft.Compute resource provider to download the secret and create the equivalent certificate on the VM.

NOTE

The URL of the secret needs to include the version as well. An example URL looks like below

<https://contosovault.vault.azure.net:443/secrets/contososecret/01h9db0df2cd4300a20ence585a6s7ve>

Templates

You can get the link to the URL in the template using the below code

```
"certificateUrl": "[reference(resourceId(resourceGroup().name, 'Microsoft.KeyVault/vaults/secrets', '<vault-name>', '<secret-name>'), '2015-06-01').secretUriWithVersion]"
```

PowerShell

You can get this URL using the below PowerShell command

```
$secretURL = (Get-AzureKeyVaultSecret -VaultName "<vault name>" -Name "<secret name>").Id
```

Step 5: Reference your self-signed certificates URL while creating a VM

Azure Resource Manager Templates

While creating a VM through templates, the certificate gets referenced in the secrets section and the winRM section as below:

```

"osProfile": {
    ...
    "secrets": [
        {
            "sourceVault": {
                "id": "<resource id of the Key Vault containing the secret>"
            },
            "vaultCertificates": [
                {
                    "certificateUrl": "<URL for the certificate you got in Step 4>",
                    "certificateStore": "<Name of the certificate store on the VM>"
                }
            ]
        }
    ],
    "windowsConfiguration": {
        ...
        "winRM": {
            "listeners": [
                {
                    "protocol": "http"
                },
                {
                    "protocol": "https",
                    "certificateUrl": "<URL for the certificate you got in Step 4>"
                }
            ]
        },
        ...
    }
},

```

A sample template for the above can be found here at [201-vm-winrm-keyvault-windows](#)

Source code for this template can be found on [GitHub](#)

PowerShell

```

$vm = New-AzureRmVMConfig -VMName "<VM name>" -VMSize "<VM Size>"
$credential = Get-Credential
$secretURL = (Get-AzureKeyVaultSecret -VaultName "<vault name>" -Name "<secret name>").Id
$vm = Set-AzureRmVMOperatingSystem -VM $vm -Windows -ComputerName "<Computer Name>" -Credential $credential -
WinRMHttp -WinRMHttps -WinRMCertificateUrl $secretURL
$sourceVaultId = (Get-AzureRmKeyVault -ResourceGroupName "<Resource Group name>" -VaultName "<Vault
Name>").ResourceId
$CertificateStore = "My"
$vm = Add-AzureRmVMSecret -VM $vm -SourceVaultId $sourceVaultId -CertificateStore $CertificateStore -
CertificateUrl $secretURL

```

Step 6: Connecting to the VM

Before you can connect to the VM you'll need to make sure your machine is configured for WinRM remote management. Start PowerShell as an administrator and execute the below command to make sure you're set up.

```
Enable-PSRemoting -Force
```

NOTE

You might need to make sure the WinRM service is running if the above does not work. You can do that using

```
Get-Service WinRM
```

Once the setup is done, you can connect to the VM using the below command

```
Enter-PSSession -ConnectionUri https://<public-ip-dns-of-the-vm>:5986 -Credential $cred -SessionOption (New-PSSessionOption -SkipCACheck -SkipCNCheck -SkipRevocationCheck) -Authentication Negotiate
```

Get started with Role-Based Access Control in the Azure portal

12/21/2017 • 3 min to read • [Edit Online](#)

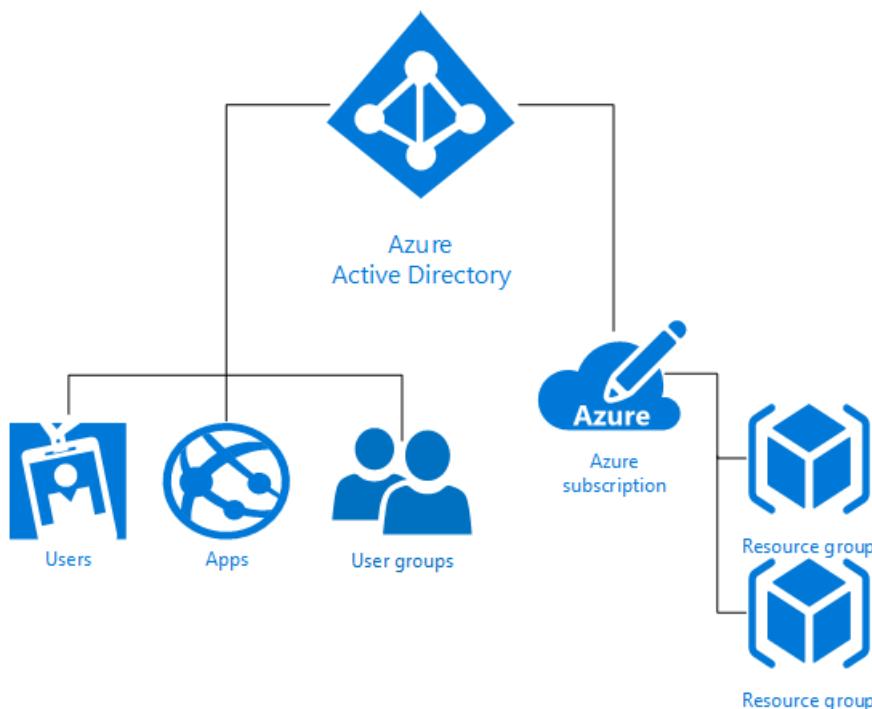
Security-oriented companies should focus on giving employees the exact permissions they need. Too many permissions can expose an account to attackers. Too few permissions means that employees can't get their work done efficiently. Azure Role-Based Access Control (RBAC) helps address this problem by offering fine-grained access management for Azure.

Using RBAC, you can segregate duties within your team and grant only the amount of access to users that they need to perform their jobs. Instead of giving everybody unrestricted permissions in your Azure subscription or resources, you can allow only certain actions. For example, use RBAC to let one employee manage virtual machines in a subscription, while another can manage SQL databases within the same subscription.

Basics of access management in Azure

Each Azure subscription is associated with one Azure Active Directory (AD) directory. Users, groups, and applications from that directory can manage resources in the Azure subscription. Assign these access rights using the Azure portal, Azure command-line tools, and Azure Management APIs.

Grant access by assigning the appropriate RBAC role to users, groups, and applications at a certain scope. The scope of a role assignment can be a subscription, a resource group, or a single resource. A role assigned at a parent scope also grants access to the children contained within it. For example, a user with access to a resource group can manage all the resources it contains, like websites, virtual machines, and subnets.



The RBAC role that you assign dictates what resources the user, group, or application can manage within that scope.

Built-in roles

Azure RBAC has three basic roles that apply to all resource types:

- **Owner** has full access to all resources including the right to delegate access to others.
- **Contributor** can create and manage all types of Azure resources but can't grant access to others.
- **Reader** can view existing Azure resources.

The rest of the RBAC roles in Azure allow management of specific Azure resources. For example, the Virtual Machine Contributor role allows the user to create and manage virtual machines. It does not give them access to the virtual network or the subnet that the virtual machine connects to.

[RBAC built-in roles](#) lists the roles available in Azure. It specifies the operations and scope that each built-in role grants to users. If you're looking to define your own roles for even more control, see how to build [Custom roles in Azure RBAC](#).

Resource hierarchy and access inheritance

- Each **subscription** in Azure belongs to only one directory. (But each directory can have more than one subscription.)
- Each **resource group** belongs to only one subscription.
- Each **resource** belongs to only one resource group.

Access that you grant at parent scopes is inherited at child scopes. For example:

- You assign the Reader role to an Azure AD group at the subscription scope. The members of that group can view every resource group and resource in the subscription.
- You assign the Contributor role to an application at the resource group scope. It can manage resources of all types in that resource group, but not other resource groups in the subscription.

Azure RBAC vs. classic subscription administrators

[Classic subscription administrators and co-admins](#) have full access to the Azure subscription. They can manage resources using the [Azure portal](#), Azure Resource Manager APIs, and the classic deployment model APIs. In the RBAC model, classic administrators are assigned the Owner role at the subscription scope.

Only the Azure portal and the new Azure Resource Manager APIs support Azure RBAC. Users and applications that are assigned RBAC roles cannot use the the Azure classic deployment model APIs.

Authorization for management vs. data operations

Azure RBAC only supports management operations of the Azure resources in the Azure portal and Azure Resource Manager APIs. It cannot authorize all data level operations for Azure resources. For example, you can authorize someone to manage Storage Accounts, but not to the blobs or tables within a Storage Account. Similarly, a SQL database can be managed, but not the tables within it.

Next Steps

- Get started with [Role-Based Access Control in the Azure portal](#).
- See the [RBAC built-in roles](#)
- Define your own [Custom roles in Azure RBAC](#)

Apply policies to Windows VMs with Azure Resource Manager

11/15/2017 • 3 min to read • [Edit Online](#)

By using policies, an organization can enforce various conventions and rules throughout the enterprise. Enforcement of the desired behavior can help mitigate risk while contributing to the success of the organization. In this article, we describe how you can use Azure Resource Manager policies to define the desired behavior for your organization's Virtual Machines.

For an introduction to policies, see [What is Azure Policy?](#).

Permitted Virtual Machines

To ensure that virtual machines for your organization are compatible with an application, you can restrict the permitted operating systems. In the following policy example, you allow only Windows Server 2012 R2 Datacenter Virtual Machines to be created:

```
{
  "if": {
    "allOf": [
      {
        "field": "type",
        "in": [
          "Microsoft.Compute/disks",
          "Microsoft.Compute/virtualMachines",
          "Microsoft.Compute/VirtualMachineScaleSets"
        ]
      },
      {
        "not": {
          "allOf": [
            {
              "field": "Microsoft.Compute/imagePublisher",
              "in": [
                "MicrosoftWindowsServer"
              ]
            },
            {
              "field": "Microsoft.Compute/imageOffer",
              "in": [
                "WindowsServer"
              ]
            },
            {
              "field": "Microsoft.Compute/imageSku",
              "in": [
                "2012-R2-Datacenter"
              ]
            },
            {
              "field": "Microsoft.Compute/imageVersion",
              "in": [
                "latest"
              ]
            }
          ]
        }
      }
    ],
    "then": {
      "effect": "deny"
    }
  }
}
```

Use a wild card to modify the preceding policy to allow any Windows Server Datacenter image:

```
{
  "field": "Microsoft.Compute/imageSku",
  "like": "*Datacenter"
}
```

Use anyOf to modify the preceding policy to allow any Windows Server 2012 R2 Datacenter or higher image:

```
{
  "anyOf": [
    {
      "field": "Microsoft.Compute/imageSku",
      "like": "2012-R2-Datacenter*"
    },
    {
      "field": "Microsoft.Compute/imageSku",
      "like": "2016-Datacenter*"
    }
  ]
}
```

For information about policy fields, see [Policy aliases](#).

Managed disks

To require the use of managed disks, use the following policy:

```
{
  "if": {
    "anyOf": [
      {
        "allOf": [
          {
            "field": "type",
            "equals": "Microsoft.Compute/virtualMachines"
          },
          {
            "field": "Microsoft.Compute/virtualMachines/osDisk.uri",
            "exists": true
          }
        ]
      },
      {
        "allOf": [
          {
            "field": "type",
            "equals": "Microsoft.Compute/VirtualMachineScaleSets"
          },
          {
            "anyOf": [
              {
                "field": "Microsoft.Compute/VirtualMachineScaleSets/osDisk.vhdContainers",
                "exists": true
              },
              {
                "field": "Microsoft.Compute/VirtualMachineScaleSets/osdisk.imageUrl",
                "exists": true
              }
            ]
          }
        ]
      }
    ],
    "then": {
      "effect": "deny"
    }
  }
}
```

Images for Virtual Machines

For security reasons, you can require that only approved custom images are deployed in your environment. You can specify either the resource group that contains the approved images, or the specific approved images.

The following example requires images from an approved resource group:

```
{  
  "if": {  
    "allOf": [  
      {  
        "field": "type",  
        "in": [  
          "Microsoft.Compute/virtualMachines",  
          "Microsoft.Compute/VirtualMachineScaleSets"  
        ]  
      },  
      {  
        "not": {  
          "field": "Microsoft.Compute/imageId",  
          "contains": "resourceGroups/CustomImage"  
        }  
      }  
    ],  
    "then": {  
      "effect": "deny"  
    }  
  }  
}
```

The following example specifies the approved image IDs:

```
{  
  "field": "Microsoft.Compute/imageId",  
  "in": ["{imageId1}","{imageId2}"]  
}
```

Virtual Machine extensions

You may want to forbid usage of certain types of extensions. For example, an extension may not be compatible with certain custom virtual machine images. The following example shows how to block a specific extension. It uses publisher and type to determine which extension to block.

```
{
  "if": {
    "allOf": [
      {
        "field": "type",
        "equals": "Microsoft.Compute/virtualMachines/extensions"
      },
      {
        "field": "Microsoft.Compute/virtualMachines/extensions/publisher",
        "equals": "Microsoft.Compute"
      },
      {
        "field": "Microsoft.Compute/virtualMachines/extensions/type",
        "equals": "{extension-type}"
      }
    ]
  },
  "then": {
    "effect": "deny"
  }
}
```

Azure Hybrid Use Benefit

When you have an on-premise license, you can save the license fee on your virtual machines. When you don't have the license, you should forbid the option. The following policy forbids usage of Azure Hybrid Use Benefit (AHUB):

```
{
  "if": {
    "allOf": [
      {
        "field": "type",
        "in": [ "Microsoft.Compute/virtualMachines", "Microsoft.Compute/VirtualMachineScaleSets" ]
      },
      {
        "field": "Microsoft.Compute/licenseType",
        "exists": true
      }
    ]
  },
  "then": {
    "effect": "deny"
  }
}
```

Next steps

- After defining a policy rule (as shown in the preceding examples), you need to create the policy definition and assign it to a scope. The scope can be a subscription, resource group, or resource. To assign policies, see [Use Azure portal to assign and manage resource policies](#), [Use PowerShell to assign policies](#), or [Use Azure CLI to assign policies](#).
- For an introduction to resource policies, see [What is Azure Policy?](#).
- For guidance on how enterprises can use Resource Manager to effectively manage subscriptions, see [Azure enterprise scaffold - prescriptive subscription governance](#).

Set up Key Vault for virtual machines in Azure Resource Manager

6/27/2017 • 1 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using the Resource Manager deployment model, which Microsoft recommends for new deployments instead of the classic deployment model.

In Azure Resource Manager stack, secrets/certificates are modeled as resources that are provided by the resource provider of Key Vault. To learn more about Key Vault, see [What is Azure Key Vault?](#)

NOTE

1. In order for Key Vault to be used with Azure Resource Manager virtual machines, the **EnabledForDeployment** property on Key Vault must be set to true. You can do this in various clients.
2. The Key Vault needs to be created in the same subscription and location as the Virtual Machine.

Use PowerShell to set up Key Vault

To create a key vault by using PowerShell, see [Get started with Azure Key Vault](#).

For new key vaults, you can use this PowerShell cmdlet:

```
New-AzureRmKeyVault -VaultName 'ContosoKeyVault' -ResourceGroupName 'ContosoResourceGroup' -Location 'East Asia' -EnabledForDeployment
```

For existing key vaults, you can use this PowerShell cmdlet:

```
Set-AzureRmKeyVaultAccessPolicy -VaultName 'ContosoKeyVault' -EnabledForDeployment
```

Use CLI to set up Key Vault

To create a key vault by using the command-line interface (CLI), see [Manage Key Vault using CLI](#).

For CLI, you have to create the key vault before you assign the deployment policy. You can do this by using the following command:

```
azure keyvault set-policy ContosoKeyVault --enabled-for-deployment true
```

Use templates to set up Key Vault

While you use a template, you need to set the `enabledForDeployment` property to `true` for the Key Vault resource.

```
{  
  "type": "Microsoft.KeyVault/vaults",  
  "name": "ContosoKeyVault",  
  "apiVersion": "2015-06-01",  
  "location": "<location-of-key-vault>",  
  "properties": {  
    "enabledForDeployment": "true",  
    ....  
    ....  
  }  
}
```

For other options that you can configure when you create a key vault by using templates, see [Create a key vault](#).

Overview of the features in Azure Backup

12/18/2017 • 20 min to read • [Edit Online](#)

Azure Backup is the Azure-based service you can use to back up (or protect) and restore your data in the Microsoft cloud. Azure Backup replaces your existing on-premises or off-site backup solution with a cloud-based solution that is reliable, secure, and cost-competitive. Azure Backup offers multiple components that you download and deploy on the appropriate computer, server, or in the cloud. The component, or agent, that you deploy depends on what you want to protect. All Azure Backup components (no matter whether you're protecting data on-premises or in the cloud) can be used to back up data to a Recovery Services vault in Azure. See the [Azure Backup components table](#) (later in this article) for information about which component to use to protect specific data, applications, or workloads.

[Watch a video overview of Azure Backup](#)

Why use Azure Backup?

Traditional backup solutions have evolved to treat the cloud as an endpoint, or static storage destination, similar to disks or tape. While this approach is simple, it is limited and doesn't take full advantage of an underlying cloud platform, which translates to an expensive, inefficient solution. Other solutions are expensive because you end up paying for the wrong type of storage, or storage that you don't need. Other solutions are often inefficient because they don't offer you the type or amount of storage you need, or administrative tasks require too much time. In contrast, Azure Backup delivers these key benefits:

Automatic storage management - Hybrid environments often require heterogeneous storage - some on-premises and some in the cloud. With Azure Backup, there is no cost for using on-premises storage devices. Azure Backup automatically allocates and manages backup storage, and it uses a pay-as-you-use model. Pay-as-you-use means that you only pay for the storage that you consume. For more information, see the [Azure pricing article](#).

Unlimited scaling - Azure Backup uses the underlying power and unlimited scale of the Azure cloud to deliver high-availability - with no maintenance or monitoring overhead. You can set up alerts to provide information about events, but you don't need to worry about high-availability for your data in the cloud.

Multiple storage options - An aspect of high-availability is storage replication. Azure Backup offers two types of replication: [locally redundant storage](#) and [geo-redundant storage](#). Choose the backup storage option based on need:

- Locally redundant storage (LRS) replicates your data three times (it creates three copies of your data) in a paired datacenter in the same region. LRS is a low-cost option for protecting your data from local hardware failures.
- Geo-redundant storage (GRS) replicates your data to a secondary region (hundreds of miles away from the primary location of the source data). GRS costs more than LRS, but GRS provides a higher level of durability for your data, even if there is a regional outage.

Unlimited data transfer - Azure Backup does not limit the amount of inbound or outbound data you transfer. Azure Backup also does not charge for the data that is transferred. However, if you use the Azure Import/Export service to import large amounts of data, there is a cost associated with inbound data. For more information about this cost, see [Offline-backup workflow in Azure Backup](#). Outbound data refers to data transferred from a Recovery Services vault during a restore operation.

Data encryption - Data encryption allows for secure transmission and storage of your data in the public cloud. You store the encryption passphrase locally, and it is never transmitted or stored in Azure. If it is necessary to

restore any of the data, only you have encryption passphrase, or key.

Application-consistent backup - Whether backing up a file server, virtual machine, or SQL database, you need to know that a recovery point has all required data to restore the backup copy. Azure Backup provides application-consistent backups, which ensure additional fixes are not needed to restore the data. Restoring application consistent data reduces the restoration time, allowing you to quickly return to a running state.

Long-term retention - Instead of switching backup copies from disk to tape and moving the tape to an off-site location, you can use Azure for short-term and long-term retention. Azure doesn't limit the length of time data remains in a Backup or Recovery Services vault. You can keep data in a vault for as long as you like. Azure Backup has a limit of 9999 recovery points per protected instance. See the [Backup and retention](#) section in this article for an explanation of how this limit may impact your backup needs.

Which Azure Backup components should I use?

If you aren't sure which Azure Backup component works for your needs, see the following table for information about what you can protect with each component. The Azure portal provides a wizard, which is built into the portal, to guide you through choosing the component to download and deploy. The wizard, which is part of the Recovery Services vault creation, leads you through the steps for selecting a backup goal, and choosing the data or application to protect.

COMPONENT	BENEFITS	LIMITS	WHAT IS PROTECTED?	WHERE ARE BACKUPS STORED?
Azure Backup (MARS) agent	<ul style="list-style-type: none">• Back up files and folders on physical or virtual Windows OS (VMs can be on-premises or in Azure)• No separate backup server required.	<ul style="list-style-type: none">• Backup 3x per day• Not application aware; file, folder, and volume-level restore only,• No support for Linux.	<ul style="list-style-type: none">• Files,• Folders,• System State	Recovery Services vault
System Center DPM	<ul style="list-style-type: none">• Application-aware snapshots (VSS)• Full flexibility for when to take backups• Recovery granularity (all)• Can use Recovery Services vault• Linux support on Hyper-V and VMware VMs• Back up and restore VMware VMs using DPM 2012 R2	Cannot back up Oracle workload.	<ul style="list-style-type: none">• Files,• Folders,• Volumes,• VMs,• Applications,• Workloads	<ul style="list-style-type: none">• Recovery Services vault,• Locally attached disk,• Tape (on-premises only)

COMPONENT	BENEFITS	LIMITS	WHAT IS PROTECTED?	WHERE ARE BACKUPS STORED?
Azure Backup Server	<ul style="list-style-type: none"> • App aware snapshots (VSS) • Full flexibility for when to take backups • Recovery granularity (all) • Can use Recovery Services vault • Linux support on Hyper-V and VMware VMs • Back up and restore VMware VMs • Does not require a System Center license 	<ul style="list-style-type: none"> • Cannot back up Oracle workload. • Always requires live Azure subscription • No support for tape backup 	<ul style="list-style-type: none"> • Files, • Folders, • Volumes, • VMs, • Applications, • Workloads 	<ul style="list-style-type: none"> • Recovery Services vault, • Locally attached disk
Azure IaaS VM Backup	<ul style="list-style-type: none"> • Native backups for Windows/Linux • No specific agent installation required • Fabric-level backup with no backup infrastructure needed 	<ul style="list-style-type: none"> • Back up VMs once-a-day • Restore VMs only at disk level • Cannot back up on-premises 	<ul style="list-style-type: none"> • VMs, • All disks (using PowerShell) 	Recovery Services vault

What are the deployment scenarios for each component?

COMPONENT	CAN BE DEPLOYED IN AZURE?	CAN BE DEPLOYED ON-PREMISES?	TARGET STORAGE SUPPORTED
Azure Backup (MARS) agent	Yes The Azure Backup agent can be deployed on any Windows Server VM that runs in Azure.	Yes The Backup agent can be deployed on any Windows Server VM or physical machine.	Recovery Services vault
System Center DPM	Yes Learn more about how to protect workloads in Azure by using System Center DPM.	Yes Learn more about how to protect workloads and VMs in your datacenter.	Locally attached disk, Recovery Services vault, tape (on-premises only)
Azure Backup Server	Yes Learn more about how to protect workloads in Azure by using Azure Backup Server.	Yes Learn more about how to protect workloads in Azure by using Azure Backup Server.	Locally attached disk, Recovery Services vault

COMPONENT	CAN BE DEPLOYED IN AZURE?	CAN BE DEPLOYED ON-PREMISES?	TARGET STORAGE SUPPORTED
Azure IaaS VM Backup	Yes Part of Azure fabric Specialized for backup of Azure infrastructure as a service (IaaS) virtual machines .	No Use System Center DPM to back up virtual machines in your datacenter.	Recovery Services vault

Which applications and workloads can be backed up?

The following table provides a matrix of the data and workloads that can be protected using Azure Backup. The Azure Backup solution column has links to the deployment documentation for that solution.

DATA OR WORKLOAD	SOURCE ENVIRONMENT	AZURE BACKUP SOLUTION
Files and folders	Windows Server	Azure Backup agent , System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
Files and folders	Windows computer	Azure Backup agent , System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
Hyper-V virtual machine (Windows)	Windows Server	System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
Hyper-V virtual machine (Linux)	Windows Server	System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
VMware virtual machine	Windows Server	System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)

DATA OR WORKLOAD	SOURCE ENVIRONMENT	AZURE BACKUP SOLUTION
Microsoft SQL Server	Windows Server	System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
Microsoft SharePoint	Windows Server	System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
Microsoft Exchange	Windows Server	System Center DPM (+ the Azure Backup agent), Azure Backup Server (includes the Azure Backup agent)
Azure IaaS VMs (Windows)	running in Azure	Azure Backup (VM extension)
Azure IaaS VMs (Linux)	running in Azure	Azure Backup (VM extension)

Linux support

The following table shows the Azure Backup components that have support for Linux.

COMPONENT	LINUX (AZURE ENDORSED) SUPPORT
Azure Backup (MARS) agent	No (Only Windows based agent)
System Center DPM	<ul style="list-style-type: none"> File-consistent backup of Linux Guest VMs on Hyper-V and VMWare VM restore of Hyper-V and VMWare Linux Guest VMs <i>File-consistent backup not available for Azure VM</i>
Azure Backup Server	<ul style="list-style-type: none"> File-consistent backup of Linux Guest VMs on Hyper-V and VMWare VM restore of Hyper-V and VMWare Linux Guest VMs <i>File-consistent backup not available for Azure VM</i>
Azure IaaS VM Backup	Application-consistent backup using pre-script and post-script framework Granular file recovery Restore all VM disks VM restore

Using Premium Storage VMs with Azure Backup

Azure Backup protects Premium Storage VMs. Azure Premium Storage is solid-state drive (SSD)-based storage designed to support I/O-intensive workloads. Premium Storage is attractive for virtual machine (VM) workloads. For more information about Premium Storage, see the article, [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#).

Back up Premium Storage VMs

While backing up Premium Storage VMs, the Backup service creates a temporary staging location, named "AzureBackup-", in the Premium Storage account. The size of the staging location is equal to the size of the recovery point snapshot. Be sure the Premium Storage account has adequate free space to accommodate the temporary staging location. For more information, see the article, [premium storage limitations](#). Once the backup job finishes, the staging location is deleted. The price of storage used for the staging location is consistent with all [Premium storage pricing](#).

NOTE

Do not modify or edit the staging location.

Restore Premium Storage VMs

Premium Storage VMs can be restored to either Premium Storage or to normal storage. Restoring a Premium Storage VM recovery point back to Premium Storage is the typical process of restoration. However, it can be cost effective to restore a Premium Storage VM recovery point to standard storage. This type of restoration can be used if you need a subset of files from the VM.

Using managed disk VMs with Azure Backup

Azure Backup protects managed disk VMs. Managed disks free you from managing storage accounts of virtual machines and greatly simplify VM provisioning.

Back up managed disk VMs

Backing up VMs on managed disks is no different than backing up Resource Manager VMs. In the Azure portal, you can configure the backup job directly from the Virtual Machine view or from the Recovery Services vault view. You can back up VMs on managed disks through RestorePoint collections built on top of managed disks. Azure Backup also supports backing up managed disk VMs encrypted using Azure Disk encryption(ADE).

Restore managed disk VMs

Azure Backup allows you to restore a complete VM with managed disks, or restore managed disks to a storage account. Azure manages the managed disks during the restore process. You (the customer) manage the storage account created as part of the restore process. When restoring managed encrypted VMs, the VM's keys and secrets should exist in the key vault prior to starting the restore operation.

What are the features of each Backup component?

The following sections provide tables that summarize the availability or support of various features in each Azure Backup component. See the information following each table for additional support or details.

Storage

FEATURE	AZURE BACKUP AGENT	SYSTEM CENTER DPM	AZURE BACKUP SERVER	AZURE IAAS VM BACKUP
Recovery Services vault				
Disk storage				
Tape storage				

FEATURE	AZURE BACKUP AGENT	SYSTEM CENTER DPM	AZURE BACKUP SERVER	AZURE IAAS VM BACKUP
Compression (in Recovery Services vault)	●	●	●	
Incremental backup	●	●	●	●
Disk deduplication		◐	◐	

Key ● = Supported ⚡ = Partially Supported <blank> = Not Supported

The Recovery Services vault is the preferred storage target across all components. System Center DPM and Azure Backup Server also provide the option to have a local disk copy. However, only System Center DPM provides the option to write data to a tape storage device.

Compression

Backups are compressed to reduce the required storage space. The only component that does not use compression is the VM extension. The VM extension copies all backup data from your storage account to the Recovery Services vault in the same region. No compression is used when transferring the data. Transferring the data without compression slightly inflates the storage used. However, storing the data without compression allows for faster restoration, should you need that recovery point.

Disk Deduplication

You can take advantage of deduplication when you deploy System Center DPM or Azure Backup Server [on a Hyper-V virtual machine](#). Windows Server performs data deduplication (at the host level) on virtual hard disks (VHDs) that are attached to the virtual machine as backup storage.

NOTE

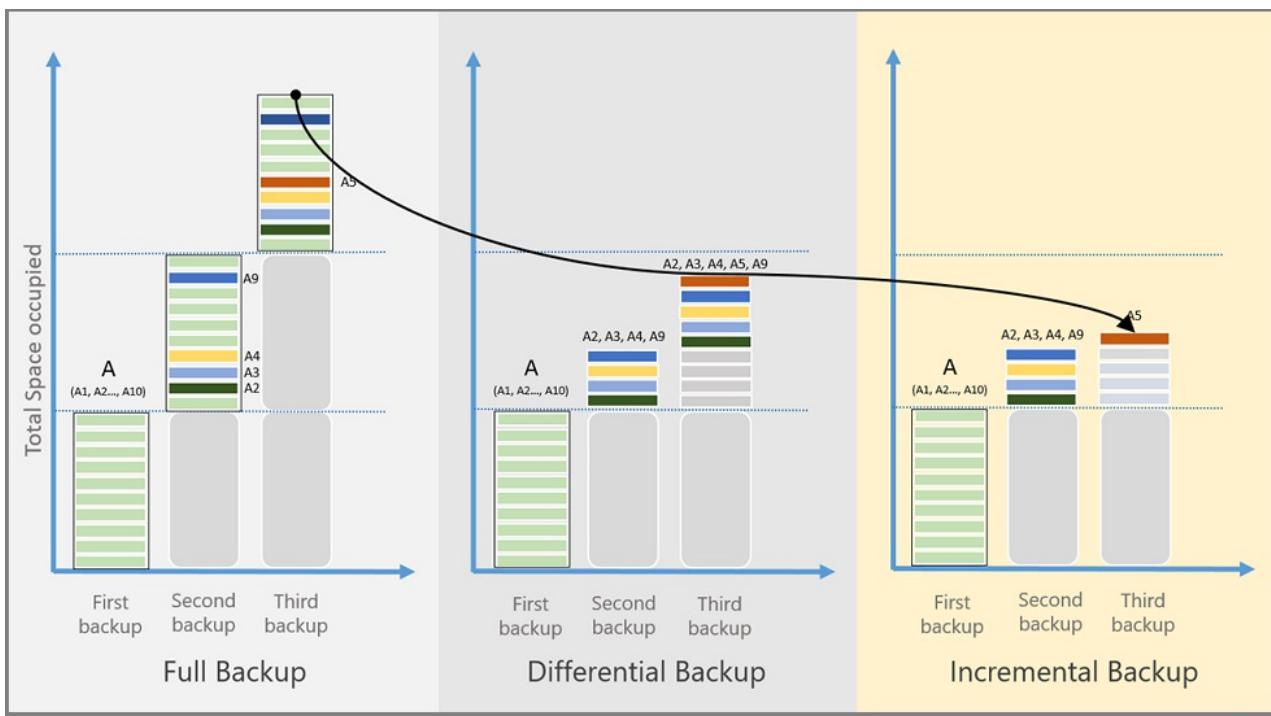
Deduplication is not available in Azure for any Backup component. When System Center DPM and Backup Server are deployed in Azure, the storage disks attached to the VM cannot be deduplicated.

Incremental backup explained

Every Azure Backup component supports incremental backup regardless of the target storage (disk, tape, Recovery Services vault). Incremental backup ensures that backups are storage and time efficient, by transferring only those changes made since the last backup.

Comparing Full, Differential and Incremental backup

Storage consumption, recovery time objective (RTO), and network consumption varies for each type of backup method. To keep the backup total cost of ownership (TCO) down, you need to understand how to choose the best backup solution. The following image compares Full Backup, Differential Backup, and Incremental Backup. In the image, data source A is composed of 10 storage blocks A1-A10, which are backed up monthly. Blocks A2, A3, A4, and A9 change in the first month, and block A5 changes in the next month.



With **Full Backup**, each backup copy contains the entire data source. Full backup consumes a large amount of network bandwidth and storage, each time a backup copy is transferred.

Differential backup stores only the blocks that changed since the initial full backup, which results in a smaller amount of network and storage consumption. Differential backups don't retain redundant copies of unchanged data. However, because the data blocks that remain unchanged between subsequent backups are transferred and stored, differential backups are inefficient. In the second month, changed blocks A2, A3, A4, and A9 are backed up. In the third month, these same blocks are backed up again, along with changed block A5. The changed blocks continue to be backed up until the next full backup happens.

Incremental Backup achieves high storage and network efficiency by storing only the blocks of data that changed since the previous backup. With incremental backup, there is no need to take regular full backups. In the example, after the full backup is taken for the first month, changed blocks A2, A3, A4, and A9 are marked as changed and transferred for the second month. In the third month, only changed block A5 is marked and transferred. Moving less data saves storage and network resources, which decreases TCO.

Security

FEATURE	AZURE BACKUP AGENT	SYSTEM CENTER DPM	AZURE BACKUP SERVER	AZURE IAAS VM BACKUP
Network security (to Azure)	●	●	●	●
Data security (in Azure)	●	●	●	●

Key



= Supported



= Partially Supported

<blank> = Not Supported

Network security

All backup traffic from your servers to the Recovery Services vault is encrypted using Advanced Encryption Standard 256. The backup data is sent over a secure HTTPS link. The backup data is also stored in the Recovery Services vault in encrypted form. Only you, the Azure customer, have the passphrase to unlock this data. Microsoft cannot decrypt the backup data at any point.

WARNING

Once you establish the Recovery Services vault, only you have access to the encryption key. Microsoft never maintains a copy of your encryption key, and does not have access to the key. If the key is misplaced, Microsoft cannot recover the backup data.

Data security

Backing up Azure VMs requires setting up encryption *within* the virtual machine. Use BitLocker on Windows virtual machines and **dm-crypt** on Linux virtual machines. Azure Backup does not automatically encrypt backup data that comes through this path.

Network

FEATURE	AZURE BACKUP AGENT	SYSTEM CENTER DPM	AZURE BACKUP SERVER	AZURE IAAS VM BACKUP
Network compression (to backup server)				
Network compression (to Recovery Services vault)				
Network protocol (to backup server)		TCP	TCP	
Network protocol (to Recovery Services vault)	HTTPS	HTTPS	HTTPS	HTTPS

Key = Supported <blank> = Not Supported

The VM extension (on the IaaS VM) reads the data directly from the Azure storage account over the storage network, so it is not necessary to compress this traffic.

If you use a System Center DPM server or Azure Backup Server as a secondary backup server, compress the data going from the primary server to the backup server. Compressing data before backing it up to DPM or Azure Backup Server, saves bandwidth.

Network Throttling

The Azure Backup agent offers network throttling, which allows you to control how network bandwidth is used during data transfer. Throttling can be helpful if you need to back up data during work hours but do not want the backup process to interfere with other internet traffic. Throttling for data transfer applies to back up and restore activities.

Backup and retention

Azure Backup has a limit of 9999 recovery points, also known as backup copies or snapshots, per *protected instance*. A protected instance is a computer, server (physical or virtual), or workload configured to back up data to Azure. For more information, see the section, [What is a protected instance](#). An instance is protected once a backup copy of data has been saved. The backup copy of data is the protection. If the source data was lost or became corrupt, the backup copy could restore the source data. The following table shows the maximum backup frequency for each component. Your backup policy configuration determines how quickly you consume the recovery points. For example, if you create a recovery point each day, then you can retain recovery points for 27 years before you run out. If you take a monthly recovery point, you can retain recovery points for 833 years before you run out. The

Backup service does not set an expiration time limit on a recovery point.

	AZURE BACKUP AGENT	SYSTEM CENTER DPM	AZURE BACKUP SERVER	AZURE IAAS VM BACKUP
Backup frequency (to Recovery Services vault)	Three backups per day	Two backups per day	Two backups per day	One backup per day
Backup frequency (to disk)	Not applicable	<ul style="list-style-type: none">• Every 15 minutes for SQL Server• Every hour for other workloads	<ul style="list-style-type: none">• Every 15 minutes for SQL Server• Every hour for other workloads	Not applicable
Retention options	Daily, weekly, monthly, yearly	Daily, weekly, monthly, yearly	Daily, weekly, monthly, yearly	Daily, weekly, monthly, yearly
Maximum recovery points per protected instance	9999	9999	9999	9999
Maximum retention period	Depends on backup frequency	Depends on backup frequency	Depends on backup frequency	Depends on backup frequency
Recovery points on local disk	Not applicable	<ul style="list-style-type: none">• 64 for File Servers,• 448 for Application Servers	<ul style="list-style-type: none">• 64 for File Servers,• 448 for Application Servers	Not applicable
Recovery points on tape	Not applicable	Unlimited	Not applicable	Not applicable

What is a protected instance

A protected instance is a generic reference to a Windows computer, a server (physical or virtual), or SQL database that has been configured to back up to Azure. An instance is protected once you configure a backup policy for the computer, server, or database, and create a backup copy of the data. Subsequent copies of the backup data for that protected instance (which are called recovery points), increase the amount of storage consumed. You can create up to 9999 recovery points for a protected instance. If you delete a recovery point from storage, it does not count against the 9999 recovery point total. Some common examples of protected instances are virtual machines, application servers, databases, and personal computers running the Windows operating system. For example:

- A virtual machine running the Hyper-V or Azure IaaS hypervisor fabric. The guest operating systems for the virtual machine can be Windows Server or Linux.
- An application server: The application server can be a physical or virtual machine running Windows Server and workloads with data that needs to be backed up. Common workloads are Microsoft SQL Server, Microsoft Exchange server, Microsoft SharePoint server, and the File Server role on Windows Server. To back up these workloads you need System Center Data Protection Manager (DPM) or Azure Backup Server.
- A personal computer, workstation, or laptop running the Windows operating system.

What is a Recovery Services vault?

A Recovery Services vault is an online storage entity in Azure used to hold data such as backup copies, recovery points, and backup policies. You can use Recovery Services vaults to hold backup data for Azure services and on-premises servers and workstations. Recovery Services vaults make it easy to organize your backup data, while minimizing management overhead. You can create as many Recovery Services vaults as you like, within a

subscription.

Backup vaults, which are based on Azure Service Manager, were the first version of the vault. Recovery Services vaults, which add the Azure Resource Manager model features, are the second version of the vault. See the [Recovery Services vault overview article](#) for a full description of the feature differences. You can no longer create use the portal to create Backup vaults, but Backup vaults are still supported. You must use the Azure portal to manage your Backup vaults.

IMPORTANT

You can upgrade your Backup vaults to Recovery Services vaults. For details, see the article [Upgrade a Backup vault to a Recovery Services vault](#). Microsoft encourages you to upgrade your Backup vaults to Recovery Services vaults.

After **November 30, 2017**, you will no longer be able to use PowerShell to create Backup vaults.

By November 30, 2017:

- All remaining Backup vaults will be automatically upgraded to Recovery Services vaults.
- You won't be able to access backup data in the Classic portal. Instead, use the Azure portal to access your backup data in Recovery Services vaults.

How does Azure Backup differ from Azure Site Recovery?

Azure Backup and Azure Site Recovery are related in that both services back up data and can restore that data. However, these services serve different purposes in providing business continuity and disaster recovery in your business. Use Azure Backup to protect and restore data at a more granular level. For example, if a presentation on a laptop became corrupted, you would use Azure Backup to restore the presentation. If you wanted to replicate the configuration and data on a VM across another datacenter, use Azure Site Recovery.

Azure Backup protects data on-premises and in the cloud. Azure Site Recovery coordinates virtual-machine and physical-server replication, failover, and fallback. Both services are important because your disaster recovery solution needs to keep your data safe and recoverable (Backup) *and* keep your workloads available (Site Recovery) when outages occur.

The following concepts can help you make important decisions around backup and disaster recovery.

CONCEPT	DETAILS	BACKUP	DISASTER RECOVERY (DR)
Recovery point objective (RPO)	The amount of acceptable data loss if a recovery needs to be done.	Backup solutions have wide variability in their acceptable RPO. Virtual machine backups usually have an RPO of one day, while database backups have RPOs as low as 15 minutes.	Disaster recovery solutions have low RPOs. The DR copy can be behind by a few seconds or a few minutes.
Recovery time objective (RTO)	The amount of time that it takes to complete a recovery or restore.	Because of the larger RPO, the amount of data that a backup solution needs to process is typically much higher, which leads to longer RTOs. For example, it can take days to restore data from tapes, depending on the time it takes to transport the tape from an off-site location.	Disaster recovery solutions have smaller RTOs because they are more in sync with the source. Fewer changes need to be processed.

CONCEPT	DETAILS	BACKUP	DISASTER RECOVERY (DR)
Retention	How long data needs to be stored	For scenarios that require operational recovery (data corruption, inadvertent file deletion, OS failure), backup data is typically retained for 30 days or less. From a compliance standpoint, data might need to be stored for months or even years. Backup data is ideally suited for archiving in such cases.	Disaster recovery needs only operational recovery data, which typically takes a few hours or up to a day. Because of the fine-grained data capture used in DR solutions, using DR data for long-term retention is not recommended.

Next steps

Use one of the following tutorials for detailed, step-by-step, instructions for protecting data on Windows Server, or protecting a virtual machine (VM) in Azure:

- [Back up Files and Folders](#)
- [Backup Azure Virtual Machines](#)

For details about protecting other workloads, try one of these articles:

- [Back up your Windows Server](#)
- [Back up application workloads](#)
- [Backup Azure IaaS VMs](#)

Back up a virtual machine in Azure

12/19/2017 • 3 min to read • [Edit Online](#)

Azure backups can be created through the Azure portal. This method provides a browser-based user interface to create and configure Azure backups and all related resources. You can protect your data by taking backups at regular intervals. Azure Backup creates recovery points that can be stored in geo-redundant recovery vaults. This article details how to back up a virtual machine (VM) with the Azure portal.

This quickstart enables backup on an existing Azure VM. If you need to create a VM, you can [create a VM with the Azure portal](#).

Log in to Azure

Log in to the Azure portal at <http://portal.azure.com>.

Select a VM to back up

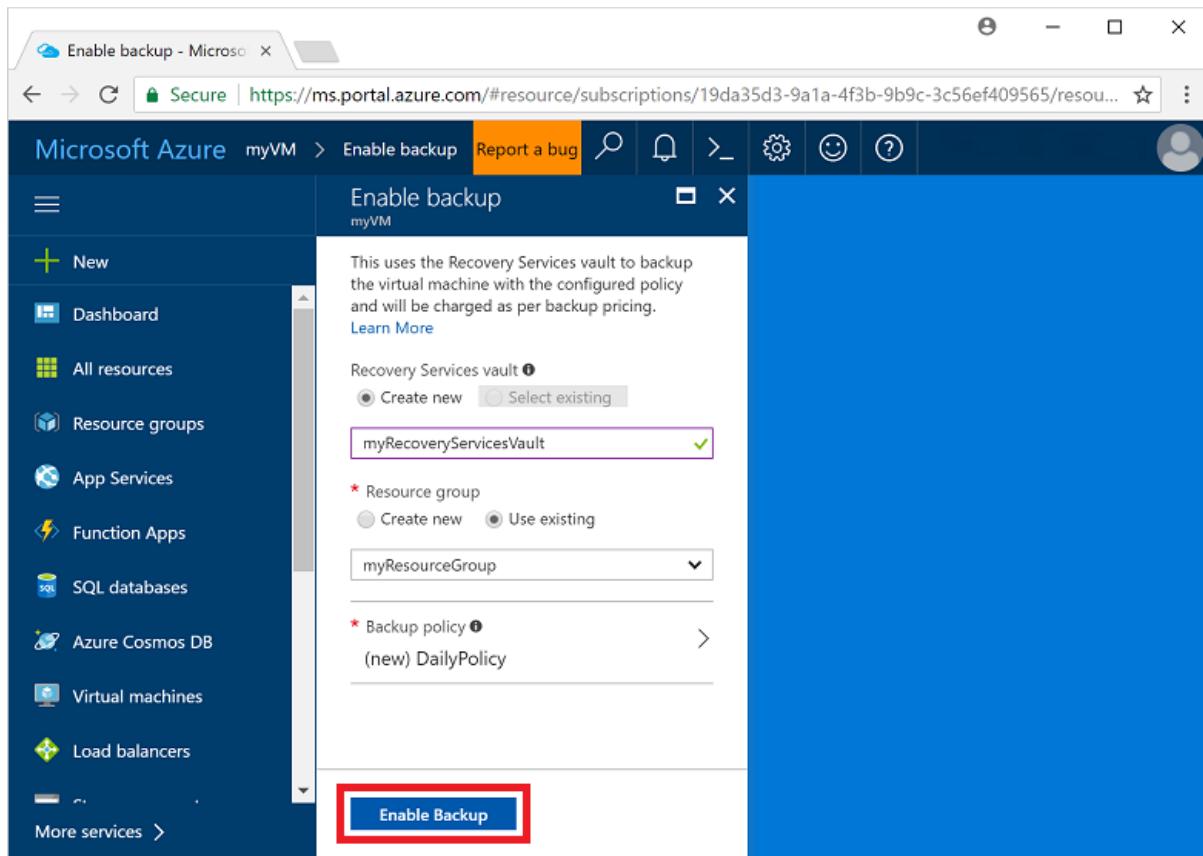
Create a simple scheduled daily backup to a Recovery Services Vault.

1. In the menu on the left, select **Virtual machines**.
2. From the list, choose a VM to back up. If you used the sample VM quickstart commands, the VM is named *myVM* in the *myResourceGroup* resource group.
3. In the **Operations** section, choose **Backup**. The **Enable backup** window opens.

Enable backup on a VM

A Recovery Services vault is a logical container that stores the backup data for each protected resource, such as Azure VMs. When the backup job for a protected resource runs, it creates a recovery point inside the Recovery Services vault. You can then use one of these recovery points to restore data to a given point in time.

1. Select **Create new** and provide a name for the new vault, such as *myRecoveryServicesVault*.
2. If not already selected, choose **Use existing**, then select the resource group of your VM from the drop-down menu.



By default, the vault is set for Geo-Redundant storage. To further protect your data, this storage redundancy level ensures that your backup data is replicated to a secondary Azure region that is hundreds of miles away from the primary region.

You create and use policies to define when a backup job runs and how long the recovery points are stored. The default protection policy runs a backup job each day and retains recovery points for 30 days. You can use these default policy values to quickly protect your VM.

3. To accept the default backup policy values, select **Enable Backup**.

It takes a few moments to create the Recovery Services vault.

Start a backup job

You can start a backup now rather than wait for the default policy to run the job at the scheduled time. This first backup job creates a full recovery point. Each backup job after this initial backup creates incremental recovery points. Incremental recovery points are storage and time-efficient, as they only transfer changes made since the last backup.

1. On the **Backup** window for your VM, select **Backup now**.

The screenshot shows the Azure portal interface for a virtual machine named 'myVM'. On the left, there's a navigation menu with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Networking, Disks, Size, Availability set, Configuration, and a highlighted 'Backup' option. The main content area has tabs for Settings, Backup now, Restore VM, File Recovery, and Stop backup. Under the 'Backup now' tab, there's a section for 'Essentials' with details about the Recovery services vault ('myRecoveryServicesVault'), Subscription name ('Visual Studio Enterprise'), Subscription ID, Item type (Azure virtual machine), and Backup policy ('DailyPolicy'). Below this is a 'Restore points' section showing 'Last 30 days' and 'Last 7 days' both with a count of 0.

2. To accept the backup retention policy of 30 days, leave the default **Retain Backup Till** date. To start the job, select **Backup**.

Monitor the backup job

In the **Backup** window for your VM, the status of the backup and number of completed restore points are shown. Once the VM backup job is complete, information on the **Last backup time**, **Latest restore point**, and **Oldest restore point** is shown on the right-hand side of the **Overview** window.

Clean up deployment

When no longer needed, you can disable protection on the VM, remove the restore points and Recovery Services vault, then delete the resource group and associated VM resources

If you are going to continue on to a Backup tutorial that explains how to restore data for your VM, skip the steps in this section and go to [Next steps](#).

1. Select the **Backup** option for your VM.
2. Select ...**More** to show additional options, then choose **Stop backup**.

The screenshot shows the Azure Recovery Services vault overview page. At the top, there are navigation links: Settings, Backup now, Restore VM, File Recovery, and More (which is highlighted with a red box). Below this, there's a section titled 'Essentials' with a dropdown arrow. To the right of this section is a context menu with the following items:

- Stop backup
- Last backup: 9/18/2017
- Resume backup
- Latest restore: 9/18/2017, 9:01:00 PM (16 minute(s) ago)
- Delete backup data (highlighted with a red box)
- Oldest restore point: 9/18/2017, 9:01:00 PM (16 minute(s) ago)
- Backup policy: DailyPolicy
- Backup Pre-Check: Passed

On the left side of the main content area, there are several details listed:

- Recovery services vault: myRecoveryServicesVault
- Subscription name: Visual Studio Enterprise
- Subscription ID
- Item type: Azure virtual machine
- Last backup status: Success

3. Select **Delete Backup Data** from the drop-down menu.
4. In the **Type the name of the Backup item** dialog, enter your VM name, such as *myVM*. Select **Stop Backup**

Once the VM backup has been stopped and recovery points removed, you can delete the resource group. If you used an existing VM, you may wish to leave the resource group and VM in place.

5. In the menu on the left, select **Resource groups**.
6. From the list, choose your resource group. If you used the sample VM quickstart commands, the resource group is named *myResourceGroup*.
7. Select **Delete resource group**. To confirm, enter the resource group name, then select **Delete**.

The screenshot shows the Azure Resource Groups blade for the 'myResourceGroup'. At the top, there's a search bar and a 'Delete resource group' button (highlighted with a red box). Below this, there's a sidebar with links: Overview (which is selected and highlighted with a blue background), Activity log, Access control (IAM), and Tags. The main content area shows the following details:

- Subscription name (change): Visual Studio Enterprise
- Deployment: 2 Succeeded
- Subscription ID

At the bottom of the blade, there are filter options: Filter by name..., All types, All locations, and No grouping.

Next steps

In this quickstart, you created a Recovery Services vault, enabled protection on a VM, and created the initial recovery point. To learn more about Azure Backup and Recovery Services, continue to the tutorials.

[Back up multiple Azure VMs](#)

Back up a virtual machine in Azure with PowerShell

12/19/2017 • 4 min to read • [Edit Online](#)

The Azure PowerShell module is used to create and manage Azure resources from the command line or in scripts. You can protect your data by taking backups at regular intervals. Azure Backup creates recovery points that can be stored in geo-redundant recovery vaults. This article details how to back up a virtual machine (VM) with the Azure PowerShell module. You can also perform these steps with the [Azure CLI](#) or [Azure portal](#).

This quickstart enables backup on an existing Azure VM. If you need to create a VM, you can [create a VM with Azure PowerShell](#).

This quickstart requires the Azure PowerShell module version 4.4 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to install or upgrade, see [Install Azure PowerShell module](#).

Log in to Azure

Log in to your Azure subscription with the `Login-AzureRmAccount` command and follow the on-screen directions.

```
Login-AzureRmAccount
```

The first time you use Azure Backup, you must register the Azure Recovery Service provider in your subscription with [Register-AzureRmResourceProvider](#).

```
Register-AzureRmResourceProvider -ProviderNamespace "Microsoft.RecoveryServices"
```

Create a recovery services vault

A Recovery Services vault is a logical container that stores the backup data for each protected resource, such as Azure VMs. When the backup job for a protected resource runs, it creates a recovery point inside the Recovery Services vault. You can then use one of these recovery points to restore data to a given point in time.

Create a Recovery Services vault with [New-AzureRmRecoveryServicesVault](#). Specify the same resource group and location as the VM you wish to protect. If you used the [sample script](#) to create your VM, the resource group is named *myResourceGroup*, the VM is named *myVM*, and the resources are in the *WestEurope* location.

```
New-AzureRmRecoveryServicesVault `  
-ResourceGroupName "myResourceGroup" `  
-Name "myRecoveryServicesVault" `  
-Location "WestEurope"
```

By default, the vault is set for Geo-Redundant storage. To further protect your data, this storage redundancy level ensures that your backup data is replicated to a secondary Azure region that is hundreds of miles away from the primary region.

To use this vault with the remaining steps, set the vault context with [Set-AzureRmRecoveryServicesVaultContext](#)

```
Get-AzureRmRecoveryServicesVault `  
-Name "myRecoveryServicesVault" | Set-AzureRmRecoveryServicesVaultContext
```

Enable backup for an Azure VM

You create and use policies to define when a backup job runs and how long the recovery points are stored. The default protection policy runs a backup job each day, and retains recovery points for 30 days. You can use these default policy values to quickly protect your VM. First, set the default policy with [Get-AzureRmRecoveryServicesBackupProtectionPolicy](#):

```
$policy = Get-AzureRmRecoveryServicesBackupProtectionPolicy -Name "DefaultPolicy"
```

To enable backup protection for a VM, use [Enable-AzureRmRecoveryServicesBackupProtection](#). Specify the policy to use, then the resource group and VM to protect:

```
Enable-AzureRmRecoveryServicesBackupProtection `  
-ResourceGroupName "myResourceGroup" `  
-Name "myVM" `  
-Policy $policy
```

Start a backup job

To start a backup now rather than wait for the default policy to run the job at the scheduled time, use [Backup-AzureRmRecoveryServicesBackupItem](#). This first backup job creates a full recovery point. Each backup job after this initial backup creates incremental recovery points. Incremental recovery points are storage and time-efficient, as they only transfer changes made since the last backup.

In the following set of commands, you specify a container in the Recovery Services vault that holds your backup data with [Get-AzureRmRecoveryServicesBackupContainer](#). Each VM to back up is treated as an item. To start a backup job, obtain information on your VM item with [Get-AzureRmRecoveryServicesBackupItem](#).

```
$backupcontainer = Get-AzureRmRecoveryServicesBackupContainer `  
-ContainerType "AzureVM" `  
-FriendlyName "myVM"  
  
$item = Get-AzureRmRecoveryServicesBackupItem `  
-Container $backupcontainer `  
-WorkloadType "AzureVM"  
  
Backup-AzureRmRecoveryServicesBackupItem -Item $item
```

As this first backup job creates a full recovery point, the process can take up to 20 minutes.

Monitor the backup job

To monitor the status of backup jobs, use [Get-AzureRmRecoveryServicesBackupJob](#):

```
Get-AzureRmRecoveryServicesBackupJob
```

The output is similar to the following example, which shows the backup job is **InProgress**:

WorkloadName	Operation	Status	StartTime	EndTime	JobID
myvm	Backup	InProgress	9/18/2017 9:38:02 PM		9f9e8f14
myvm	ConfigureBackup	Completed	9/18/2017 9:33:18 PM	9/18/2017 9:33:51 PM	fe79c739

When the *Status* of the backup job reports *Completed*, your VM is protected with Recovery Services and has a full

recovery point stored.

Clean up deployment

When no longer needed, you can disable protection on the VM, remove the restore points and Recovery Services vault, then delete the resource group and associated VM resources. If you used an existing VM, you can skip the final [Remove-AzureRmResourceGroup](#) cmdlet to leave the resource group and VM in place.

If you are going to continue on to a Backup tutorial that explains how to restore data for your VM, skip the steps in this section and go to [Next steps](#).

```
Disable-AzureRmRecoveryServicesBackupProtection -Item $item -RemoveRecoveryPoints  
$vault = Get-AzureRmRecoveryServicesVault -Name "myRecoveryServicesVault"  
Remove-AzureRmRecoveryServicesVault -Vault $vault  
Remove-AzureRmResourceGroup -Name "myResourceGroup"
```

Next steps

In this quickstart, you created a Recovery Services vault, enabled protection on a VM, and created the initial recovery point. To learn more about Azure Backup and Recovery Services, continue to the tutorials.

[Back up multiple Azure VMs](#)

Use Azure portal to back up multiple virtual machines

12/4/2017 • 6 min to read • [Edit Online](#)

When you back up data in Azure, you store that data in an Azure resource called a Recovery Services vault. The Recovery Services vault resource is available from the Settings menu of most Azure services. The benefit of having the Recovery Services vault integrated into the Settings menu of most Azure services makes it very easy to back up data. However, individually working with each database or virtual machine in your business is tedious. What if you want to back up the data for all virtual machines in one department, or in one location? It is easy to back up multiple virtual machines by creating a backup policy and applying that policy to the desired virtual machines. This tutorial explains how to:

- Create a Recovery Services vault
- Define a backup policy
- Apply the backup policy to protect multiple virtual machines
- Trigger an on-demand backup job for the protected virtual machines

Log in to the Azure portal

Log in to the [Azure portal](#).

Create a Recovery Services vault

The Recovery Services vault contains the backup data, and the backup policy applied to the protected virtual machines. Backing up virtual machines is a local process. You cannot back up a virtual machine from one location to a Recovery Services vault in another location. So, for each Azure location that has virtual machines to be backed up, at least one Recovery Services vault must exist in that location.

1. On the left-hand menu, select **More services** and in the services list, type *Recovery Services*. As you type, the list of resources filters. When you see Recovery Services vaults in the list, select it to open the Recovery Services vaults menu.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with various service icons and names: New, Dashboard, Resource groups, All resources, Recent, App Services, Virtual machines (classic), Virtual machines, SQL databases, Cloud services (classic), Subscriptions, Azure Active Directory, Monitor, Security Center, and Billing. Below the sidebar is a button labeled 'More services >'. At the top center, there's a search bar with the placeholder 'Shift+Space to toggle favorites' and a red box around the word 'Reco'. To the right of the search bar are several icons: Report a bug, a magnifying glass, a bell, a gear, a smiley face, and a question mark. The main content area has a title 'Microsoft Azure' and a sub-section titled 'Reco' with a red box around it. It lists 'Advisor', 'Cognitive Services', 'Recovery Services vaults' (which has a red box around it), and 'Site recovery vaults (classic)'. On the far right, there's a 'Help + support' section with a blue icon.

2. In the **Recovery Services vaults** menu, click **Add** to open the Recovery Services vault menu.

The screenshot shows the 'Recovery Services vaults' blade. On the left, there's a sidebar with icons for New, Dashboard, Resource groups, All resources, Recent, App Services, Virtual machines (classic), Virtual machines, SQL databases, Cloud services (classic), Subscriptions, Azure Active Directory, Monitor, Security Center, and Billing. Below the sidebar is a button labeled 'More services >'. In the center, there's a 'Recovery Services vaults' section with a 'Microsoft' logo. A red box highlights the '+ Add' button. To its right is a 'Assign Tags' button and a 'More' button. Below these are sections for 'Subscriptions: SubscriptionID' and '0 items'. On the right, a 'Recovery Services vault' configuration dialog is open. It includes fields for 'Name' (set to 'myRecoveryServicesVault'), 'Subscription' (set to 'SubscriptionID'), 'Resource group' (radio button set to 'Use existing' with 'myResourceGroup' selected), 'Location' (set to 'West Europe'), and a 'Create' button (which is also highlighted with a red box). There's also a 'Pin to dashboard' checkbox and an 'Automation options' link.

3. In the Recovery Services vault menu,

- Type *myRecoveryServicesVault* in **Name**,
- The current subscription ID appears in **Subscription**. If you have additional subscriptions, you could choose another subscription for the new vault.
- For **Resource group** select **Use existing** and choose *myResourceGroup*. If *myResourceGroup* doesn't exist, select **Create new** and type *myResourceGroup*.

- From the **Location** drop-down menu, choose *West Europe*.
- Click **Create** to create your Recovery Services vault.

A Recovery Services vault must be in the same location as the virtual machines being protected. If you have virtual machines in multiple regions, create a Recovery Services vault in each region. This tutorial creates a Recovery Services vault in *West Europe* because that is where *myVM* (the virtual machine created with the quickstart) was created.

It can take several minutes for the Recovery Services vault to be created. Monitor the status notifications in the upper right-hand area of the portal. Once your vault is created, it appears in the list of Recovery Services vaults.

When you create a Recovery Services vault, by default the vault has geo-redundant storage. To provide data resiliency, geo-redundant storage replicates the data multiple times across two Azure regions.

Set backup policy to protect VMs

After creating the Recovery Services vault, the next step is to configure the vault for the type of data, and to set the backup policy. Backup policy is the schedule for how often and when recovery points are taken. Policy also includes the retention range for the recovery points. For this tutorial let's assume your business is a sports complex with a hotel, stadium, and restaurants and concessions, and you are protecting the data on the virtual machines. The following steps create a backup policy for the financial data.

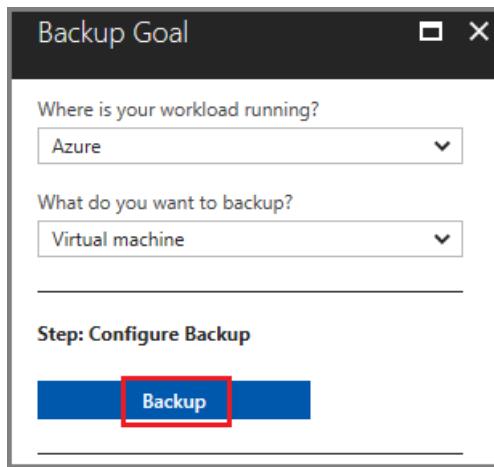
- From the list of Recovery Services vaults, select **myRecoveryServicesVault** to open its dashboard.

The screenshot shows two windows side-by-side. The left window is titled 'Recovery Services vaults' and lists two vaults: 'myRecoveryServicesVault' (selected and highlighted with a red box) and 'Contoso-vault'. The right window is titled 'myRecoveryServicesVault' and contains the following details:

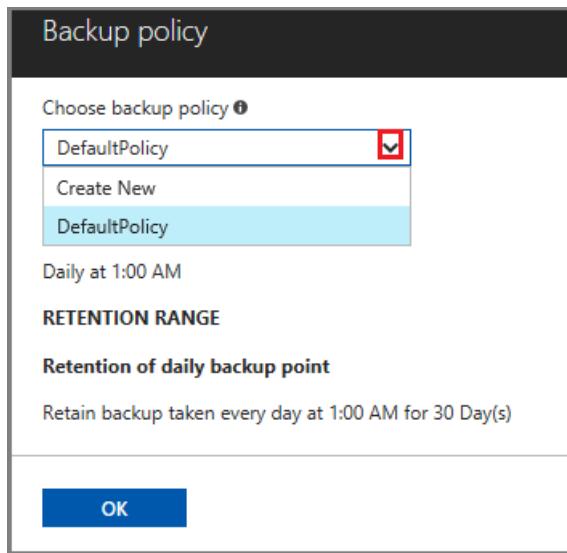
- Essentials:**
 - Resource group (change): myResourceGroup
 - Status: Active
 - Location: West Europe
 - Subscription name (change): MSDNOnDallas
 - Subscription ID: f7f09258-6753-4ca2-b1ae-193798e2c9d8
- Monitoring:**
 - Backup Alerts (last 24...): Critical 0, Warning 0
 - Backup Pre-Check Status (Azure VMs): CRITICAL 0, WARNING 0

- On the vault dashboard menu, click **Backup** to open the Backup menu.
- On the Backup Goal menu, in the **Where is your workload running** drop-down menu, choose *Azure*. From the **What do you want to backup** drop-down, choose *Virtual machine*, and click **Backup**.

These actions prepare the Recovery Services vault for interacting with a virtual machine. Recovery Services vaults have a default policy that creates a restore point each day, and retains the restore points for 30 days.

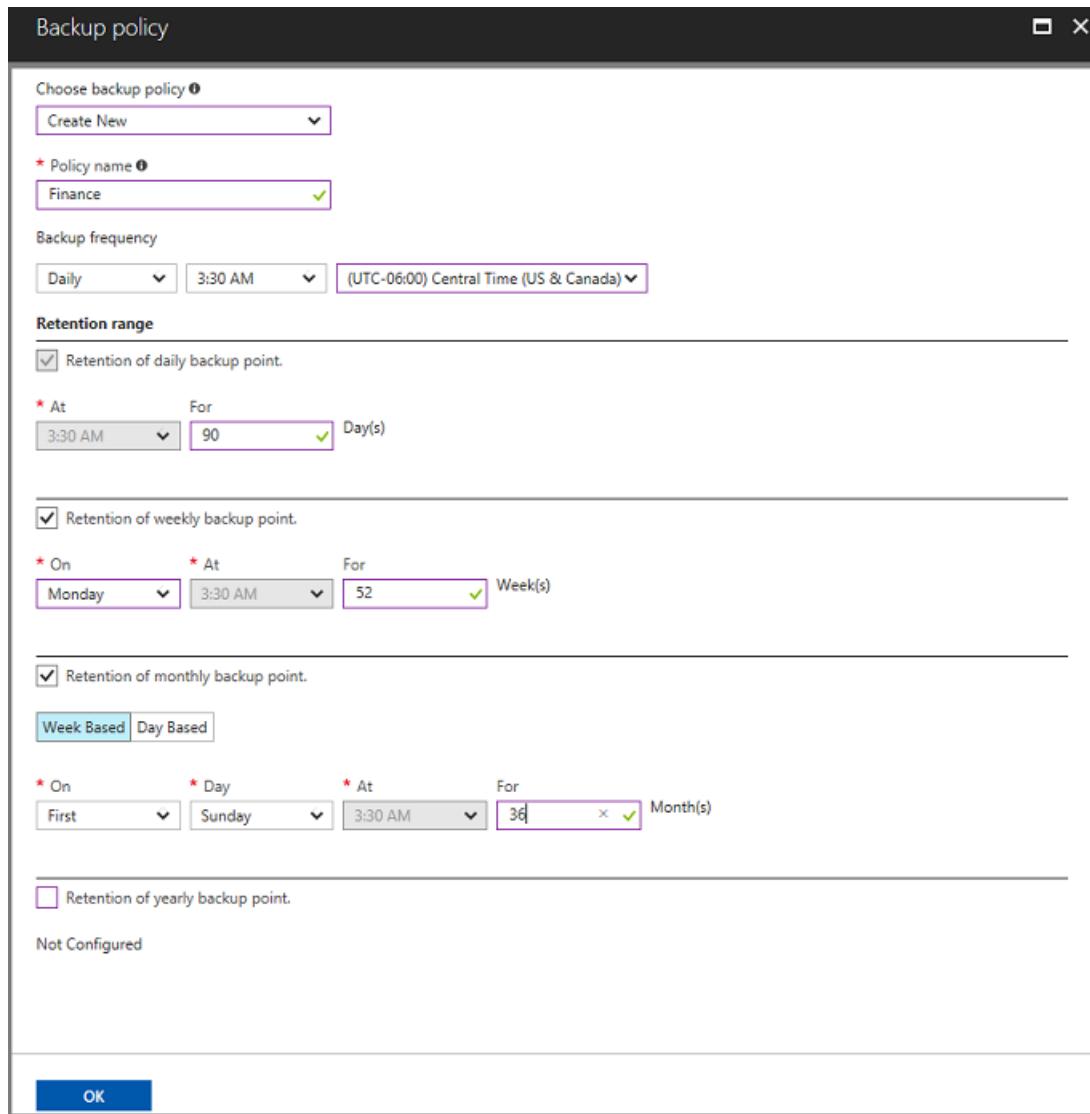


4. To create a new policy, on the Backup policy menu, from the **Choose backup policy** drop-down menu, select *Create New*.



5. In the **Backup policy** menu, for **Policy Name** type *Finance*. Enter the following changes for the Backup policy:

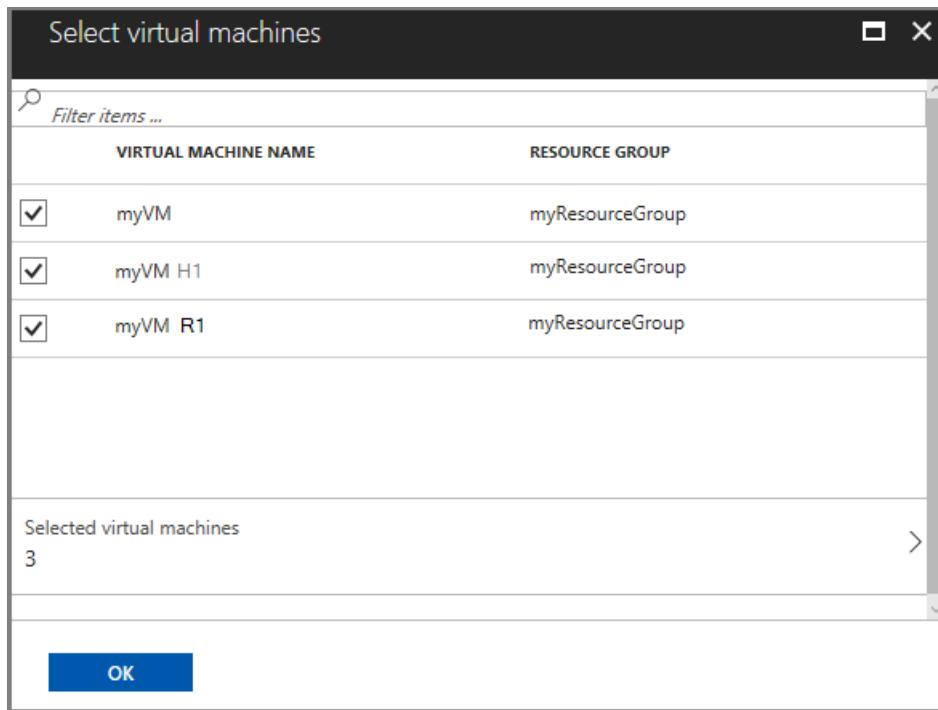
- For **Backup frequency** set the timezone for *Central Time*. Since the sports complex is in Texas, the owner wants the timing to be local. Leave the backup frequency set to Daily at 3:30AM.
- For **Retention of daily backup point**, set the period to 90 days.
- For **Retention of weekly backup point**, use the *Monday* restore point and retain it for 52 weeks.
- For **Retention of monthly backup point**, use the restore point from First Sunday of the month, and retain it for 36 months.
- Deselect the **Retention of yearly backup point** option. The leader of Finance doesn't want to keep data longer than 36 months.
- Click **OK** to create the backup policy.



After creating the backup policy, associate the policy with the virtual machines.

6. In the **Select virtual machines** dialog select *myVM* and click **OK** to deploy the backup policy to the virtual machines.

All virtual machines that are in the same location, and are not already associated with a backup policy, appear. *myVMH1* and *myVMR1* are selected to be associated with the *Finance* policy.



When the deployment completes, you receive a notification that deployment successfully completed.

Initial backup

You have enabled backup for the Recovery Services vaults, but an initial backup has not been created. It is a disaster recovery best practice to trigger the first backup, so that your data is protected.

To run an on-demand backup job:

1. On the vault dashboard, click **3** under **Backup Items**, to open the Backup Items menu.

The **Backup Items** menu opens.

2. On the **Backup Items** menu, click **Azure Virtual Machine** to open the list of virtual machines associated with the vault.

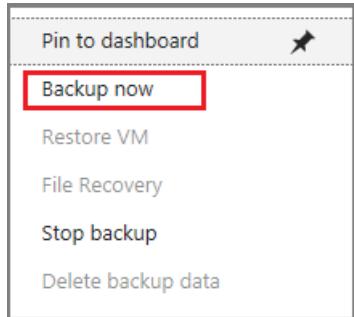
BACKUP MANAGEMENT TYPE	BACKUP ITEM COUNT
Azure Virtual Machine	3
Azure Backup Agent	0
Azure Backup Server	0

The **Backup Items** list opens.

NAME	RESOURCE GROUP	BACKUP PRE-CHECK	LAST BACKUP STATUS	LATEST RESTORE POI	Context menu
myVM	myResourceGroup	Passed	Warning(Initial backu...)		...
buntu	rasquill-security	Passed	Warning(Initial backu...)		...
ops	rhelfiles	Passed	Warning(Initial backu...)		...

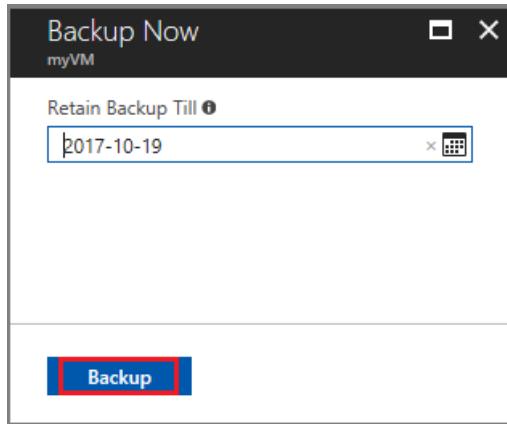
3. On the **Backup Items** list, click the ellipses **...** to open the Context menu.

4. On the Context menu, select **Backup now**.



The Backup Now menu opens.

5. On the Backup Now menu, enter the last day to retain the recovery point, and click **Backup**.



Deployment notifications let you know the backup job has been triggered, and that you can monitor the progress of the job on the Backup jobs page. Depending on the size of your virtual machine, creating the initial backup may take a while.

When the initial backup job completes, you can see its status in the Backup job menu. The on-demand backup job created the initial restore point for *myVM*. If you want to back up other virtual machines, repeat these steps for each virtual machine.

NAME	RESOURCE GROUP	BACKUP PRE-CHECK	LAST BACKUP STATUS	LATEST RESTORE POINT	...
myVM	myResourceGroup	Passed	Success	9/19/2017 6:52:32 PM	...

Clean up resources

If you plan to continue on to work with subsequent tutorials, do not clean up the resources created in this tutorial. If you do not plan to continue, use the following steps to delete all resources created by this tutorial in the Azure portal.

1. On the **myRecoveryServicesVault** dashboard, click **3** under **Backup Items**, to open the Backup Items menu.

myRecoveryServicesVault
Recovery Services vault

Search (Ctrl+ /)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

SETTINGS

Properties

Locks

Automation script

GETTING STARTED

Backup

Site Recovery

MONITORING AND REPORTS

Jobs

Alerts and Events

Backup Reports

Backup items

3

Backup management servers

0

Replicated items

0

Resource group (change)
myResourceGroup

Status

Active

Location

West Europe

Subscription name (change)
subscriptionID

Subscription ID

subscription number

Backup Alerts (last 24...)

Critical 0

Warning 0

Backup Pre-Check Status (Azure VMs)

CRITICAL 0

WARNING 0

0 TOTAL

Site Recovery Health

Unhealthy serv... 0

Events 0

Updates availa... 0

- On the **Backup Items** menu, click **Azure Virtual Machine** to open the list of virtual machines associated with the vault.

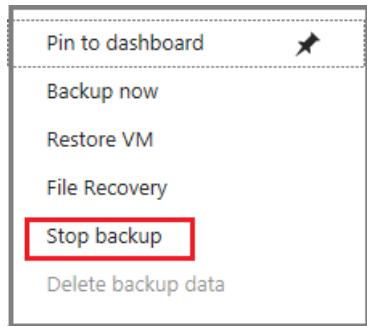
BACKUP MANAGEMENT TYPE	BACKUP ITEM COUNT
Azure Virtual Machine	3
Azure Backup Agent	0
Azure Backup Server	0

The **Backup Items** list opens.

- In the **Backup Items** menu, click the ellipsis to open the Context menu.

Backup Items (Azure Virtual Machine)					
myRecoveryServicesVault					
Actions		Details			
Refresh		Add Filter			
NAME	RESOURCE GROUP	BACKUP PRE-CHECK	LAST BACKUP STATUS	LATEST RESTORE POINT	
myVM	myResourceGroup	Passed	Success	9/19/2017 6:22:37 PM	...
myVM H1	myResourceGroup	Passed	Success	9/19/2017 6:32:35 PM	...
myVM R1	myResourceGroup	Passed	Success	9/19/2017 6:56:17 PM	...

4. On the context menu select **Stop backup** to open Stop Backup menu.



5. In the **Stop Backup** menu, select the upper drop-down menu and choose **Delete Backup Data**.
6. In the **Type the name of the Backup item** dialog, type *myVM*.
7. Once the backup item is verified (a checkmark appears), **Stop backup** button is enabled. Click **Stop Backup** to stop the policy and delete the restore points.

Stop Backup

myVM

Delete Backup Data

This option will stop all scheduled backup jobs, deletes backup data and can't be undone.

* Type the name of Backup Item
myVM

Reason (optional)
Others

Comments

Stop backup

8. In the **myRecoveryServicesVault** menu, click **Delete**.

The screenshot shows the Azure Recovery Services vault interface. The left sidebar includes 'Overview', 'Activity log', 'Access control (IAM)', and 'Tags'. The main area has tabs for 'Backup', 'Replicate', and 'Delete' (which is highlighted with a red box). A survey prompt is displayed above the 'Essentials' section. The 'Essentials' section provides summary information about the vault's resource group, status, and backup management.

Resource group (change)	Backup items
myResourceGroup	0

Status	Backup management servers
Active	0

Once the vault is deleted, you return to the list of Recovery Services vaults.

Next steps

In this tutorial you used the Azure portal to:

- Create a Recovery Services vault
- Set the vault to protect virtual machines
- Create a custom backup and retention policy
- Assign the policy to protect multiple virtual machines
- Trigger an on-demand back up for virtual machines

Continue to the next tutorial to restore an Azure virtual machine from disk.

[Restore VMs using CLI](#)

Prepay for Virtual Machines with Reserved VM Instances

11/16/2017 • 2 min to read • [Edit Online](#)

Prepay for virtual machines and save money with Reserved Virtual Machine Instances. For more information, see [Reserved Virtual Machine Instances offering](#).

You can buy Reserved Virtual Machine Instances in the [Azure portal](#). To buy a Reserved Virtual Machine Instance:

- You must be in an Owner role for at least one Enterprise or Pay-As-You-Go subscription.
- For Enterprise subscriptions, reservation purchases must be enabled in the [EA portal](#).

Buy a Reserved Virtual Machine Instance

1. Log in to the [Azure portal](#).
2. Select **More Services > Reservations**.
3. Select **Add** to purchase a new reservation.
4. Fill in the required fields. Running VM instances that match the attributes you select qualify to get the reservation discount. The actual number of your VM instances that get the discount depend on the scope and quantity selected.

FIELD	DESCRIPTION
Name	The name of this reservation.
Subscription	The subscription used to pay for the reservation. The payment method on the subscription is charged the upfront costs for the reservation. The subscription type must be an enterprise agreement (offer number: MS-AZR-0017P) or Pay-As-You-Go (offer number: MS-AZR-0003P). For an enterprise subscription, the charges are deducted from the enrollment's monetary commitment balance or charged as overage. For Pay-As-You-Go subscription, the charges are billed to the credit card or invoice payment method on the subscription.
Scope	The reservation's scope can cover one subscription or multiple subscriptions (shared scope). If you select: <ul style="list-style-type: none">• Single subscription - The reservation discount is applied to VMs in this subscription.• Shared - The reservation discount is applied to VMs running in any subscriptions within your billing context. For enterprise customers, the shared scope is the enrollment and includes all subscriptions (except dev/test subscriptions) within the enrollment. For Pay-As-You-Go customers, the shared scope is all Pay-As-You-Go subscriptions created by the account administrator.
Location	The Azure region that's covered by the reservation.

FIELD	DESCRIPTION
VM Size	The size of the VM instances.
Term	One year or three years.
Quantity	The number of instances being purchased within the reservation. The quantity is the number of running VM instances that can get the billing discount. For example, if you are running 10 Standard_DS1_v2 VMs in East US, then you would specify quantity as 10 to maximize the benefit for all running machines.

5. You can view the cost of the reservation when you select **Calculate cost**.

COSTS

Calculate cost

Cost per VM	270	USD
Total VMs	10	Standard_DS1_v2
Reservation cost*	2,700	USD

Estimated savings* **53%**

Payment will be processed using the payment instrument of type 'Enrollment' on file for the Microsoft Azure Enterprise subscription.

*Additional taxes may apply. Estimated savings are calculated based on the current on-demand rate for Virtual Machines in the selected subscription.

6. Select **Purchase**.

7. Select **View this Reservation** to see the status of your purchase.

Your Reservation ProdDS1VMReservation has been submitted.

Thank you for purchasing an Azure Reservation. The order has been submitted and the Reservation term will begin as soon as payment has been processed.

[View this Reservation \(ID: f39d1c73-af74-4308-9083-fbf5efab37d1\)](#)

Total cost	2700 USD
Payment subscription	Microsoft Azure Enterprise
Scope	Shared
Location	East US
VM size	Standard_DS1_v2
Term	One year
Quantity	10

Next steps after buying a reservation

The reservation discount is applied automatically to the number of running virtual machines that match the reservation scope and attributes. You can update the scope of the reservation through [Azure portal](#), PowerShell, CLI or through the API.

To learn how to manage a reservation, see [Manage Azure Reserved Virtual Machine Instances](#).

Understanding Azure virtual machine usage

12/6/2017 • 7 min to read • [Edit Online](#)

By analyzing your Azure usage data, powerful consumption insights can be gained – insights that can enable better cost management and allocation throughout your organization. This document provides a deep dive into your Azure Compute consumption details. For more details on general Azure usage, navigate to [Understanding your bill](#).

Download your usage details

To begin, [download your usage details](#). The table below provides the definition and example values of usage for Virtual Machines deployed via the Azure Resource Manager. This document does not contain detailed information for VMs deployed via our classic model.

FIELDS	MEANING	EXAMPLE VALUES
Usage Date	The date when the resource was used.	"11/23/2017"
Meter ID	Identifies the top-level service for which this usage belongs to.	"Virtual Machines"
Meter Sub-Category	<p>The billed meter identifier.</p> <ul style="list-style-type: none">For Compute Hour usage, there is a meter for each VM Size + OS (Windows, Non-Windows) + Region.For Premium software usage, there is a meter for each software type. Most premium software images have different meters for each core size. For more information, visit the Compute Pricing Page.	"2005544f-659d-49c9-9094-8e0aea1be3a5"
Meter Name	This is specific for each service in Azure. For compute, it is always "Compute Hours".	"Compute Hours"
Meter Region	Identifies the location of the datacenter for certain services that are priced based on datacenter location.	"JA East"
Unit	Identifies the unit that the service is charged in. Compute resources are billed per hour.	"Hours"
Consumed	The amount of the resource that has been consumed for that day. For Compute, we bill for each minute the VM ran for a given hour (up to 6 decimals of accuracy).	"1", "0.5"

FIELDS	MEANING	EXAMPLE VALUES
Resource Location	Identifies the datacenter where the resource is running.	"JA East"
Consumed Service	The Azure platform service that you used.	"Microsoft.Compute"
Resource Group	The resource group in which the deployed resource is running in. For more information, see Azure Resource Manager overview .	"MyRG"
Instance ID	The identifier for the resource. The identifier contains the name you specify for the resource when it was created. For VMs, the Instance ID will contain the SubscriptionId, ResourceGroupName, and VMName (or scale set name for scale set usage).	<pre>"/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx/ resourceGroups/MyRG/providers/Microsoft.Compute/virtualMachines/MyVM1"</pre> <p>or</p> <pre>"/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx/ resourceGroups/MyRG/providers/Microsoft.Compute/virtualMachineScaleSets/MyVMSS1"</pre>
Tags	Tag you assign to the resource. Use tags to group billing records. Learn how to tag your Virtual Machines . This is available for Resource Manager VMs only.	<pre>"</pre> <pre>{"myDepartment":"RD","myUser":"myName"}"</pre>
Additional Info	<p>Service-specific metadata. For VMs, we populate the following in the additional info field:</p> <ul style="list-style-type: none"> • Image Type- specific image that you ran. Find the full list of supported strings below under Image Types. • Service Type: the size that you deployed. • VMName: name of your VM. This is only populated for scale set VMs. If you need your VM Name for scale set VMs, you can find that in the Instance ID string above. • UsageType: This specifies the type of usage this represents. <ul style="list-style-type: none"> ◦ ComputeHR is the Compute Hour usage for the underlying VM, like Standard_D1_v2. ◦ ComputeHR_SW is the premium software charge if the VM is using premium software, like Microsoft R Server. 	<p>Virtual Machines</p> <pre>{"ImageType":"Canonical","ServiceType":"Standard_DS1_v2","VMName":"","UsageType":"ComputeHR"}</pre> <p>Virtual Machine Scale Sets</p> <pre>{"ImageType":"Canonical","ServiceType":"Standard_DS1_v2","VMName":"myVM1","UsageType":"ComputeHR"}</pre> <p>Premium Software</p> <pre>{"ImageType": "", "ServiceType": "Standard_DS1_v2", "VMName": "", "UsageType": "ComputeHR_SW"}</pre>

Image Type

For some images in the Azure gallery, the image type is populated in the Additional Info field. This enables users to understand and track what they have deployed on their Virtual Machine. The values that are populated in this field based on the image you have deployed are the following:

- BitRock
- Canonical
- FreeBSD
- Open Logic
- Oracle
- SLES for SAP
- SQL Server 14 Preview on Windows Server 2012 R2 Preview
- SUSE
- SUSE Premium
- StorSimple Cloud Appliance
- Red Hat
- Red Hat for SAP Business Applications
- Red Hat for SAP HANA
- Windows Client BYOL
- Windows Server BYOL
- Windows Server Preview

Service Type

The service type field in the Additional Info field corresponds to the exact VM size you deployed. Premium storage VMs (SSD-based) and non-premium storage VMs (HDD-based) are priced the same. If you deploy an SSD-based size, like Standard_DS2_v2, you see the non-SSD size ('Standard_D2_v2 VM') in the Meter Sub-Category column and the SSD-size ('Standard_DS2_v2') in the Additional Info field.

Region Names

The region name populated in the Resource Location field in the usage details varies from the region name used in the Azure Resource Manager. Here is a mapping between the region values:

RESOURCE MANAGER REGION NAME	RESOURCE LOCATION IN USAGE DETAILS
australiaeast	AU East
australiasoutheast	AU Southeast
brazilsouth	BR South
CanadaCentral	CA Central
CanadaEast	CA East
CentralIndia	IN Central
centralus	Central US

RESOURCE MANAGER REGION NAME	RESOURCE LOCATION IN USAGE DETAILS
chinaeast	China East
chinanorth	China North
eastasia	East Asia
eastus	East US
eastus2	East US 2
GermanyCentral	DE Central
GermanyNortheast	DE Northeast
japaneast	JA East
japanwest	JA West
KoreaCentral	KR Central
KoreaSouth	KR South
northcentralus	North Central US
northeurope	North Europe
southcentralus	South Central US
southeastasia	Southeast Asia
SouthIndia	IN South
UKNorth	US North
uksouth	UK South
UKSouth2	UK South 2
ukwest	UK West
USDoDCentral	US DoD Central
USDoDEast	US DoD East
USGovArizona	USGov Arizona
usgoviowa	USGov Iowa
USGovTexas	USGov Texas

RESOURCE MANAGER REGION NAME	RESOURCE LOCATION IN USAGE DETAILS
usgovvirginia	USGov Virginia
westcentralus	US West Central
westeurope	West Europe
WestIndia	IN West
westus	West US
westus2	US West 2

Virtual machine usage FAQ

What resources are charged when deploying a VM?

VMs acquire costs for the VM itself, any premium software running on the VM, the storage account\managed disk associated with the VM, and the networking bandwidth transfers from the VM.

How can I tell if a VM is using Azure Hybrid Benefit in the Usage CSV?

If you deploy using the [Azure Hybrid Benefit](#), you are charged the Non-Windows VM rate since you are bringing your own license to the cloud. In your bill, you can distinguish which Resource Manager VMs are running Azure Hybrid Benefit because they have either "Windows_Server BYOL" or "Windows_Client BYOL" in the ImageType column.

How are Basic vs. Standard VM Types differentiated in the Usage CSV?

Both Basic and Standard A-Series VMs are offered. If you deploy a Basic VM, in the Meter Sub Category, it has the string "Basic." If you deploy a Standard A-Series VM, then the VM size appears as "A1 VM" since Standard is the default. To learn more about the differences between Basic and Standard, see the [Pricing Page](#).

What are ExtraSmall, Small, Medium, Large, and ExtraLarge sizes?

ExtraSmall - ExtraLarge are the legacy names for Standard_A0 – Standard_A4. In classic VM usage records, you might see this convention used if you have deployed these sizes.

What is the difference between Meter Region and Resource Location?

The Meter Region is associated with the meter. For some Azure services who use one price for all regions, the Meter Region field could be blank. However, since VMs have dedicated prices per region for Virtual Machines, this field is populated. Similarly, the Resource Location for Virtual Machines is the location where the VM is deployed. The Azure region in both fields are the same, although they might have a different string convention for the region name.

Why is the ImageType value blank in the Additional Info field?

The ImageType field is only populated for a subset of images. If you did not deploy one of the images above, the ImageType is blank.

Why is the VMName blank in the Additional Info?

The VMName is only populated in the Additional Info field for VMs in a scale set. The InstanceID field contains the VM name for non-scale set VMs.

What does ComputeHR mean in the UsageType field in the Additional Info?

ComputeHR stands for Compute Hour which represents the usage event for the underlying infrastructure cost. If the UsageType is ComputeHR_SW, the usage event represents the premium software charge for the VM.

How do I know if I am charged for premium software?

When exploring which VM Image best fits your needs, be sure to check out the [Azure Marketplace](#). The image has the software plan rate. If you see "Free" for the rate, there is no additional cost for the software.

What is the difference between Microsoft.ClassicCompute and Microsoft.Compute in the Consumed service?

Microsoft.ClassicCompute represents classic resources deployed via the Azure Service Manager. If you deploy via the Resource Manager, then Microsoft.Compute is populated in the consumed service. Learn more about the [Azure Deployment models](#).

Why is the InstanceID field blank for my Virtual Machine usage?

If you deploy via the classic deployment model, the InstanceID string is not available.

Why are the tags for my VMs not flowing to the usage details?

Tags only flow to you the Usage CSV for Resource Manager VMs only. Classic resource tags are not available in the usage details.

How can the consumed quantity be more than 24 hours one day?

In the Classic model, billing for resources is aggregated at the Cloud Service level. If you have more than one VM in a Cloud Service that uses the same billing meter, your usage is aggregated together. VMs deployed via Resource Manager are billed at the VM level, so this aggregation will not apply.

Why is pricing not available for DS/FS/GS/LS sizes on the pricing page?

Premium storage capable VMs are billed at the same rate as non-premium storage capable VMs. Only your storage costs differ. Visit the [storage pricing page](#) for more information.

Next steps

To learn more about your usage details, see [Understand your bill for Microsoft Azure](#).

Common PowerShell commands for creating and managing Azure Virtual Machines

8/21/2017 • 3 min to read • [Edit Online](#)

This article covers some of the Azure PowerShell commands that you can use to create and manage virtual machines in your Azure subscription. For more detailed help with specific command-line switches and options, you can use **Get-Help command**.

See [How to install and configure Azure PowerShell](#) for information about installing the latest version of Azure PowerShell, selecting your subscription, and signing in to your account.

These variables might be useful for you if running more than one of the commands in this article:

- \$location - The location of the virtual machine. You can use [Get-AzureRmLocation](#) to find a [geographical region](#) that works for you.
- \$myResourceGroup - The name of the resource group that contains the virtual machine.
- \$myVM - The name of the virtual machine.

Create a VM

TASK	COMMAND
Create a VM configuration	\$vm = New-AzureRmVMConfig -VMName \$myVM -VMSize "Standard_D1_v1" The VM configuration is used to define or update settings for the VM. The configuration is initialized with the name of the VM and its size .
Add configuration settings	\$vm = Set-AzureRmVMOperatingSystem -VM \$vm -Windows -ComputerName \$myVM -Credential \$cred -ProvisionVMAgent -EnableAutoUpdate Operating system settings including credentials are added to the configuration object that you previously created using New-AzureRmVMConfig .
Add a network interface	\$vm = Add-AzureRmVMNetworkInterface -VM \$vm -Id \$nic.Id A VM must have a network interface to communicate in a virtual network. You can also use Get-AzureRmNetworkInterface to retrieve an existing network interface object.
Specify a platform image	\$vm = Set-AzureRmVMSourceImage -VM \$vm -PublisherName "publisher_name" -Offer "publisher_offer" -Skus "product_sku" -Version "latest" Image information is added to the configuration object that you previously created using New-AzureRmVMConfig . The object returned from this command is only used when you set the OS disk to use a platform image.

TASK	COMMAND
Set OS disk to use a platform image	\$vm = Set-AzureRmVMOSDisk -VM \$vm -Name "myOSDisk" -VhdUri "http://mystore1.blob.core.windows.net/vhds/myOSDisk.vhd" - CreateOption FromImage The name of the operating system disk and its location in storage is added to the configuration object that you previously created.
Set OS disk to use a generalized image	\$vm = Set-AzureRmVMOSDisk -VM \$vm -Name "myOSDisk" -SourceImageUri "https://mystore1.blob.core.windows.net/system/Microsoft.Co mpute/Images/myimages/myprefix-osDisk.{guid}.vhf" -VhdUri "https://mystore1.blob.core.windows.net/vhds/disk_name.vhd" -CreateOption FromImage -Windows The name of the operating system disk, the location of the source image, and the disk's location in storage is added to the configuration object.
Set OS disk to use a specialized image	\$vm = Set-AzureRmVMOSDisk -VM \$vm -Name "myOSDisk" -VhdUri "http://mystore1.blob.core.windows.net/vhds/" - CreateOption Attach -Windows
Create a VM	-ResourceGroupName \$myResourceGroup -Location \$location -VM \$vm All resources are created in a resource group . Before you run this command, run New-AzureRmVMConfig, Set-AzureRmVMOperatingSystem, Set-AzureRmVMSourceImage, Add-AzureRmVMNetworkInterface, and Set-AzureRmVMOSDisk.

Get information about VMs

TASK	COMMAND
List VMs in a subscription	Get-AzureRmVM
List VMs in a resource group	Get-AzureRmVM -ResourceGroupName \$myResourceGroup To get a list of resource groups in your subscription, use Get-AzureRmResourceGroup .
Get information about a VM	Get-AzureRmVM -ResourceGroupName \$myResourceGroup - Name \$myVM

Manage VMs

TASK	COMMAND
Start a VM	Start-AzureRmVM -ResourceGroupName \$myResourceGroup -Name \$myVM

TASK	COMMAND
Stop a VM	<code>Stop-AzureRmVM -ResourceGroupName \$myResourceGroup -Name \$myVM</code>
Restart a running VM	<code>Restart-AzureRmVM -ResourceGroupName \$myResourceGroup -Name \$myVM</code>
Delete a VM	<code>Remove-AzureRmVM -ResourceGroupName \$myResourceGroup -Name \$myVM</code>
Generalize a VM	<p><code>Set-AzureRmVm -ResourceGroupName \$myResourceGroup -Name \$myVM -Generalized</code></p> <p>Run this command before you run <code>Save-AzureRmVMImage</code>.</p>
Capture a VM	<p><code>Save-AzureRmVMImage -ResourceGroupName \$myResourceGroup -VMName \$myVM -DestinationContainerName "myImageContainer" -VHDNamePrefix "myImagePrefix" -Path "C:\filepath\filename.json"</code></p> <p>A virtual machine must be prepared, shut down and generalized to be used to create an image. Before you run this command, run <code>Set-AzureRmVm</code>.</p>
Update a VM	<p><code>Update-AzureRmVM -ResourceGroupName \$myResourceGroup -VM \$vm</code></p> <p>Get the current VM configuration using <code>Get-AzureRmVM</code>, change configuration settings on the VM object, and then run this command.</p>
Add a data disk to a VM	<p><code>Add-AzureRmVMDataDisk -VM \$vm -Name "myDataDisk" -VhdUri "https://mystore1.blob.core.windows.net/vhds/myDataDisk.vhd" -LUN # -Caching ReadWrite -DiskSizeInGB # -CreateOption Empty</code></p> <p>Use <code>Get-AzureRmVM</code> to get the VM object. Specify the LUN number and the size of the disk. Run <code>Update-AzureRmVM</code> to apply the configuration changes to the VM. The disk that you add is not initialized.</p>
Remove a data disk from a VM	<p><code>Remove-AzureRmVMDataDisk -VM \$vm -Name "myDataDisk"</code></p> <p>Use <code>Get-AzureRmVM</code> to get the VM object. Run <code>Update-AzureRmVM</code> to apply the configuration changes to the VM.</p>
Add an extension to a VM	<p><code>Set-AzureRmVMExtension -ResourceGroupName \$myResourceGroup -Location \$location -VMName \$myVM -Name "extensionName" -Publisher "publisherName" -Type "extensionType" -TypeHandlerVersion "#.#" -Settings \$Settings -ProtectedSettings \$ProtectedSettings</code></p> <p>Run this command with the appropriate configuration information for the extension that you want to install.</p>

TASK	COMMAND
Remove a VM extension	<pre>Remove-AzureRmVMExtension -ResourceGroupName \$myResourceGroup -Name "extensionName" -VMName \$myVM</pre>

Next steps

- See the basic steps for creating a virtual machine in [Create a Windows VM using Resource Manager and PowerShell](#).

Move a Windows VM to another Azure subscription or resource group

12/15/2017 • 2 min to read • [Edit Online](#)

This article walks you through how to move a Windows VM between resource groups or subscriptions. Moving between subscriptions can be handy if you originally created a VM in a personal subscription and now want to move it to your company's subscription to continue your work.

IMPORTANT

You cannot move Managed Disks at this time.

New resource IDs are created as part of the move. Once the VM has been moved, you need to update your tools and scripts to use the new resource IDs.

Use the portal to move a VM to a different subscription

You can move a VM and its associated resources to a different subscription using the portal.

1. Open the [Azure portal](#).
2. Click **Browse > Virtual machines** and select the VM you would like to move from the list.
3. At the top of the page for the VM, select the → **Move** button and then select **Move to another subscription**.
The **Move resources** page opens.
4. Select each of the resources to move. In most cases, you should move all of the related resources that are listed.
5. Select the **Subscription** where you want the VM to be moved.
6. Select an existing **Resource group** or type a name to have a new resource group created.
7. When you are done, select that you understand that new resource IDs are created and those need to be used with the VM once it is moved, then click **OK**.

Use the portal to move a VM to another resource group

You can move a VM and its associated resources to another resource group using the portal.

1. Open the [Azure portal](#).
2. Click **Browse > Virtual machines** and select the VM you would like to move from the list.
3. At the top of the page for the VM, select the → **Move** button and then select **Move to another resource group**.
The **Move resources** page opens.
4. Select each of the resources to move. In most cases, you should move all of the related resources that are listed.
5. Select an existing **Resource group** or type a name to have a new resource group created.
6. When you are done, select that you understand that new resource IDs are created and those need to be used with the VM once it is moved, then click **OK**.

Use Powershell to move a VM

To move a virtual machine to another resource group, you need to make sure that you also move all of the dependent resources. To use the `Move-AzureRMResource` cmdlet, you need the `ResourceId` of each of the resources. You can get a list of the `ResourceId`'s using the [Find-AzureRMResource](#) cmdlet.

```
Find-AzureRMResource -ResourceGroupNameContains <sourceResourceGroupName> | Format-table -Property ResourceId
```

To move a VM we need to move multiple resources. We can use the output of `Find-AzureRMResource` to create a comma separated list of the ResourceIds and pass that to [Move-AzureRMResource](#) to move them to the destination.

```
Move-AzureRmResource -DestinationResourceGroupName "<myDestinationResourceGroup>" `  
-ResourceId <myResourceId,myResourceId,myResourceId>
```

To move the resources to different subscription, include the **-DestinationSubscriptionId** parameter.

```
Move-AzureRmResource -DestinationSubscriptionId "<myDestinationSubscriptionID>" `  
-DestinationResourceGroupName "<myDestinationResourceGroup>" `  
-ResourceId <myResourceId,myResourceId,myResourceId>
```

You will be asked to confirm that you want to move the specified resources.

Next steps

You can move many different types of resources between resource groups and subscriptions. For more information, see [Move resources to new resource group or subscription](#).

Resize a Windows VM

6/27/2017 • 2 min to read • [Edit Online](#)

This article shows you how to resize a Windows VM, created in the Resource Manager deployment model using Azure Powershell.

After you create a virtual machine (VM), you can scale the VM up or down by changing the VM size. In some cases, you must deallocate the VM first. This can happen if the new size is not available on the hardware cluster that is currently hosting the VM.

Resize a Windows VM not in an availability set

1. List the VM sizes that are available on the hardware cluster where the VM is hosted.

```
Get-AzureRmVMSize -ResourceGroupName <resourceGroupName> -VMName <vmName>
```

2. If the desired size is listed, run the following commands to resize the VM. If the desired size is not listed, go on to step 3.

```
$vm = Get-AzureRmVM -ResourceGroupName <resourceGroupName> -VMName <vmName>
$vm.HardwareProfile.VmSize = "<newVmSize>"
Update-AzureRmVM -VM $vm -ResourceGroupName <resourceGroupName>
```

3. If the desired size is not listed, run the following commands to deallocate the VM, resize it, and restart the VM.

```
$rgname = "<resourceGroupName>"
$vmname = "<vmName>"
Stop-AzureRmVM -ResourceGroupName $rgname -VMName $vmname -Force
$vm = Get-AzureRmVM -ResourceGroupName $rgname -VMName $vmname
$vm.HardwareProfile.VmSize = "<newVmSize>"
Update-AzureRmVM -VM $vm -ResourceGroupName $rgname
Start-AzureRmVM -ResourceGroupName $rgname -Name $vmname
```

WARNING

Deallocating the VM releases any dynamic IP addresses assigned to the VM. The OS and data disks are not affected.

Resize a Windows VM in an availability set

If the new size for a VM in an availability set is not available on the hardware cluster currently hosting the VM, then all VMs in the availability set will need to be deallocated to resize the VM. You also might need to update the size of other VMs in the availability set after one VM has been resized. To resize a VM in an availability set, perform the following steps.

1. List the VM sizes that are available on the hardware cluster where the VM is hosted.

```
Get-AzureRmVMSize -ResourceGroupName <resourceGroupName> -VMName <vmName>
```

2. If the desired size is listed, run the following commands to resize the VM. If it is not listed, go to step 3.

```
$vm = Get-AzureRmVM -ResourceGroupName <resourceGroupName> -VMName <vmName>
$vm.HardwareProfile.VmSize = "<newVmSize>"
Update-AzureRmVM -VM $vm -ResourceGroupName <resourceGroupName>
```

3. If the desired size is not listed, continue with the following steps to deallocate all VMs in the availability set, resize VMs, and restart them.
4. Stop all VMs in the availability set.

```
$rg = "<resourceGroupName>"
$as = Get-AzureRmAvailabilitySet -ResourceGroupName $rg
$vmIDs = $as.VirtualMachinesReferences
foreach ($vmId in $vmIDs){
    $string = $vmID.Id.Split("/")
    $vmName = $string[8]
    Stop-AzureRmVM -ResourceGroupName $rg -Name $vmName -Force
}
```

5. Resize and restart the VMs in the availability set.

```
$rg = "<resourceGroupName>"
$newSize = "<newVmSize>"
$as = Get-AzureRmAvailabilitySet -ResourceGroupName $rg
$vmIDs = $as.VirtualMachinesReferences
foreach ($vmId in $vmIDs){
    $string = $vmID.Id.Split("/")
    $vmName = $string[8]
    $vm = Get-AzureRmVM -ResourceGroupName $rg -Name $vmName
    $vm.HardwareProfile.VmSize = $newSize
    Update-AzureRmVM -ResourceGroupName $rg -VM $vm
    Start-AzureRmVM -ResourceGroupName $rg -Name $vmName
}
```

Next steps

- For additional scalability, run multiple VM instances and scale out. For more information, see [Automatically scale Windows machines in a Virtual Machine Scale Set](#).

How to tag a Windows virtual machine in Azure

6/27/2017 • 4 min to read • [Edit Online](#)

This article describes different ways to tag a Windows virtual machine in Azure through the Resource Manager deployment model. Tags are user-defined key/value pairs which can be placed directly on a resource or a resource group. Azure currently supports up to 15 tags per resource and resource group. Tags may be placed on a resource at the time of creation or added to an existing resource. Please note that tags are supported for resources created via the Resource Manager deployment model only. If you want to tag a Linux virtual machine, see [How to tag a Linux virtual machine in Azure](#).

Tagging a Virtual Machine through Templates

First, let's look at tagging through templates. [This template](#) places tags on the following resources: Compute (Virtual Machine), Storage (Storage Account), and Network (Public IP Address, Virtual Network, and Network Interface). This template is for a Windows VM but can be adapted for Linux VMs.

Click the **Deploy to Azure** button from the [template link](#). This will navigate to the [Azure portal](#) where you can deploy this template.

Simple deployment of a VM with Tags

 [Deploy to Azure](#)

 [Visualize](#)

This template includes the following tags: *Department*, *Application*, and *Created By*. You can add/edit these tags directly in the template if you would like different tag names.

```
"apiVersion": "2015-05-01-preview",
"type": "Microsoft.Compute/virtualMachines",
"name": "[variables('vmName')]",
"location": "[variables('location')]",
"tags": {
    "Department": "[parameters('departmentName')]",
    "Application": "[parameters('applicationName')]",
    "Created By": "[parameters('createdBy')]"
},
```

As you can see, the tags are defined as key/value pairs, separated by a colon (:). The tags must be defined in this format:

```
"tags": {
    "Key1" : "Value1",
    "Key2" : "Value2"
}
```

Save the template file after you finish editing it with the tags of your choice.

Next, in the **Edit Parameters** section, you can fill out the values for your tags.

DEPARTMENTNAME (string) ⓘ

APPLICATIONNAME (string) ⓘ

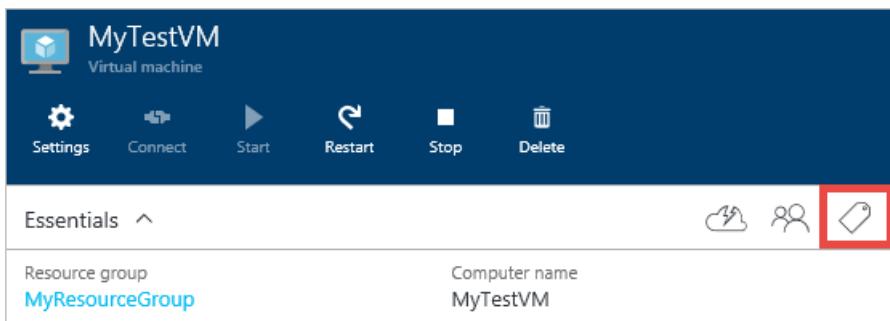
CREATEDBY (string) ⓘ

Click **Create** to deploy this template with your tag values.

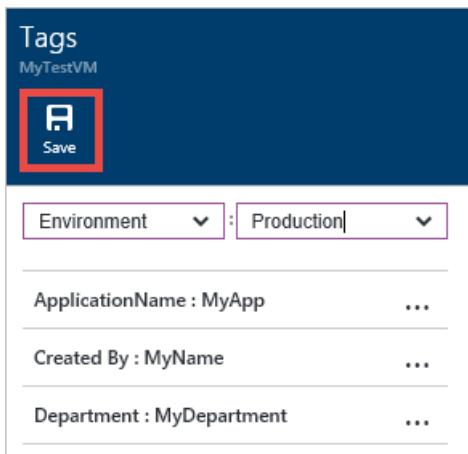
Tagging through the Portal

After creating your resources with tags, you can view, add, and delete tags in the portal.

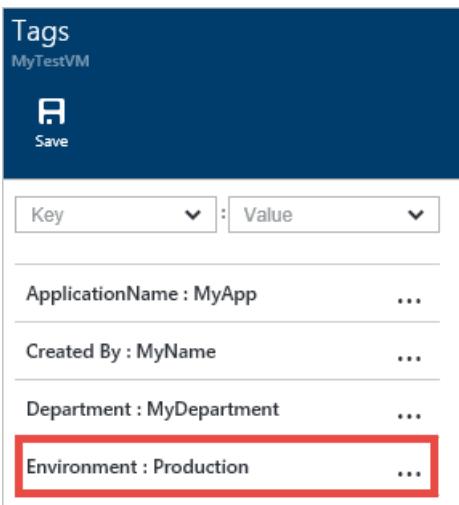
Select the tags icon to view your tags:



Add a new tag through the portal by defining your own Key/Value pair, and save it.



Your new tag should now appear in the list of tags for your resource.



Tagging with PowerShell

To create, add, and delete tags through PowerShell, you first need to set up your [PowerShell environment with Azure Resource Manager](#). Once you have completed the setup, you can place tags on Compute, Network, and Storage resources at creation or after the resource is created via PowerShell. This article will concentrate on viewing/editing tags placed on Virtual Machines.

First, navigate to a Virtual Machine through the `Get-AzureRmVM` cmdlet.

```
PS C:\> Get-AzureRmVM -ResourceGroupName "MyResourceGroup" -Name "MyTestVM"
```

If your Virtual Machine already contains tags, you will then see all the tags on your resource:

```
Tags : {  
    "Application": "MyApp1",  
    "Created By": "MyName",  
    "Department": "MyDepartment",  
    "Environment": "Production"  
}
```

If you would like to add tags through PowerShell, you can use the `Set-AzureRmResource` command. Note when updating tags through PowerShell, tags are updated as a whole. So if you are adding one tag to a resource that already has tags, you will need to include all the tags that you want to be placed on the resource. Below is an example of how to add additional tags to a resource through PowerShell Cmdlets.

This first cmdlet sets all of the tags placed on *MyTestVM* to the `$tags` variable, using the `Get-AzureRmResource` and `Tags` property.

```
PS C:\> $tags = (Get-AzureRmResource -ResourceGroupName MyResourceGroup -Name MyTestVM).Tags
```

The second command displays the tags for the given variable.

```
PS C:\> $tags

Name      Value
----      -----
Value     MyDepartment
Name      Department
Value     MyApp1
Name      Application
Value     MyName
Name      Created By
Value     Production
Name      Environment
```

The third command adds an additional tag to the `$tags` variable. Note the use of the `+=` to append the new key/value pair to the `$tags` list.

```
PS C:\> $tags += @{$Name="Location";Value="MyLocation"}
```

The fourth command sets all of the tags defined in the `$tags` variable to the given resource. In this case, it is `MyTestVM`.

```
PS C:\> Set-AzureRmResource -ResourceGroupName MyResourceGroup -Name MyTestVM -ResourceType
"Microsoft.Compute/VirtualMachines" -Tag $tags
```

The fifth command displays all of the tags on the resource. As you can see, `Location` is now defined as a tag with `MyLocation` as the value.

```
PS C:\> (Get-AzureRmResource -ResourceGroupName MyResourceGroup -Name MyTestVM).Tags

Name      Value
----      -----
Value     MyDepartment
Name      Department
Value     MyApp1
Name      Application
Value     MyName
Name      Created By
Value     Production
Name      Environment
Value     MyLocation
Name      Location
```

To learn more about tagging through PowerShell, check out the [Azure Resource Cmdlets](#).

Viewing your tags in the usage details

Tags placed on Compute, Network, and Storage resources in the Resource Manager deployment model will be populated in your usage details in the [billing portal](#).

Click on **Download usage details** to view the usage details in your subscription.

NEXT BILL (ESTIMATED):

\$0.00

DATE PURCHASED
6/24/2014

CURRENT BILLING PERIOD
5/24/2015 - 6/23/2015

 [Download usage details](#)

-  [Contact Microsoft Support](#)
-  [Edit subscription details](#)
-  [Change subscription address](#)
-  [Partner Information](#)
-  [Cancel Subscription](#)

Select your billing statement and the **Version 2** usage details:

Click here to [Understand Your Bill](#).

Current period	View Current Statement	Download Usage	▼
7/24/2014 - 8/23/2014		Version 2 - Preview	
		Download Usage	Version 1

From the usage details, you can see all of the tags in the **Tags** column:

Consumed Service	Resource Group	Instance Id	Tags
"Microsoft.Compute"	"MYRESOURCEGROUP"	"/subscriptions/00000000-0000-0000-0000-000000000000/resourceGroups/MYRESOURCEGROUP/providers/Microsoft.Compute/virtualMachines/MyWindowsVM"	"[{"Department":"MyDepartment","Application":"MyApp1","Created By":"MyName","Type":"Virtual Machine","Environment":"Production","Location":"MyLocation"}]"
"Microsoft.Storage"	"myresourcegroup"	"/subscriptions/00000000-0000-0000-0000-000000000000/resourceGroups/myresourcegroup/providers/Microsoft.Storage/storageAccounts/mystorageaccount"	"[{"Application":"MyApp1","Created By":"MyName","Department":"MyDepartment","Type":"Storage Account"}]"
"Microsoft.Network"	"MyResourceGroup"	"/subscriptions/00000000-0000-0000-0000-000000000000/resourceGroups/MyResourceGroup/providers/Microsoft.Network/publicIPAddresses/myPublicIP"	"[{"Department":"MyDepartment","Application":"MyApp1","Created By":"MyName","Type":"Public IP"}]"

By analyzing these tags along with usage, organizations will be able to gain new insights into their consumption data.

Next steps

- To learn more about tagging your Azure resources, see [Azure Resource Manager Overview](#) and [Using Tags to organize your Azure Resources](#).
- To see how tags can help you manage your use of Azure resources, see [Understanding your Azure Bill](#) and [Gain insights into your Microsoft Azure resource consumption](#).

Custom Script Extension for Windows

12/7/2017 • 3 min to read • [Edit Online](#)

The Custom Script Extension downloads and executes scripts on Azure virtual machines. This extension is useful for post deployment configuration, software installation, or any other configuration / management task. Scripts can be downloaded from Azure storage or GitHub, or provided to the Azure portal at extension run time. The Custom Script extension integrates with Azure Resource Manager templates, and can also be run using the Azure CLI, PowerShell, Azure portal, or the Azure Virtual Machine REST API.

This document details how to use the Custom Script Extension using the Azure PowerShell module, Azure Resource Manager templates, and details troubleshooting steps on Windows systems.

Prerequisites

Operating System

The Custom Script Extension for Windows can be run against Windows 10 Client, Windows Server 2008 R2, 2012, 2012 R2, and 2016 releases.

Script Location

The script needs to be stored in Azure Blob storage, or any other location accessible through a valid URL.

Internet Connectivity

The Custom Script Extension for Windows requires that the target virtual machine is connected to the internet.

Extension schema

The following JSON shows the schema for the Custom Script Extension. The extension requires a script location (Azure Storage or other location with valid URL), and a command to execute. If using Azure Storage as the script source, an Azure storage account name and account key is required. These items should be treated as sensitive data and specified in the extensions protected setting configuration. Azure VM extension protected setting data is encrypted, and only decrypted on the target virtual machine.

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Compute/virtualMachines/extensions",
  "name": "config-app",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'), copyIndex())]",
    "[variables('musicstoresqlName')]"
  ],
  "tags": {
    "displayName": "config-app"
  },
  "properties": {
    "publisher": "Microsoft.Compute",
    "type": "CustomScriptExtension",
    "typeHandlerVersion": "1.9",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "fileUris": [
        "script location"
      ]
    },
    "protectedSettings": {
      "commandToExecute": "myExecutionCommand",
      "storageAccountName": "myStorageAccountName",
      "storageAccountKey": "myStorageAccountKey"
    }
  }
}
```

Property values

NAME	VALUE / EXAMPLE
apiVersion	2015-06-15
publisher	Microsoft.Compute
type	extensions
typeHandlerVersion	1.9
fileUris (e.g.)	https://raw.githubusercontent.com/Microsoft/dotnet-core-sample-templates/master/dotnet-core-music-windows/scripts/configure-music-app.ps1
commandToExecute (e.g.)	powershell -ExecutionPolicy Unrestricted -File configure-music-app.ps1
storageAccountName (e.g.)	examplestorageacct

NAME	VALUE / EXAMPLE
storageAccountKey (e.g.)	TmJK/1N3AbAZ3q/+hOXoi/l73zOqsaxXDHqa9Y83/v5UpXQp2DQIBuv2Tifp60cE/OaHsJZmQZ7teQfczQj8hg==

Note – these property names are case sensitive. Use the names as seen above to avoid deployment issues.

Template deployment

Azure VM extensions can be deployed with Azure Resource Manager templates. The JSON schema detailed in the previous section can be used in an Azure Resource Manager template to run the Custom Script Extension during an Azure Resource Manager template deployment. A sample template that includes the Custom Script Extension can be found here, [GitHub](#).

PowerShell deployment

The `Set-AzureRmVMCustomScriptExtension` command can be used to add the Custom Script extension to an existing virtual machine. For more information, see [Set-AzureRmVMCustomScriptExtension](#).

```
Set-AzureRmVMCustomScriptExtension -ResourceGroupName myResourceGroup ` 
-VMName myVM ` 
-Location myLocation ` 
-FileUri myURL ` 
-Run 'myScript.ps1' ` 
-Name DemoScriptExtension
```

Troubleshoot and support

Troubleshoot

Data about the state of extension deployments can be retrieved from the Azure portal, and by using the Azure PowerShell module. To see the deployment state of extensions for a given VM, run the following command.

```
Get-AzureRmVMExtension -ResourceGroupName myResourceGroup -VMName myVM -Name myExtensionName
```

Extension execution output is logged to files found under the following directory on the target virtual machine.

```
C:\WindowsAzure\Logs\Plugins\Microsoft.Compute.CustomScriptExtension
```

The specified files are downloaded into the following directory on the target virtual machine.

```
C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1.*\Downloads\<n>
```

where `<n>` is a decimal integer which may change between executions of the extension. The `1.*` value matches the actual, current `typeHandlerVersion` value of the extension. For example, the actual directory could be `C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1.8\Downloads\2`.

When executing the `commandToExecute` command, the extension will have set this directory (e.g., `...\Downloads\2`) as the current working directory. This enables the use of relative paths to locate the files downloaded via the `fileUris` property. See the table below for examples.

Since the absolute download path may vary over time, it is better to opt for relative script/file paths in the `commandToExecute` string, whenever possible. For example:

```
"commandToExecute": "powershell.exe . . . -File './scripts/myscript.ps1'"
```

Path information after the first URI segment is retained for files downloaded via the `fileUris` property list. As shown in the table below, downloaded files are mapped into download subdirectories to reflect the structure of the `fileUris` values.

Examples of Downloaded Files

URI IN FILEURIS	RELATIVE DOWNLOADED LOCATION	ABSOLUTE DOWNLOADED LOCATION *
<code>https://someAcct.blob.core.windows.net/aContainer/script` ./scripts/myscript.ps1</code>		<code>C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1</code>
<code>https://someAcct.blob.core.windows.net/aContainer/topLevel` ./topLevel.ps1</code>		<code>C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1</code>

* As above, the absolute directory paths will change over the lifetime of the VM, but not within a single execution of the CustomScript extension.

Support

If you need more help at any point in this article, you can contact the Azure experts on the [MSDN Azure and Stack Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select Get support. For information about using Azure Support, read the [Microsoft Azure support FAQ](#).

Use the D: drive as a data drive on a Windows VM

6/27/2017 • 2 min to read • [Edit Online](#)

If your application needs to use the D drive to store data, follow these instructions to use a different drive letter for the temporary disk. Never use the temporary disk to store data that you need to keep.

If you resize or **Stop (Deallocate)** a virtual machine, this may trigger placement of the virtual machine to a new hypervisor. A planned or unplanned maintenance event may also trigger this placement. In this scenario, the temporary disk will be reassigned to the first available drive letter. If you have an application that specifically requires the D: drive, you need to follow these steps to temporarily move the pagefile.sys, attach a new data disk and assign it the letter D and then move the pagefile.sys back to the temporary drive. Once complete, Azure will not take back the D: if the VM moves to a different hypervisor.

For more information about how Azure uses the temporary disk, see [Understanding the temporary drive on Microsoft Azure Virtual Machines](#)

Attach the data disk

First, you'll need to attach the data disk to the virtual machine. To do this using the portal, see [How to attach a managed data disk in the Azure portal](#).

Temporarily move pagefile.sys to C drive

1. Connect to the virtual machine.
2. Right-click the **Start** menu and select **System**.
3. In the left-hand menu, select **Advanced system settings**.
4. In the **Performance** section, select **Settings**.
5. Select the **Advanced** tab.
6. In the **Virtual memory** section, select **Change**.
7. Select the **C** drive and then click **System managed size** and then click **Set**.
8. Select the **D** drive and then click **No paging file** and then click **Set**.
9. Click **Apply**. You will get a warning that the computer needs to be restarted for the changes to take affect.
10. Restart the virtual machine.

Change the drive letters

1. Once the VM restarts, log back on to the VM.
2. Click the **Start** menu and type **diskmgmt.msc** and hit Enter. Disk Management will start.
3. Right-click on **D**, the Temporary Storage drive, and select **Change Drive Letter and Paths**.
4. Under Drive letter, select a new drive such as **T** and then click **OK**.
5. Right-click on the data disk, and select **Change Drive Letter and Paths**.
6. Under Drive letter, select drive **D** and then click **OK**.

Move pagefile.sys back to the temporary storage drive

1. Right-click the **Start** menu and select **System**
2. In the left-hand menu, select **Advanced system settings**.
3. In the **Performance** section, select **Settings**.

4. Select the **Advanced** tab.
5. In the **Virtual memory** section, select **Change**.
6. Select the OS drive **C** and click **No paging file** and then click **Set**.
7. Select the temporary storage drive **T** and then click **System managed size** and then click **Set**.
8. Click **Apply**. You will get a warning that the computer needs to be restarted for the changes to take affect.
9. Restart the virtual machine.

Next steps

- You can increase the storage available to your virtual machine by [attaching a additional data disk](#).

Change the availability set for a Windows VM

8/30/2017 • 2 min to read • [Edit Online](#)

The following steps describe how to change the availability set of a VM using Azure PowerShell. A VM can only be added to an availability set when it is created. In order to change the availability set, you need to delete and recreate the virtual machine.

Change the availability set using PowerShell

1. Capture the following key details from the VM to be modified.

Name of the VM

```
$vm = Get-AzureRmVM -ResourceGroupName <Name-of-resource-group> -Name <name-of-VM>  
$vm.Name
```

VM Size

```
$vm.HardwareProfile.VmSize
```

Network primary network interface and optional network interfaces if they exist on the VM

```
$vm.NetworkProfile.NetworkInterfaces[0].Id
```

OS Disk Profile

```
$vm.StorageProfile.OsDisk.OsType  
$vm.StorageProfile.OsDisk.Name  
$vm.StorageProfile.OsDisk.Vhd.Uri
```

Disk profiles for each data disk

```
$vm.StorageProfile.DataDisks[<index>].Lun  
$vm.StorageProfile.DataDisks[<index>].Vhd.Uri
```

VM extensions installed

```
$vm.Extensions
```

2. Delete the VM without deleting any of the disks or the network interfaces.

```
Remove-AzureRmVM -ResourceGroupName <resourceGroupName> -Name <vmName>
```

3. Create the availability set if it does not already exist

```
New-AzureRmAvailabilitySet -ResourceGroupName <resourceGroupName> -Name <availabilitySetName> -Location <location>
```

4. Recreate the VM using the new availability set

```
$vm2 = New-AzureRmVMConfig -VMName <VM-name> -VMSize <vm-size> -AvailabilitySetId <availability-set-id>

Set-AzureRmVMOSDisk -CreateOption "Attach" -VM <vmConfig> -VhdUri <osDiskURI> -Name <osDiskName> [-Windows | -Linux]

Add-AzureRmVMNetworkInterface -VM <vmConfig> -Id <nictId>

New-AzureRmVM -ResourceGroupName <resourceGroupName> -Location <location> -VM <vmConfig>
```

5. Add data disks and extensions. For more information, see [Attach Data Disk to VM and Extensions in Resource Manager templates](#). Data disks and extensions can be added to the VM using PowerShell or Azure CLI.

Example Script

The following script provides an example of gathering the required information, deleting the original VM and then recreating it in a new availability set.

```

#set variables
$rg = "demo-resource-group"
$vmName = "demo-vm"
$newAvailSetName = "demo-as"
$outFile = "C:\temp\outfile.txt"

#Get VM Details
$OriginalVM = get-azurermmvm -ResourceGroupName $rg -Name $vmName

#Output VM details to file
"VM Name: " | Out-File -FilePath $outFile
$OriginalVM.Name | Out-File -FilePath $outFile -Append

"Extensions: " | Out-File -FilePath $outFile -Append
$OriginalVM.Extensions | Out-File -FilePath $outFile -Append

"VmSize: " | Out-File -FilePath $outFile -Append
$OriginalVM.HardwareProfile.VmSize | Out-File -FilePath $outFile -Append

"NIC: " | Out-File -FilePath $outFile -Append
$OriginalVM.NetworkProfile.NetworkInterfaces[0].Id | Out-File -FilePath $outFile -Append

"OSType: " | Out-File -FilePath $outFile -Append
$OriginalVM.StorageProfile.OsDisk.OsType | Out-File -FilePath $outFile -Append

"OS Disk: " | Out-File -FilePath $outFile -Append
$OriginalVM.StorageProfile.OsDisk.Vhd.Uri | Out-File -FilePath $outFile -Append

if ($OriginalVM.StorageProfile.DataDisks) {
    "Data Disk(s): " | Out-File -FilePath $outFile -Append
    $OriginalVM.StorageProfile.DataDisks | Out-File -FilePath $outFile -Append
}

#Remove the original VM
Remove-AzureRmVM -ResourceGroupName $rg -Name $vmName

#Create new availability set if it does not exist
$availSet = Get-AzureRmAvailabilitySet -ResourceGroupName $rg -Name $newAvailSetName -ErrorAction Ignore
if (-Not $availSet) {
    $availset = New-AzureRmAvailabilitySet -ResourceGroupName $rg -Name $newAvailSetName -Location
$OriginalVM.Location
}

#Create the basic configuration for the replacement VM
$NewVM = New-AzureRmVMConfig -VMName $OriginalVM.Name -VmSize $OriginalVM.HardwareProfile.VmSize -
AvailabilitySetId $availSet.Id
Set-AzureRmVMOSDisk -VM $NewVM -VhdUri $OriginalVM.StorageProfile.OsDisk.Vhd.Uri -Name $OriginalVM.Name -
CreateOption Attach -Windows

#Add Data Disks
foreach ($disk in $OriginalVM.StorageProfile.DataDisks ) {
    Add-AzureRmVMDataDisk -VM $NewVM -Name $disk.Name -VhdUri $disk.Vhd.Uri -Caching $disk.Caching -Lun
$disk.Lun -CreateOption Attach -DiskSizeInGB $disk.DiskSizeGB
}

#Add NIC(s)
foreach ($nic in $OriginalVM.NetworkProfile.NetworkInterfaces) {
    Add-AzureRmVMNetworkInterface -VM $NewVM -Id $nic.Id
}

#Create the VM
New-AzureRmVM -ResourceGroupName $rg -Location $OriginalVM.Location -VM $NewVM -DisableBginfoExtension

```

Next steps

Add additional storage to your VM by adding an additional [data disk](#).

Download the template for a VM

12/1/2017 • 1 min to read • [Edit Online](#)

When you create a VM in Azure using the portal or PowerShell, a Resource Manager template is automatically created for you. You can use this template to quickly duplicate a deployment. The template contains information about all of the resources in a resource group. For a virtual machine, this means the template contains everything that is created in support of the VM in that resource group, including the networking resources.

Download the template using the portal

1. Log in to the [Azure portal](#).
2. On the left menu, select **Virtual Machines**.
3. Select the virtual machine from the list.
4. Select **Automation script**.
5. Select **Download** from the menu at the top and save the .zip file to your local computer.
6. Open the .zip file and extract the files to a folder. The .zip file contains:
 - deploy.ps1
 - deploy.sh
 - deployer.rb
 - DeploymentHelper.cs
 - parameters.json
 - template.json

The template.json file is the template.

Download the template using PowerShell

You can also download the .json template file using the `Export-AzureRMResourceGroup` cmdlet. You can use the `-path` parameter to provide the filename and path for the .json file. This example shows how to download the template for the resource group named **myResourceGroup** to the **C:\users\public\downloads** folder on your local computer.

```
Export-AzureRmResourceGroup -ResourceGroupName "myResourceGroup" -Path "C:\users\public\downloads"
```

Next steps

To learn more about deploying resources using templates, see [Resource Manager template walkthrough](#).

Azure Virtual Machine Agent overview

11/8/2017 • 2 min to read • [Edit Online](#)

The Microsoft Azure Virtual Machine Agent (AM Agent) is a secured, lightweight process that manages VM interaction with the Azure Fabric Controller. The VM Agent has a primary role in enabling and executing Azure virtual machine extensions. VM Extensions enabling post deployment configuration of virtual machines, such as installing and configuring software. Virtual machine extensions also enable recovery features such as resetting the administrative password of a virtual machine. Without the Azure VM Agent, virtual machine extensions cannot be run.

This document details installation, detection, and removal of the Azure Virtual Machine Agent.

Install the VM Agent

Azure gallery image

The Azure VM Agent is installed by default on any Windows virtual machine deployed from an Azure Gallery image. When deploying an Azure gallery image from the Portal, PowerShell, Command Line Interface, or an Azure Resource Manager template, the Azure VM Agent is also be installed.

Manual installation

The Windows VM agent can be manually installed using a Windows installer package. Manual installation may be necessary when creating a custom virtual machine image that will be deployed in Azure. To manually install the Windows VM Agent, download the VM Agent installer from this location [Windows Azure VM Agent Download](#).

The VM Agent can be installed by double-clicking the windows installer file. For an automated or unattended installation of the VM agent, run the following command.

```
msiexec.exe /i WindowsAzureVmAgent.2.7.1198.778.rd_art_stable.160617-1120.fre /quiet
```

Detect the VM Agent

PowerShell

The Azure Resource Manager PowerShell module can be used to retrieve information about Azure Virtual Machines. Running `Get-AzureRmVM` returns quite a bit of information including the provisioning state for the Azure VM Agent.

```
Get-AzureRmVM
```

The following is just a subset of the `Get-AzureRmVM` output. Notice the `ProvisionVMAgent` property nested inside `OSProfile`, this property can be used to determine if the VM agent has been deployed to the virtual machine.

```
OSProfile :  
ComputerName : myVM  
AdminUsername : muUserName  
WindowsConfiguration :  
ProvisionVMAgent : True  
EnableAutomaticUpdates : True
```

The following script can be used to return a concise list of virtual machine names and the state of the VM Agent.

```
$vms = Get-AzureRmVM

foreach ($vm in $vms) {
    $agent = $vm | Select -ExpandProperty OSProfile | Select -ExpandProperty WindowsConfiguration | Select
    ProvisionVMAgent
    Write-Host $vm.Name $agent.ProvisionVMAgent
}
```

Manual Detection

When logged in to a Windows Azure VM, task manager can be used to examine running processes. To check for the Azure VM Agent, open Task Manager > click the details tab, and look for a process name `WindowsAzureGuestAgent.exe`. The presence of this process indicates that the VM agent is installed.

Upgrade the VM Agent

The Azure VM Agent for Windows is automatically upgraded. As new virtual machines are deployed to Azure, they receive the latest VM agent. Custom VM images should be manually updated to include the new VM agent.

Handling planned maintenance notifications for Windows virtual machines

12/29/2017 • 12 min to read • [Edit Online](#)

Azure periodically performs updates to improve the reliability, performance, and security of the host infrastructure for virtual machines. Updates are changes like patching the hosting environment or upgrading and decommissioning hardware. A majority of these updates are performed without any impact to the hosted virtual machines. However, there are cases where updates do have an impact:

- If the maintenance does not require a reboot, Azure uses in-place migration to pause the VM while the host is updated.
- If maintenance requires a reboot, you get a notice of when the maintenance is planned. In these cases, you are given a time window where you can start the maintenance yourself, when it works for you.

Planned maintenance that requires a reboot, is scheduled in waves. Each wave has different scope (regions).

- A wave starts with a notification to customers. By default, notification is sent to subscription owner and co-owners. You can add more recipients and messaging options like email, SMS, and Webhooks, to the notifications using Azure [Activity Log Alerts](#).
- At the time of the notification, a *self-service window* is made available. During this window, you can find which of your virtual machines are included in this wave and proactively start maintenance according to your own scheduling needs.
- After the self-service window, a *scheduled maintenance window* begins. At some point during this window, Azure schedules and applies the required maintenance to your virtual machine.

The goal in having two windows is to give you enough time to start maintenance and reboot your virtual machine while knowing when Azure will automatically start maintenance.

You can use the Azure portal, PowerShell, REST API, and CLI to query for the maintenance windows for your VMs and start self-service maintenance.

NOTE

If you try to start maintenance and the request fails, Azure marks your VM as **skipped**. You will no longer be able to use the Customer Initiated Maintenance option. Your VM will have to be rebooted by Azure during the scheduled maintenance phase.

Should you start maintenance using during the self-service window?

The following guidelines should help you to decide whether you should use this capability and start maintenance at your own time.

NOTE

Self-service maintenance might not be available for all of your VMs. To determine if proactive redeploy is available for your VM, look for the **Start now** in the maintenance status. Self-service maintenance is currently not available for Cloud Services (Web/Worker Role), Service Fabric, and Virtual Machine Scale Sets.

Self-service maintenance is not recommended for deployments using **availability sets** since these are highly

available setups, where only one update domain is impacted at any given time.

- Let Azure trigger the maintenance, but be aware that the order of update domains being impacted does not necessarily happen sequentially and that there is a 30-minute pause between update domains.
- If a temporary loss of some of your capacity (1/update domain count) is a concern, it can easily be compensated for by allocating additional instances during the maintenance period **Don't** use self-service maintenance in the following scenarios:
 - If you shut down your VMs frequently, either manually, using Dev/Test labs, using auto-shutdown, or following a schedule, it could revert the maintenance status and therefore cause additional downtime.
 - On short-lived VMs which you know will be deleted before the end of the maintenance wave.
 - For workloads with a large state stored in the local (ephemeral) disk that is desired to be maintained upon update.
 - For cases where you resize your VM often, as it could revert the maintenance status.
 - If you have adopted scheduled events which enable proactive failover or graceful shutdown of your workload, 15 minutes before start of maintenance shutdown

Use self-service maintenance, if you are planning to run your VM uninterrupted during the scheduled maintenance phase and none of the counter-indications mentioned above are applicable.

It is best to use self-service maintenance in the following cases:

- You need to communicate an exact maintenance window to your management or end-customer.
- You need to complete the maintenance by a given date.
- You need to control the sequence of maintenance, e.g., multi-tier application to guarantee safe recovery.
- You need more than 30 minutes of VM recovery time between two update domains (UDs). To control the time between update domains, you must trigger maintenance on your VMs one update domain (UD) at a time.

View VMs scheduled for maintenance in the portal

Once a planned maintenance wave is scheduled, and notifications are sent, you can observe the list of virtual machines that are impacted by the upcoming maintenance wave.

You can use the Azure portal and look for VMs scheduled for maintenance.

1. Sign in to the [Azure portal](#).
2. In the left navigation, click **Virtual Machines**.
3. In the Virtual Machines pane, click the **Columns** button to open the list of available columns.
4. Select and add the following columns:

Maintenance - shows the maintenance status for the VM. The following are the potential values:

VALUE	DESCRIPTION
Start now	The VM is in the self-service maintenance window which lets you initiate the maintenance yourself. See below on how to start maintenance on your VM
Scheduled	The VM is scheduled for maintenance with no option for you to initiate maintenance. You can learn of the maintenance window by selecting the Auto-Scheduled window in this view or by clicking on the VM

Value	Description
Completed	You have successfully initiated and completed maintenance on your VM.
Skipped	You have selected to initiate maintenance with no success. Azure has canceled the maintenance for your VM and will reschedule it in a later time
Retry later	You have selected to initiate maintenance and Azure was not able to fulfill your request. In this case, you can try again in a later time.

Maintenance Pro-Active - shows the time window when you can self-start maintenance on your VMs.

Maintenance Scheduled - shows the time window when Azure will reboot your VM in order to complete maintenance.

Notification and alerts in the portal

Azure communicates a schedule for planned maintenance by sending an email to the subscription owner and co-owners group. You can add additional recipients and channels to this communication by creating Azure activity log alerts. For more information, see [Monitor subscription activity with the Azure Activity Log](#)

1. Sign in to the [Azure portal](#).
2. In the menu on the left, select **Monitor**.
3. In the **Monitor - Activity log** pane, select **Alerts**.
4. In the **Monitor - Alerts** pane, click **+ Add activity log alert**.
5. Complete the information in the **Add activity log alert** page and make sure you set the following in **Criteria**:
Type: Maintenance **Status:** All (Do not set status to Active or Resolved) **Level:** All

To learn more on how to configure Activity Log Alerts, see [Create activity log alerts](#)

Start Maintenance on your VM from the portal

While looking at the VM details, you will be able to see more maintenance-related details.

At the top of the VM details view, a new notification ribbon will be added if your VM is included in a planned maintenance wave. In addition, a new option is added to start maintenance when possible.

Click on the maintenance notification to see the maintenance page with more details on the planned maintenance. From there you will be able to **start maintenance** on your VM.

Once you start maintenance, your virtual machine will be rebooted and the maintenance status will be updated to reflect the result within few minutes.

If you missed the window where you can start maintenance, you will still be able to see the window when your VM will be rebooted by Azure.

Check maintenance status using PowerShell

You can also use Azure Powershell to see when VMs are scheduled for maintenance. Planned maintenance information is available from the [Get-AzureRmVM](#) cmdlet when you use the `-status` parameter.

Maintenance information is returned only if there is maintenance planned. If there is no maintenance scheduled that impacts the VM, the cmdlet does not return any maintenance information.

```
Get-AzureRmVM -ResourceGroupName rgName -Name vmName -Status
```

The following properties are returned under MaintenanceRedeployStatus:

VALUE	DESCRIPTION
IsCustomerInitiatedMaintenanceAllowed	Indicates whether you can start maintenance on the VM at this time
PreMaintenanceWindowStartTime	The beginning of the maintenance self-service window when you can initiate maintenance on your VM
PreMaintenanceWindowEndTime	The end of the maintenance self-service window when you can initiate maintenance on your VM
MaintenanceWindowStartTime	The beginning of the maintenance scheduled in which Azure initiates maintenance on your VM
MaintenanceWindowEndTime	The end of the maintenance scheduled window in which Azure initiates maintenance on your VM
LastOperationResultCode	The result of the last attempt to initiate maintenance on the VM

You can also get the maintenance status for all VMs in a resource group by using [Get-AzureRmVM](#) and not specifying a VM.

```
Get-AzureRmVM -ResourceGroupName rgName -Status
```

The following PowerShell function takes your subscription ID and prints out a list of VMs that are scheduled for maintenance.

```
function MaintenanceIterator
{
    Select-AzureRmSubscription -SubscriptionId $args[0]

    $rgList= Get-AzureRmResourceGroup

    for ($rgIdx=0; $rgIdx -lt $rgList.Length ; $rgIdx++)
    {
        $rg = $rgList[$rgIdx]
        $vmList = Get-AzureRMVM -ResourceGroupName $rg.ResourceGroupName
        for ($vmIdx=0; $vmIdx -lt $vmList.Length ; $vmIdx++)
        {
            $vm = $vmList[$vmIdx]
            $vmDetails = Get-AzureRMVM -ResourceGroupName $rg.ResourceGroupName -Name $vm.Name -Status
            if ($vmDetails.MaintenanceRedeployStatus )
            {
                Write-Output "VM: $($vmDetails.Name) IsCustomerInitiatedMaintenanceAllowed:
$($vmDetails.MaintenanceRedeployStatus.IsCustomerInitiatedMaintenanceAllowed)
$($vmDetails.MaintenanceRedeployStatus.LastOperationMessage)"
            }
        }
    }
}
```

Start maintenance on your VM using PowerShell

Using information from the function in the previous section, the following starts maintenance on a VM if

IsCustomerInitiatedMaintenanceAllowed is set to true.

```
Restart-AzureRmVM -PerformMaintenance -name $vm.Name -ResourceGroupName $rg.ResourceGroupName
```

Classic deployments

If you still have legacy VMs that were deployed using the classic deployment model, you can use PowerShell to query for VMs and initiate maintenance.

To get the maintenance status of a VM, type:

```
Get-AzureVM -ServiceName <Service name> -Name <VM name>
```

To start maintenance on your classic VM, type:

```
Restart-AzureVM -InitiateMaintenance -ServiceName <service name> -Name <VM name>
```

FAQ

Q: Why do you need to reboot my virtual machines now?

A: While the majority of updates and upgrades to the Azure platform do not impact virtual machine's availability, there are cases where we can't avoid rebooting virtual machines hosted in Azure. We have accumulated several changes which require us to restart our servers which will result in virtual machines reboot.

Q: If I follow your recommendations for High Availability by using an Availability Set, am I safe?

A: Virtual machines deployed in an availability set or virtual machine scale sets have the notion of Update Domains (UD). When performing maintenance, Azure honors the UD constraint and will not reboot virtual machines from different UD (within the same availability set). Azure also waits for at least 30 minutes before moving to the next group of virtual machines.

For more information about high availability, see [Regions and availability for virtual machines in Azure](#).

Q: How do I get notified about planned maintenance?

A: A planned maintenance wave starts by setting a schedule to one or more Azure regions. Soon after, an email notification is sent to the subscription owners (one email per subscription). Additional channels and recipients for this notification could be configured using Activity Log Alerts. In case you deploy a virtual machine to a region where planned maintenance is already scheduled, you will not receive the notification but rather need to check the maintenance state of the VM.

Q: I don't see any indication of planned maintenance in the portal, Powershell, or CLI, What is wrong?

A: Information related to planned maintenance is available during a planned maintenance wave only for the VMs which are going to be impacted by it. In other words, if you see no data, it could be that the maintenance wave has already completed (or not started) or that your virtual machine is already hosted in an updated server.

Q: Is there a way to know exactly when my virtual machine will be impacted?

A: When setting the schedule, we define a time window of several days. However, the exact sequencing of servers (and VMs) within this window is unknown. Customers who would like to know the exact time for their VMs can use [scheduled events](#) and query from within the virtual machine and receive a 15 minute notification before a VM

reboot.

Q: How long will it take you to reboot my virtual machine?

A: Depending on the size of your VM, reboot may take up to several minutes during the self-service maintenance window. During the Azure initiated reboots in the scheduled maintenance window, the reboot will typically take about 25 minutes. Note that in case you use Cloud Services (Web/Worker Role), Virtual Machine Scale Sets, or availability sets, you will be given 30 minutes between each group of VMs (UD) during the scheduled maintenance window.

Q: What is the experience in the case of Cloud Services (Web/Worker Role), Service Fabric, and Virtual Machine Scale Sets?

A: While these platforms are impacted by planned maintenance, customers using these platforms are considered safe given that only VMs in a single Upgrade Domain (UD) will be impacted at any given time. Self-service maintenance is currently not available for Cloud Services (Web/Worker Role), Service Fabric, and Virtual Machine Scale Sets.

Q: I have received an email about hardware decommissioning, is this the same as planned maintenance?

A: While hardware decommissioning is a planned maintenance event, we have not yet onboarded this use case to the new experience.

Q: I don't see any maintenance information on my VMs. What went wrong?

A: There are several reasons why you're not seeing any maintenance information on your VMs:

1. You are using a subscription marked as Microsoft internal.
2. Your VMs are not scheduled for maintenance. It could be that the maintenance wave has ended, canceled or modified so that your VMs are no longer impacted by it.
3. You don't have the **Maintenance** column added to your VM list view. While we have added this column to the default view, customers who configured to see non-default columns must manually add the **Maintenance** column to their VM list view.

Q: My VM is scheduled for maintenance for the second time. Why?

A: There are several use cases where you will see your VM scheduled for maintenance after you have already completed your maintenance-redeploy:

1. We have canceled the maintenance wave and restarted it with a different payload. It could be that we've detected faulted payload and we simply need to deploy an additional payload.
2. Your VM was *service healed* to another node due to a hardware fault
3. You have selected to stop (deallocate) and restart the VM
4. You have **auto shutdown** turned on for the VM

Q: Maintenance of my availability set takes a long time, and I now see "skipped" status on some of my availability set instances. Why?

A: If you have clicked to update multiple instances in an availability set in short succession, Azure will queue these requests and starts to update only the VMs in one update domain (UD) at a time. However, since there might be a pause between update domains, the update might appear to take longer. If the update queue takes longer than 60 minutes, some instances will show the **skipped** state even if they have been updated successfully. To avoid this incorrect status, update your availability sets by clicking only on instance within one availability set and wait for the update on that VM to complete before clicking on the next VM in a different update domain.

Next steps

Learn how you can register for maintenance events from within the VM using [Scheduled Events](#).

Azure Metadata Service: Scheduled Events (Preview) for Windows VMs

12/6/2017 • 6 min to read • [Edit Online](#)

NOTE

Previews are made available to you on the condition that you agree to the terms of use. For more information, see [Microsoft Azure Supplemental Terms of Use for Microsoft Azure Previews](#).

Scheduled Events is an Azure Metadata Service that gives your application time to prepare for virtual machine maintenance. It provides information about upcoming maintenance events (e.g. reboot) so your application can prepare for them and limit disruption. It is available for all Azure Virtual Machine types including PaaS and IaaS on both Windows and Linux.

For information about Scheduled Events on Linux, see [Scheduled Events for Linux VMs](#).

Why Scheduled Events?

Many applications can benefit from time to prepare for virtual machine maintenance. The time can be used to perform application specific tasks that improve availability, reliability, and serviceability including:

- Checkpoint and restore
- Connection draining
- Primary replica failover
- Removal from load balancer pool
- Event logging
- Graceful shutdown

Using Scheduled Events your application can discover when maintenance will occur and trigger tasks to limit its impact.

Scheduled Events provides events in the following use cases:

- Platform initiated maintenance (e.g. Host OS Update)
- User initiated maintenance (e.g. user restarts or redeploys a VM)

The basics

Azure Metadata service exposes information about running Virtual Machines using a REST Endpoint accessible from within the VM. The information is available via a non-routable IP so that it is not exposed outside the VM.

Scope

Scheduled events are delivered to:

- All Virtual Machines in a Cloud Service
- All Virtual Machines in an Availability Set
- All Virtual Machines in a Scale Set Placement Group.

As a result, you should check the `Resources` field in the event to identify which VMs are going to be impacted.

Discovering the endpoint

For VNET enabled VMs, the full endpoint for the latest version of Scheduled Events is:

```
http://169.254.169.254/metadata/scheduledevents?api-version=2017-08-01
```

In the case where a Virtual Machine is created within a Virtual Network (VNet), the metadata service is available from a static non-routable IP, `169.254.169.254`. If the Virtual Machine is not created within a Virtual Network, the default cases for cloud services and classic VMs, additional logic is required to discover the IP address to use. Refer to this sample to learn how to [discover the host endpoint](#).

Versioning

The Scheduled Events Service is versioned. Versions are mandatory and the current version is `2017-08-01`.

VERSION	RELEASE NOTES
2017-08-01	<ul style="list-style-type: none">Removed prepended underscore from resource names for IaaS VMsMetadata Header requirement enforced for all requests
2017-03-01	<ul style="list-style-type: none">Public Preview Version

NOTE

Previous preview releases of scheduled events supported `{latest}` as the api-version. This format is no longer supported and will be deprecated in the future.

Using headers

When you query the Metadata Service, you must provide the header `Metadata:true` to ensure the request was not unintentionally redirected. The `Metadata:true` header is required for all scheduled events requests. Failure to include the header in the request will result in a Bad Request response from the Metadata Service.

Enabling Scheduled Events

The first time you make a request for scheduled events, Azure implicitly enables the feature on your Virtual Machine. As a result, you should expect a delayed response in your first call of up to two minutes.

NOTE

Scheduled Events is automatically disabled for your service if your service doesn't call the end point for 1 day. Once Scheduled Events is disabled for your service, there will be no events created for user initiated maintenance.

User initiated maintenance

User initiated virtual machine maintenance via the Azure portal, API, CLI, or PowerShell results in a scheduled event. This allows you to test the maintenance preparation logic in your application and allows your application to prepare for user initiated maintenance.

Restarting a virtual machine schedules an event with type `Reboot`. Redeploying a virtual machine schedules an event with type `Redeploy`.

NOTE

Currently a maximum of 100 user initiated maintenance operations can be simultaneously scheduled.

NOTE

Currently user initiated maintenance resulting in Scheduled Events is not configurable. Configurability is planned for a future release.

Using the API

Query for events

You can query for Scheduled Events simply by making the following call:

Powershell

```
curl http://169.254.169.254/metadata/scheduledevents?api-version=2017-08-01 -H @{"Metadata"="true"}
```

A response contains an array of scheduled events. An empty array means that there are currently no events scheduled. In the case where there are scheduled events, the response contains an array of events:

```
{
  "DocumentIncarnation": {IncarnationID},
  "Events": [
    {
      "EventId": {eventID},
      "EventType": "Reboot" | "Redeploy" | "Freeze",
      "ResourceType": "VirtualMachine",
      "Resources": [{resourceName}],
      "EventStatus": "Scheduled" | "Started",
      "NotBefore": {timeInUTC},
    }
  ]
}
```

Event properties

PROPERTY	DESCRIPTION
EventId	Globally unique identifier for this event. Example: <ul style="list-style-type: none">• 602d9444-d2cd-49c7-8624-8643e7171297
EventType	Impact this event causes. Values: <ul style="list-style-type: none">• Freeze : The Virtual Machine is scheduled to pause for few seconds. The CPU is suspended, but there is no impact on memory, open files, or network connections.• Reboot : The Virtual Machine is scheduled for reboot (non-persistent memory is lost).• Redeploy : The Virtual Machine is scheduled to move to another node (ephemeral disks are lost).

PROPERTY	DESCRIPTION
ResourceType	Type of resource this event impacts. Values: <ul style="list-style-type: none">VirtualMachine
Resources	List of resources this event impacts. This is guaranteed to contain machines from at most one Update Domain , but may not contain all machines in the UD. Example: <ul style="list-style-type: none">["FrontEnd_IN_0", "BackEnd_IN_0"]
Event Status	Status of this event. Values: <ul style="list-style-type: none">Scheduled : This event is scheduled to start after the time specified in the <code>NotBefore</code> property.Started : This event has started. No <code>Completed</code> or similar status is ever provided; the event will no longer be returned when the event is completed.
NotBefore	Time after which this event may start. Example: <ul style="list-style-type: none">2016-09-19T18:29:47Z

Event scheduling

Each event is scheduled a minimum amount of time in the future based on event type. This time is reflected in an event's `NotBefore` property.

EVENTTYPE	MINIMUM NOTICE
Freeze	15 minutes
Reboot	15 minutes
Redeploy	10 minutes

Starting an event

Once you have learned of an upcoming event and completed your logic for graceful shutdown, you can approve the outstanding event by making a `POST` call to the metadata service with the `EventId`. This indicates to Azure that it can shorten the minimum notification time (when possible).

The following is the json expected in the `POST` request body. The request should contain a list of `StartRequests`. Each `StartRequest` contains the `EventId` for the event you want to expedite:

```
{  
    "StartRequests" : [  
        {  
            "EventId": {"EventId}  
        }  
    ]  
}
```

Powershell

```
curl -H @{"Metadata"="true"} -Method POST -Body '{"DocumentIncarnation":"5", "StartRequests": [{"EventId": "f020ba2e-3bc0-4c40-a10b-86575a9eabd5"}]}' -Uri http://169.254.169.254/metadata/scheduledevents?api-version=2017-08-01
```

NOTE

Acknowledging an event allows the event to proceed for all Resources in the event, not just the virtual machine that acknowledges the event. You may therefore choose to elect a leader to coordinate the acknowledgement, which may be as simple as the first machine in the Resources field.

PowerShell sample

The following sample queries the metadata service for scheduled events and approves each outstanding event.

```

# How to get scheduled events
function Get-ScheduledEvents($uri)
{
    $scheduledEvents = Invoke-RestMethod -Headers @{"Metadata"="true"} -URI $uri -Method get
    $json = ConvertTo-Json $scheduledEvents
    Write-Host "Received following events: `n" $json
    return $scheduledEvents
}

# How to approve a scheduled event
function Approve-ScheduledEvent($eventId, $docIncarnation, $uri)
{
    # Create the Scheduled Events Approval Document
    $startRequests = [array]@{"EventId" = $eventId}
    $scheduledEventsApproval = @{"StartRequests" = $startRequests; "DocumentIncarnation" = $docIncarnation}

    # Convert to JSON string
    $approvalString = ConvertTo-Json $scheduledEventsApproval

    Write-Host "Approving with the following: `n" $approvalString

    # Post approval string to scheduled events endpoint
    Invoke-RestMethod -Uri $uri -Headers @{"Metadata"="true"} -Method POST -Body $approvalString
}

function Handle-ScheduledEvents($scheduledEvents)
{
    # Add logic for handling events here
}

#####
# Sample Scheduled Events Interaction #####
#####

# Set up the scheduled events URI for a VNET-enabled VM
$localhostIP = "169.254.169.254"
$scheduledEventURI = 'http://{0}/metadata/scheduledevents?api-version=2017-03-01' -f $localhostIP

# Get events
$scheduledEvents = Get-ScheduledEvents $scheduledEventURI

# Handle events however is best for your service
Handle-ScheduledEvents $scheduledEvents

# Approve events when ready (optional)
foreach($event in $scheduledEvents.Events)
{
    Write-Host "Current Event: `n" $event
    $entry = Read-Host "`nApprove event? Y/N"
    if($entry -eq "Y" -or $entry -eq "y")
    {
        Approve-ScheduledEvent $event.EventId $scheduledEvents.DocumentIncarnation $scheduledEventURI
    }
}

```

Next steps

- Review the Scheduled Events code samples in the [Azure Instance Metadata Scheduled Events Github Repository](#)
- Read more about the APIs available in the [Instance Metadata service](#).
- Learn about [planned maintenance for Windows virtual machines in Azure](#).

Azure Instance Metadata service

10/11/2017 • 7 min to read • [Edit Online](#)

The Azure Instance Metadata Service provides information about running virtual machine instances that can be used to manage and configure your virtual machines. This includes information such as SKU, network configuration, and upcoming maintenance events. For more information on what type of information is available, see [metadata categories](#).

Azure's Instance Metadata Service is a REST Endpoint accessible to all IaaS VMs created via the [Azure Resource Manager](#). The endpoint is available at a well-known non-routable IP address (`169.254.169.254`) that can be accessed only from within the VM.

IMPORTANT

This service is **generally available** in all Azure Regions. It regularly receives updates to expose new information about virtual machine instances. This page reflects the up-to-date [data categories](#) available.

Service availability

The service is available in all generally available all Azure regions. Not all API version may be available in all Azure Regions.

REGIONS	AVAILABILITY?	SUPPORTED VERSIONS
All Generally Available Global Azure Regions	Generally Available	2017-04-02, 2017-08-01
Azure Government	Generally Available	2017-04-02
Azure China	Generally Available	2017-04-02
Azure Germany	Generally Available	2017-04-02

This table is updated when there are service updates and or new supported versions are available

To try out the Instance Metadata Service, create a VM from [Azure Resource Manager](#) or the [Azure portal](#) in the above regions and follow the examples below.

Usage

Versioning

The Instance Metadata Service is versioned. Versions are mandatory and the current version on Global Azure is `2017-08-01`. Current supported versions are (`2017-04-02, 2017-08-01`)

NOTE

Previous preview releases of scheduled events supported `{latest}` as the api-version. This format is no longer supported and will be deprecated in the future.

As we add newer versions, older versions can still be accessed for compatibility if your scripts have dependencies on specific data formats. However, note that the previous preview version(2017-03-01) may not be available once the service is generally available.

Using headers

When you query the Instance Metadata Service, you must provide the header `Metadata: true` to ensure the request was not unintentionally redirected.

Retrieving metadata

Instance metadata is available for running VMs created/managed using [Azure Resource Manager](#). Access all data categories for a virtual machine instance using the following request:

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance?api-version=2017-04-02"
```

NOTE

All instance metadata queries are case-sensitive.

Data output

By default, the Instance Metadata Service returns data in JSON format (`Content-Type: application/json`). However, different APIs return data in different formats if requested. The following table is a reference of other data formats APIs may support.

API	DEFAULT DATA FORMAT	OTHER FORMATS
/instance	json	text
/scheduledevents	json	none

To access a non-default response format, specify the requested format as a querystring parameter in the request. For example:

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance?api-version=2017-04-02&format=text"
```

Security

The Instance Metadata Service endpoint is accessible only from within the running virtual machine instance on a non-routable IP address. In addition, any request with a `X-Forwarded-For` header is rejected by the service. We also require requests to contain a `Metadata: true` header to ensure that the actual request was directly intended and not a part of unintentional redirection.

Error

If there is a data element not found or a malformed request, the Instance Metadata Service returns standard HTTP errors. For example:

HTTP STATUS CODE	REASON
200 OK	
400 Bad Request	Missing <code>Metadata: true</code> header
404 Not Found	The requested element doesn't exist

HTTP STATUS CODE	REASON
405 Method Not Allowed	Only <code>GET</code> and <code>POST</code> requests are supported
429 Too Many Requests	The API currently supports a maximum of 5 queries per second
500 Service Error	Retry after some time

Examples

NOTE

All API responses are JSON strings. All following example responses are pretty-printed for readability.

Retrieving network information

Request

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance/network?api-version=2017-08-01"
```

Response

NOTE

The response is a JSON string. The following example response is pretty-printed for readability.

```
{
  "interface": [
    {
      "ipv4": {
        "ipAddress": [
          {
            "privateIpAddress": "10.1.0.4",
            "publicIpAddress": "X.X.X.X"
          }
        ],
        "subnet": [
          {
            "address": "10.1.0.0",
            "prefix": "24"
          }
        ]
      },
      "ipv6": {
        "ipAddress": []
      },
      "macAddress": "000D3AF806EC"
    }
  ]
}
```

Retrieving public IP address

```
curl -H Metadata:true
"http://169.254.169.254/metadata/instance/network/interface/0/ipv4/ipAddress/0/publicIpAddress?api-version=2017-04-02&format=text"
```

Retrieving all metadata for an instance

Request

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance?api-version=2017-08-01"
```

Response

NOTE

The response is a JSON string. The following example response is pretty-printed for readability.

```
{
  "compute": {
    "location": "westus",
    "name": "avset2",
    "offer": "UbuntuServer",
    "osType": "Linux",
    "placementGroupId": "",
    "platformFaultDomain": "1",
    "platformUpdateDomain": "1",
    "publisher": "Canonical",
    "resourceGroupName": "myrg",
    "sku": "16.04-LTS",
    "subscriptionId": "xxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx",
    "tags": "",
    "version": "16.04.201708030",
    "vmId": "13f56399-bd52-4150-9748-7190aae1ff21",
    "vmSize": "Standard_D1"
  },
  "network": {
    "interface": [
      {
        "ipv4": {
          "ipAddress": [
            {
              "privateIpAddress": "10.1.2.5",
              "publicIpAddress": "X.X.X.X"
            }
          ],
          "subnet": [
            {
              "address": "10.1.2.0",
              "prefix": "24"
            }
          ]
        },
        "ipv6": {
          "ipAddress": []
        },
        "macAddress": "000D3A36DDDE"
      }
    ]
  }
}
```

Retrieving metadata in Windows Virtual Machine

Request

Instance metadata can be retrieved in Windows via the PowerShell utility `curl`:

```
curl -H @{@"Metadata">'true'} http://169.254.169.254/metadata/instance?api-version=2017-04-02 | select -ExpandProperty Content
```

Or through the `Invoke-RestMethod` cmdlet:

```
Invoke-RestMethod -Headers @{"Metadata"="true"} -URI http://169.254.169.254/metadata/instance?api-version=2017-04-02 -Method get
```

Response

NOTE

The response is a JSON string. The following example response is pretty-printed for readability.

```
{
  "compute": {
    "location": "westus",
    "name": "SQLTest",
    "offer": "SQL2016SP1-WS2016",
    "osType": "Windows",
    "platformFaultDomain": "0",
    "platformUpdateDomain": "0",
    "publisher": "MicrosoftSQLServer",
    "sku": "Enterprise",
    "version": "13.0.400110",
    "vmId": "453945c8-3923-4366-b2d3-ea4c80e9b70e",
    "vmSize": "Standard_DS2"
  },
  "network": {
    "interface": [
      {
        "ipv4": {
          "ipAddress": [
            {
              "privateIpAddress": "10.0.1.4",
              "publicIpAddress": "X.X.X.X"
            }
          ],
          "subnet": [
            {
              "address": "10.0.1.0",
              "prefix": "24"
            }
          ]
        },
        "ipv6": {
          "ipAddress": [
            {}
          ],
          "macAddress": "002248020E1E"
        }
      ]
    }
  }
}
```

Instance metadata data categories

The following data categories are available through the Instance Metadata Service:

DATA	DESCRIPTION	VERSION INTRODUCED
location	Azure Region the VM is running in	2017-04-02
name	Name of the VM	2017-04-02
offer	Offer information for the VM image. This value is only present for images deployed from Azure image gallery.	2017-04-02
publisher	Publisher of the VM image	2017-04-02
sku	Specific SKU for the VM image	2017-04-02
version	Version of the VM image	2017-04-02
osType	Linux or Windows	2017-04-02
platformUpdateDomain	Update domain the VM is running in	2017-04-02
platformFaultDomain	Fault domain the VM is running in	2017-04-02
vmlId	Unique identifier for the VM	2017-04-02
vmSize	VM size	2017-04-02
subscriptionId	Azure subscription for the Virtual Machine	2017-08-01
tags	Tags for your Virtual Machine	2017-08-01
resourceGroupName	Resource group for your Virtual Machine	2017-08-01
placementGroupId	Placement Group of your Virtual Machine Scale set	2017-08-01
ipv4/privateIpAddress	Local IPv4 address of the VM	2017-04-02
ipv4/publicIpAddress	Public IPv4 address of the VM	2017-04-02
subnet/address	Subnet address of the VM	2017-04-02
subnet/prefix	Subnet prefix, example 24	2017-04-02
ipv6/ipAddress	Local IPv6 address of the VM	2017-04-02
macAddress	VM mac address	2017-04-02
scheduledevents	Currently in Public Preview See scheduledevents	2017-03-01

Example scenarios for usage

Tracking VM running on Azure

As a service provider, you may require to track the number of VMs running your software or have agents that need to track uniqueness of the VM. To be able to get a unique ID for a VM, use the `vmId` field from Instance Metadata Service.

Request

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance/compute/vmId?api-version=2017-04-02&format=text"
```

Response

```
5c08b38e-4d57-4c23-ac45-aca61037f084
```

Placement of containers, data-partitions based fault/update domain

For certain scenarios, placement of different data replicas is of prime importance. For example, [HDFS replica placement](#) or container placement via an [orchestrator](#) may you require to know the `platformFaultDomain` and `platformUpdateDomain` the VM is running on. You can query this data directly via the Instance Metadata Service.

Request

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance/compute/platformFaultDomain?api-version=2017-04-02&format=text"
```

Response

```
0
```

Getting more information about the VM during support case

As a service provider, you may get a support call where you would like to know more information about the VM. Asking the customer to share the compute metadata can provide basic information for the support professional to know about the kind of VM on Azure.

Request

```
curl -H Metadata:true "http://169.254.169.254/metadata/instance/compute?api-version=2017-04-02"
```

Response

NOTE

The response is a JSON string. The following example response is pretty-printed for readability.

```
{
  "compute": {
    "location": "CentralUS",
    "name": "IMDSCanary",
    "offer": "RHEL",
    "osType": "Linux",
    "platformFaultDomain": "0",
    "platformUpdateDomain": "0",
    "publisher": "RedHat",
    "sku": "7.2",
    "version": "7.2.20161026",
    "vmId": "5c08b38e-4d57-4c23-ac45-aca61037f084",
    "vmSize": "Standard_DS2"
  }
}
```

Examples of calling metadata service using different languages inside the VM

LANGUAGE	EXAMPLE
Ruby	https://github.com/Microsoft/azureimds/blob/master/IMDSSample.rb
Go Lang	https://github.com/Microsoft/azureimds/blob/master/imdssample.go
Python	https://github.com/Microsoft/azureimds/blob/master/IMDSSample.py
C++	https://github.com/Microsoft/azureimds/blob/master/IMDSSample-windows.cpp
C#	https://github.com/Microsoft/azureimds/blob/master/IMDSSample.cs
JavaScript	https://github.com/Microsoft/azureimds/blob/master/IMDSSample.js
PowerShell	https://github.com/Microsoft/azureimds/blob/master/IMDSSample.ps1
Bash	https://github.com/Microsoft/azureimds/blob/master/IMDSSample.sh

FAQ

- I am getting the error `400 Bad Request, Required metadata header not specified`. What does this mean?
 - The Instance Metadata Service requires the header `Metadata: true` to be passed in the request. Passing this header in the REST call allows access to the Instance Metadata Service.
- Why am I not getting compute information for my VM?
 - Currently the Instance Metadata Service only supports instances created with Azure Resource Manager. In the future, we may add support for Cloud Service VMs.
- I created my Virtual Machine through Azure Resource Manager a while back. Why am I not see compute metadata information?
 - For any VMs created after Sep 2016, add a `Tag` to start seeing compute metadata. For older VMs (created before Sep 2016), add/remove extensions or data disks to the VM to refresh metadata.

4. I am not seeing all data populated for new version of 2017-08-01
 - For any VMs created after Sep 2016, add a [Tag](#) to start seeing compute metadata. For older VMs (created before Sep 2016), add/remove extensions or data disks to the VM to refresh metadata.
5. Why am I getting the error [500 Internal Server Error](#)?
 - Retry your request based on exponential back off system. If the issue persists contact Azure support.
6. Where do I share additional questions/comments?
 - Send your comments on <http://feedback.azure.com>.
7. Would this work for Virtual Machine Scale Set Instance?
 - Yes Metadata service is available for Scale Set Instances.
8. How do I get support for the service?
 - To get support for the service, create a support issue in Azure portal for the VM where you are not able to get metadata response after long retries

Problem

NEW SUPPORT REQUEST



* Severity i

C - Minimal impact



* Problem type

Management



* Category

✓ Choose a category

Backup

Cannot stop, start, or restart a VM

Capacity issues that are related to SAP HANA large instances

Instance Metadata Service

Manage an Exchange Server

Manage an instance of SQL Server

Manage encrypted disks, keys or secrets, or permissions

Manage or use RDS in Azure

Manage or use a VPN

Manage or use a cluster in Azure

Manage or use a virtual network

Manage or use endpoints

Unable to delete a virtual machine

Virtual machine restarts

When did the problem start?

Choose a date



Enter a local time

File upload i

Select a file



Share diagnostic information i

[Learn more about the information we collect](#)

Next

Next steps

- Learn more about the [Scheduled Events API in public preview](#) provided by the Instance Metadata service.

How to enable nested virtualization in an Azure VM

12/7/2017 • 5 min to read • [Edit Online](#)

Nested virtualization is supported in the Dv3 and Ev3 series of Azure virtual machines. This capability provides great flexibility in supporting scenarios such as development, testing, training, and demonstration environments.

This article steps through enabling nested virtualization on an Azure VM and configuring Internet connectivity to that guest virtual machine.

Create a Dv3 or Ev3 series Azure VM

Create a new Windows Server 2016 Azure VM and choose a size from the Dv3 or Ev3 series. Ensure you choose a size large enough to support the demands of a guest virtual machine. In this example, we are using a D3_v3 size Azure VM.

You can view the regional availability of Dv3 or Ev3 series virtual machines [here](#).

NOTE

For detailed instructions on creating a new virtual machine, see [Create and Manage Windows VMs with the Azure PowerShell module](#)

Connect to your Azure VM

Create a remote desktop connection to the virtual machine.

1. Click the **Connect** button on the virtual machine properties. A Remote Desktop Protocol file (.rdp file) is created and downloaded.
2. To connect to your VM, open the downloaded RDP file. If prompted, click **Connect**. On a Mac, you need an RDP client such as this [Remote Desktop Client](#) from the Mac App Store.
3. Enter the user name and password you specified when creating the virtual machine, then click **Ok**.
4. You may receive a certificate warning during the sign-in process. Click **Yes** or **Continue** to proceed with the connection.

Enable the Hyper-V feature on the Azure VM

You can configure these settings manually or we have provided a PowerShell script to automate the configuration.

Option 1: Use a PowerShell script to configure nested virtualization

A PowerShell script to enable nested virtualization on a Windows Server 2016 host is available on [GitHub](#). The script checks pre-requisites and then configures nested virtualization on the Azure VM. A restart of the Azure VM is necessary to complete the configuration. This script may work in other environments but is not guaranteed. Check out the Azure blog post with a live video demonstration on nested virtualization running on Azure!

<https://aka.ms/AzureNVblog>.

Option 2: Configure nested virtualization manually

1. On the Azure VM, open PowerShell as an Administrator.
2. Enable the Hyper-V feature and Management Tools.

```
Install-WindowsFeature -Name Hyper-V -IncludeManagementTools -Restart
```

WARNING

This command restarts the Azure VM. You will lose your RDP connection during the restart process.

- After the Azure VM restarts, reconnect to your VM using RDP.

Set up internet connectivity for the guest virtual machine

Create a new virtual network adapter for the guest virtual machine and configure a NAT Gateway to enable Internet connectivity.

Create a NAT virtual network switch

- On the Azure VM, open PowerShell as an Administrator.
- Create an internal switch.

```
New-VMSwitch -SwitchName "InternalNATSwitch" -SwitchType Internal
```

- View the properties of the switch and note the ifIndex for the new adapter.

```
Get-NetAdapter
```

Name	InterfaceDescription	ifIndex	Status	MacAddress	LinkSpeed
vEthernet (InternalNAT)	Hyper-V Virtual Ethernet Adapter	13	Up	00-15-5D-01-04-00	10 Gbps
Ethernet 3	Microsoft Hyper-V Network Adapter #3	4	Up	00-0D-3A-F9-AC-5A	40 Gbps

NOTE

Take note of the "ifIndex" for the virtual switch you just created.

- Create an IP address for the NAT Gateway.

In order to configure the gateway, you need some information about your network:

- IPAddress - The NAT Gateway IP specifies the IPv4 or IPv6 address to use as the default gateway address for the virtual network subnet. The generic form is a.b.c.1 (for example, "192.168.0.1"). While the final position doesn't have to be .1, it usually is (based on prefix length). Typically you should use an RFC 1918 private network address space.
- PrefixLength - The subnet prefix length defines the local subnet size (subnet mask). The subnet prefix length will be an integer value between 0 and 32. 0 would map the entire internet, 32 would only allow one mapped IP. Common values range from 24 to 12 depending on how many IPs need to be attached to the NAT. A common PrefixLength is 24 -- this is a subnet mask of 255.255.255.0.
- InterfaceIndex - **ifIndex** is the interface index of the virtual switch created in the previous step.

```
New-NetIPAddress -IPAddress 192.168.0.1 -PrefixLength 24 -InterfaceIndex 13
```

Create the NAT network

In order to configure the gateway, you will need to provide information about the network and NAT Gateway:

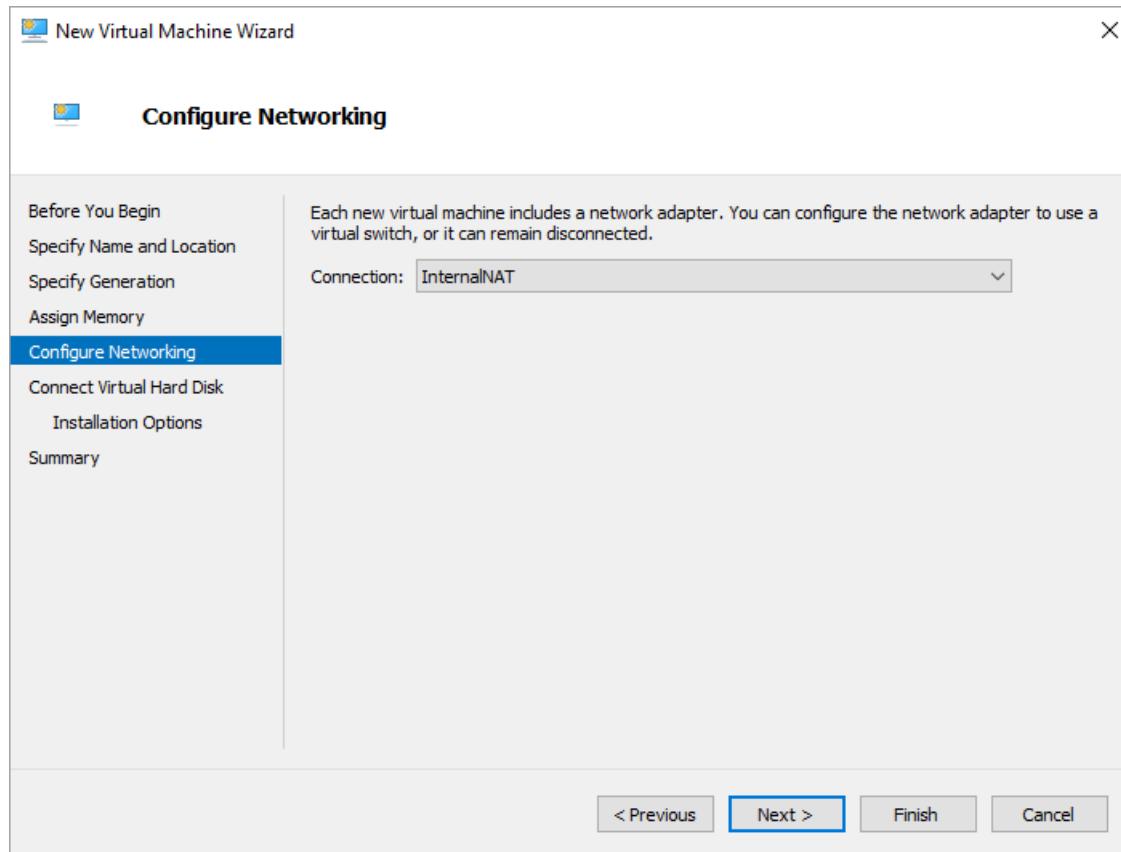
- Name - This is the name of the NAT network.
- InternalIPInterfaceAddressPrefix - The NAT subnet prefix describes both the NAT Gateway IP prefix from above as well as the NAT Subnet Prefix Length from above. The generic form will be a.b.c.0/NAT Subnet Prefix Length.

In PowerShell, create a new NAT network.

```
New-NetNat -Name "InternalNat" -InternalIPInterfaceAddressPrefix 192.168.0.0/24
```

Create the guest virtual machine

1. Open Hyper-V Manager and create a new virtual machine. Configure the virtual machine to use the new Internal network you created.



2. Install an operating system on the guest virtual machine.

NOTE

You need installation media for an operating system to install on the VM. In this case we are using Windows 10 Enterprise.

Assign an IP address to the guest virtual machine

You can assign an IP address to the guest virtual machine either by manually setting a static IP address on the guest virtual machine or configuring DHCP on the Azure VM to assign the IP address dynamically.

Option 1: Configure DHCP to dynamically assign an IP address to the guest virtual machine

Follow the steps below to configure DHCP on the host virtual machine for dynamic address assignment.

Install DHCP Server on the Azure VM

1. Open Server Manager. On the Dashboard, click **Add roles and features**. The Add Roles and Features Wizard appears.
2. In wizard, click **Next** until the Server Roles page.
3. Click to select the **DHCP Server** checkbox, click **Add Features**, and then click **Next** until you complete the wizard.
4. Click **Install**.

Configure a new DHCP scope

1. Open DHCP Manager.
2. In the navigation pane, expand the server name, right-click **IPv4**, and click **New Scope**. The New Scope Wizard appears, click **Next**.
3. Enter a Name and Description for the scope and click **Next**.
4. Define an IP Range for your DHCP Server (for example, 192.168.0.100 to 192.168.0.200).
5. Click **Next** until the Default Gateway page. Enter the IP Address you created earlier (for example, 192.168.0.1) as the Default Gateway.
6. Click **Next** until the wizard completes, leaving all default values, then click **Finish**.

Option 2: Manually set a static IP address on the guest virtual machine

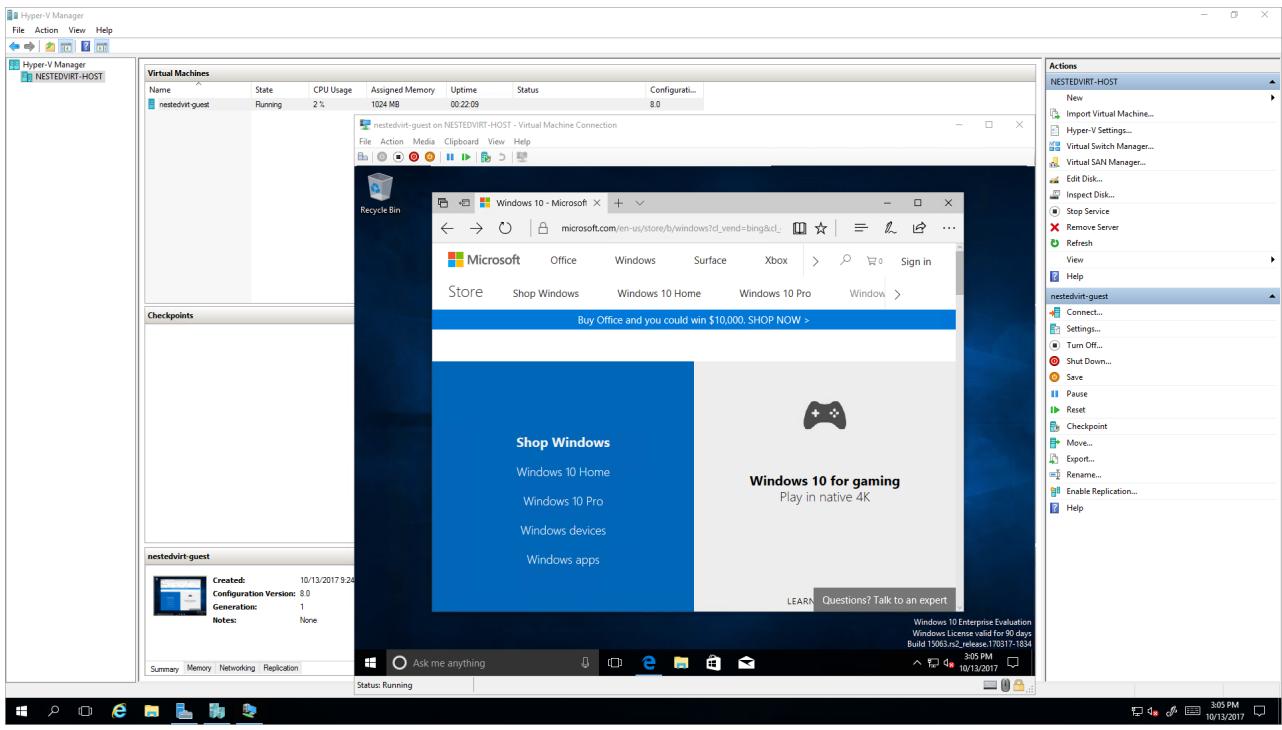
If you did not configure DHCP to dynamically assign an IP address to the guest virtual machine, follow these steps to set a static IP address.

1. On the Azure VM, open PowerShell as an Administrator.
2. Right-click the guest virtual machine and click Connect.
3. Log on to the guest virtual machine.
4. On the guest virtual machine, open the Network and Sharing Center.
5. Configure the network adapter for an address within the range of the NAT network you created in the previous section.

In this example you will use an address in the 192.168.0.0/24 range.

Test connectivity in guest virtual machine

In the guest virtual machine, open your browser and navigate to a web page.



How to find Windows VM images in the Azure Marketplace with Azure PowerShell

10/2/2017 • 2 min to read • [Edit Online](#)

This topic describes how to use Azure PowerShell to find VM images in the Azure Marketplace. Use this information to specify a Marketplace image when you create a Windows VM.

Make sure that you installed and configured the latest [Azure PowerShell module](#).

Table of commonly used Windows images

PublisherName	Offer	SKU
MicrosoftWindowsServer	WindowsServer	2016-Datacenter
MicrosoftWindowsServer	WindowsServer	2016-Datacenter-Server-Core
MicrosoftWindowsServer	WindowsServer	2016-Datacenter-with-Containers
MicrosoftWindowsServer	WindowsServer	2016-Nano-Server
MicrosoftWindowsServer	WindowsServer	2012-R2-Datacenter
MicrosoftWindowsServer	WindowsServer	2008-R2-SP1
MicrosoftDynamicsNAV	DynamicsNAV	2017
MicrosoftSharePoint	MicrosoftSharePointServer	2016
MicrosoftSQLServer	SQL2016-WS2016	Enterprise
MicrosoftSQLServer	SQL2014SP2-WS2012R2	Enterprise
MicrosoftWindowsServerHPCPack	WindowsServerHPCPack	2012R2
MicrosoftWindowsServerEssentials	WindowsServerEssentials	WindowsServerEssentials

Find specific images

When creating a new virtual machine with Azure Resource Manager, in some cases you need to specify an image with the combination of the following image properties:

- Publisher
- Offer
- SKU

For example, use these values with the [Set-AzureRMVMSourceImage](#) PowerShell cmdlet, or with a resource group template in which you must specify the type of VM to be created.

If you need to determine these values, you can run the [Get-AzureRMVMImagePublisher](#), [Get-AzureRMVMImageOffer](#), and [Get-AzureRMVMImageSku](#) cmdlets to navigate the images. You determine these values:

1. List the image publishers.
2. For a given publisher, list their offers.
3. For a given offer, list their SKUs.

First, list the publishers with the following commands:

```
$locName=<Azure location, such as West US>
Get-AzureRMVMImagePublisher -Location $locName | Select PublisherName
```

Fill in your chosen publisher name and run the following commands:

```
$pubName=<publisher>
Get-AzureRMVMImageOffer -Location $locName -Publisher $pubName | Select Offer
```

Fill in your chosen offer name and run the following commands:

```
$offerName=<offer>
Get-AzureRMVMImageSku -Location $locName -Publisher $pubName -Offer $offerName | Select Skus
```

From the output of the [Get-AzureRMVMImageSku](#) command, you have all the information you need to specify the image for a new virtual machine.

The following shows a full example:

```
$locName="West US"
Get-AzureRMVMImagePublisher -Location $locName | Select PublisherName
```

Output:

```
PublisherName
-----
a10networks
aiscaler-cache-control-ddos-and-url-rewriting-
alertlogic
AlertLogic.Extension
Barracuda.Azure.ConnectivityAgent
barracudanetworks
basho
boxless
bssw
Canonical
...
```

For the "MicrosoftWindowsServer" publisher:

```
$pubName="MicrosoftWindowsServer"
Get-AzureRMVMImageOffer -Location $locName -Publisher $pubName | Select Offer
```

Output:

```
Offer
-----
Windows-HUB
WindowsServer
WindowsServer-HUB
```

For the "WindowsServer" offer:

```
$offerName="WindowsServer"
Get-AzureRMVMImageSku -Location $locName -Publisher $pubName -Offer $offerName | Select Skus
```

Output:

```
Skus
-----
2008-R2-SP1
2008-R2-SP1-smalldisk
2012-Datacenter
2012-Datacenter-smalldisk
2012-R2-Datacenter
2012-R2-Datacenter-smalldisk
2016-Datacenter
2016-Datacenter-Server-Core
2016-Datacenter-Server-Core-smalldisk
2016-Datacenter-smalldisk
2016-Datacenter-with-Containers
2016-Nano-Server
```

From this list, copy the chosen SKU name, and you have all the information for the `Set-AzureRMVMSourceImage` PowerShell cmdlet or for a resource group template.

Next steps

Now you can choose precisely the image you want to use. To create a virtual machine quickly by using the image information, which you just found, see [Create a Windows virtual machine with PowerShell](#).

Prepare a Windows VHD or VHDX to upload to Azure

11/3/2017 • 15 min to read • [Edit Online](#)

Before you upload a Windows virtual machines (VM) from on-premises to Microsoft Azure, you must prepare the virtual hard disk (VHD or VHDX). Azure supports only generation 1 VMs that are in the VHD file format and have a fixed sized disk. The maximum size allowed for the VHD is 1,023 GB. You can convert a generation 1 VM from the VHDX file system to VHD and from a dynamically expanding disk to fixed-sized. But you can't change a VM's generation. For more information, see [Should I create a generation 1 or 2 VM in Hyper-V](#).

For more information about the support policy for Azure VM, see [Microsoft server software support for Microsoft Azure VMs](#).

NOTE

The instructions in this article apply to the 64-bit version of Windows Server 2008 R2 and later Windows server operating system. For information about running 32-bit version of operating system in Azure, see [Support for 32-bit operating systems in Azure virtual machines](#).

Convert the virtual disk to VHD and fixed size disk

If you need to convert your virtual disk to the required format for Azure, use one of the methods in this section. Back up the VM before you run the virtual disk conversion process and make sure that the Windows VHD works correctly on the local server. Resolve any errors within the VM itself before you try to convert or upload it to Azure.

After you convert the disk, create a VM that uses the converted disk. Start and sign in to the VM to finish preparing the VM for upload.

Convert disk using Hyper-V Manager

1. Open Hyper-V Manager and select your local computer on the left. In the menu above the computer list, click **Action > Edit Disk**.
2. On the **Locate Virtual Hard Disk** screen, locate and select your virtual disk.
3. On the **Choose Action** screen, and then select **Convert** and **Next**.
4. If you need to convert from VHDX, select **VHD** and then click **Next**
5. If you need to convert from a dynamically expanding disk, select **Fixed size** and then click **Next**
6. Locate and select a path to save the new VHD file to.
7. Click **Finish**.

NOTE

The commands in this article must be run on an elevated PowerShell session.

Convert disk by using PowerShell

You can convert a virtual disk by using the [Convert-VHD](#) command in Windows PowerShell. Select **Run as administrator** when you start PowerShell.

The following example command converts from VHDX to VHD, and from a dynamically expanding disk to fixed-

size:

```
Convert-VHD -Path c:\test\MY-VM.vhdx -DestinationPath c:\test\MY-NEW-VM.vhd -VHDTtype Fixed
```

In this command, replace the value for "-Path" with the path to the virtual hard disk that you want to convert and the value for "-DestinationPath" with the new path and name of the converted disk.

Convert from VMware VMDK disk format

If you have a Windows VM image in the [VMDK file format](#), convert it to a VHD by using the [Microsoft VM Converter](#). For more information, see the blog article [How to Convert a VMware VMDK to Hyper-V VHD](#).

Set Windows configurations for Azure

On the VM that you plan to upload to Azure, run all commands in the following steps from an [elevated Command Prompt window](#):

1. Remove any static persistent route on the routing table:

- To view the route table, run `route print` at the command prompt.
- Check the **Persistence Routes** sections. If there is a persistent route, use `route delete` to remove it.

2. Remove the WinHTTP proxy:

```
netsh winhttp reset proxy
```

3. Set the disk SAN policy to [Onlineall](#).

```
diskpart
```

In the open Command Prompt window, type the following commands:

```
san policy=onlineall  
exit
```

4. Set Coordinated Universal Time (UTC) time for Windows and the startup type of the Windows Time (w32time) service to **Automatically**:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\TimeZoneInformation' -name  
"RealTimeIsUniversal" 1 -Type DWord  
  
Set-Service -Name w32time -StartupType Auto
```

5. Set the power profile to the **High Performance**:

```
powercfg /setactive SCHEME_MIN
```

Check the Windows services

Make sure that each of the following Windows services is set to the **Windows default values**. These are the minimum numbers of services that must be set up to make sure that the VM has connectivity. To reset the startup settings, run the following commands:

```
Set-Service -Name bfe -StartupType Auto
Set-Service -Name dhcp -StartupType Auto
Set-Service -Name dnscache -StartupType Auto
Set-Service -Name IKEEXT -StartupType Auto
Set-Service -Name iphlpsvc -StartupType Auto
Set-Service -Name netlogon -StartupType Manual
Set-Service -Name netman -StartupType Manual
Set-Service -Name nsi -StartupType Auto
Set-Service -Name termService -StartupType Manual
Set-Service -Name MpsSvc -StartupType Auto
Set-Service -Name RemoteRegistry -StartupType Auto
```

Update Remote Desktop registry settings

Make sure that the following settings are configured correctly for remote desktop connection:

NOTE

You may receive an error message when you run the **Set-ItemProperty -Path 'HKLM:\SOFTWARE\ Policies\Microsoft\Windows NT\Terminal Services -name <object name> <value>** in these steps. The error message can be safely ignored. It means only that the domain is not pushing that configuration through a Group Policy object.

1. Remote Desktop Protocol (RDP) is enabled:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server' -name "fDenyTSConnections" -Value 0 -Type DWord

Set-ItemProperty -Path 'HKLM:\SOFTWARE\ Policies\Microsoft\Windows NT\Terminal Services' -name "fDenyTSConnections" -Value 0 -Type DWord
```

2. The RDP port is set up correctly (Default port 3389):

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp' -name "PortNumber" 3389 -Type DWord
```

When you deploy a VM, the default rules are created against port 3389. If you want to change the port number, do that after the VM is deployed in Azure.

3. The listener is listening in every network interface:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp' -name "LanAdapter" 0 -Type DWord
```

4. Configure the Network Level Authentication mode for the RDP connections:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp' -name "UserAuthentication" 1 -Type DWord

Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp' -name "SecurityLayer" 1 -Type DWord

Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp' -name "fAllowSecProtocolNegotiation" 1 -Type DWord
```

5. Set the keep-alive value:

```
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services' -name "KeepAliveEnable" 1 -Type DWord  
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services' -name "KeepAliveInterval" 1 -Type DWord  
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp' -name "KeepAliveTimeout" 1 -Type DWord
```

6. Reconnect:

```
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services' -name "fDisableAutoReconnect" 0 -Type DWord  
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp' -name "fInheritReconnectSame" 1 -Type DWord  
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp' -name "fReconnectSame" 0 -Type DWord
```

7. Limit the number of concurrent connections:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp' -name "MaxInstanceCount" 4294967295 -Type DWord
```

8. If there are any self-signed certificates tied to the RDP listener, remove them:

```
Remove-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp' -name "SSLCertificateSHA1Hash"
```

This is to make sure that you can connect at the beginning when you deploy the VM. You can also review this on a later stage after the VM is deployed in Azure if needed.

9. If the VM will be part of a Domain, check all the following settings to make sure that the former settings are not reverted. The policies that must be checked are the following:

- RDP is enabled:

Computer Configuration\Policies\Windows Settings\Administrative Templates\Components\Remote Desktop Services\Remote Desktop Session Host\Connections:

Allow users to connect remotely by using Remote Desktop

- NLA group policy:

Settings\Administrative Templates\Components\Remote Desktop Services\Remote Desktop Session Host\Security:

Require user Authentication for remote connections by using Network Level Authentication

- Keep Alive settings:

Computer Configuration\Policies\Windows Settings\Administrative Templates\Windows Components\Remote Desktop Services\Remote Desktop Session Host\Connections:

Configure keep-alive connection interval

- Reconnect settings:

Computer Configuration\Policies\Windows Settings\Administrative Templates\Windows Components\Remote Desktop Services\Remote Desktop Session Host\Connections:

Automatic reconnection

- Limit the number of connections settings:

Computer Configuration\Policies\Windows Settings\Administrative Templates\Windows Components\Remote Desktop Services\Remote Desktop Session Host\Connections:

Limit number of connections

Configure Windows Firewall rules

1. Turn on Windows Firewall on the three profiles (Domain, Standard, and Public):

```
Set-ItemProperty -Path
'HKLM:\SYSTEM\CurrentControlSet\services\SharedAccess\Parameters\FirewallPolicy\DomainProfile' -name
"EnableFirewall" -Value 1 -Type DWord
Set-ItemProperty -Path
'HKLM:\SYSTEM\CurrentControlSet\services\SharedAccess\Parameters\FirewallPolicy\PublicProfile' -name
"EnableFirewall" -Value 1 -Type DWord
Set-ItemProperty -Path
'HKLM:\SYSTEM\CurrentControlSet\services\SharedAccess\Parameters\FirewallPolicy\Standardprofile' -name
"EnableFirewall" -Value 1 -Type DWord
```

2. Run the following command in PowerShell to allow WinRM through the three firewall profiles (Domain, Private, and Public) and enable the PowerShell Remote service:

```
Enable-PSRemoting -force
netsh advfirewall firewall set rule dir=in name="Windows Remote Management (HTTP-In)" new enable=yes
netsh advfirewall firewall set rule dir=in name="Windows Remote Management (HTTP-In)" new enable=yes
```

3. Enable the following firewall rules to allow the RDP traffic

```
netsh advfirewall firewall set rule group="Remote Desktop" new enable=yes
```

4. Enable the File and Printer Sharing rule so that the VM can respond to a ping command inside the Virtual Network:

```
netsh advfirewall firewall set rule dir=in name="File and Printer Sharing (Echo Request - ICMPv4-In)" new enable=yes
```

5. If the VM will be part of a Domain, check the following settings to make sure that the former settings are not reverted. The AD policies that must be checked are the following:

- Enable the Windows Firewall profiles

Computer Configuration\Policies\Windows Settings\Administrative Templates\Network\Network Connection\Windows Firewall\Domain Profile\Windows Firewall: **Protect all network connections**

Computer Configuration\Policies\Windows Settings\Administrative Templates\Network\Network Connection\Windows Firewall\Standard Profile\Windows Firewall: **Protect all network connections**

- Enable RDP

Computer Configuration\Policies\Windows Settings\Administrative Templates\Network\Network Connection\Windows Firewall\Domain Profile\Windows Firewall: **Allow inbound Remote Desktop exceptions**

Computer Configuration\Policies\Windows Settings\Administrative Templates\Network\Network Connection\Windows Firewall\Standard Profile\Windows Firewall: **Allow inbound Remote Desktop exceptions**

- Enable ICMP-V4

Computer Configuration\Policies\Windows Settings\Administrative Templates\Network\Network Connection\Windows Firewall\Domain Profile\Windows Firewall: **Allow ICMP exceptions**

Computer Configuration\Policies\Windows Settings\Administrative Templates\Network\Network Connection\Windows Firewall\Standard Profile\Windows Firewall: **Allow ICMP exceptions**

Verify VM is healthy, secure, and accessible with RDP

1. To make sure the disk is healthy and consistent, run a check disk operation at the next VM restart:

```
Chkdsk /f
```

Make sure that the report shows a clean and healthy disk.

2. Set the Boot Configuration Data (BCD) settings.

NOTE

Make sure you run these commands on an elevated CMD window and **NOT** on PowerShell:

```
bcdedit /set {bootmgr} integrityservices enable  
bcdedit /set {default} device partition=C:  
bcdedit /set {default} integrityservices enable  
bcdedit /set {default} recoveryenabled Off  
bcdedit /set {default} osdevice partition=C:  
bcdedit /set {default} bootstatuspolicy IgnoreAllFailures
```

3. Verify that the Windows Management Instrumentations repository is consistent. To perform this, run the following command:

```
winmgmt /verifyrepository
```

If the repository is corrupted, see [WMI: Repository Corruption, or Not.](#)

4. Make sure that any other application is not using the port 3389. This port is used for the RDP service in Azure. You can run **netstat -anob** to see which ports are in used on the VM:

```
netstat -anob
```

5. If the Windows VHD that you want to upload is a domain controller, then follow these steps:

- A. Follow [these extra steps](#) to prepare the disk.
- B. Make sure that you know the DSRM password in case you have to start the VM in DSRM at some point. You may want to refer to this link to set the [DSRM password](#).
6. Make sure that the Built-in Administrator account and password are known to you. You may want to reset the current local administrator password and make sure that you can use this account to sign in to Windows through the RDP connection. This access permission is controlled by the "Allow log on through Remote Desktop Services" Group Policy object. You can view this object in the Local Group Policy Editor under:
- Computer Configuration\Windows Settings\Security Settings\Local Policies\User Rights Assignment
7. Check the following AD polices to make sure that you are not blocking your RDP access through RDP nor from the network:
- Computer Configuration\Windows Settings\Security Settings\Local Policies\User Rights Assignment\Deny access to this computer from the network
 - Computer Configuration\Windows Settings\Security Settings\Local Policies\User Rights Assignment\Deny log on through Remote Desktop Services
8. Restart the VM to make sure that Windows is still healthy can be reached by using the RDP connection. At this point, you may want to create a VM in your local Hyper-V to make sure the VM is starting completely and then test whether it is RDP reachable.
9. Remove any extra Transport Driver Interface filters, such as software that analyzes TCP packets or extra firewalls. You can also review this on a later stage after the VM is deployed in Azure if needed.
10. Uninstall any other third-party software and driver that is related to physical components or any other virtualization technology.

Install Windows Updates

The ideal configuration is to **have the patch level of the machine at the latest**. If this is not possible, make sure that the following updates are installed:

			MINIMUM FILE VERSION X64			
Component	Binary	Windows 7 & Windows Server 2008 R2	Windows 8 & Windows Server 2012	Windows 8.1 & Windows Server 2012 R2	Windows 10 & Windows Server 2016 RS1	Windows 10 RS2
Storage	disk.sys	6.1.7601.234 03 - KB3125574	6.2.9200.176 38 / 6.2.9200.217 57 - KB3137061	6.3.9600.182 03 - KB3137061	-	-
	storport.sys	6.1.7601.234 03 - KB3125574	6.2.9200.171 88 / 6.2.9200.213 06 - KB3018489	6.3.9600.185 73 - KB4022726	10.0.14393.1 358 - KB4022715	10.0.15063.3 32

			MINIMUM FILE VERSION X64			
	ntfs.sys	6.1.7601.234 03 - KB3125574	6.2.9200.176 23 / 6.2.9200.217 43 - KB3121255	6.3.9600.186 54 - KB4022726	10.0.14393.1 198 - KB4022715	10.0.15063.4 47
	lologmsg.dll	6.1.7601.234 03 - KB3125574	6.2.9200.163 84 - KB2995387	-	-	-
	Classpnp.sys	6.1.7601.234 03 - KB3125574	6.2.9200.170 61 / 6.2.9200.211 80 - KB2995387	6.3.9600.183 34 - KB3172614	10.0.14393.9 53 - KB4022715	-
	Volsnap.sys	6.1.7601.234 03 - KB3125574	6.2.9200.170 47 / 6.2.9200.211 65 - KB2975331	6.3.9600.182 65 - KB3145384	-	10.0.15063.0
	partmgr.sys	6.1.7601.234 03 - KB3125574	6.2.9200.166 81 - KB2877114	6.3.9600.174 01 - KB3000850	10.0.14393.9 53 - KB4022715	10.0.15063.0
	volmgr.sys					10.0.15063.0
	Volmgrx.sys	6.1.7601.234 03 - KB3125574	-	-	-	10.0.15063.0
	Msiscsi.sys	6.1.7601.234 03 - KB3125574	6.2.9200.210 06 - KB2955163	6.3.9600.186 24 - KB4022726	10.0.14393.1 066 - KB4022715	10.0.15063.4 47
	Msdsm.sys	6.1.7601.234 03 - KB3125574	6.2.9200.214 74 - KB3046101	6.3.9600.185 92 - KB4022726	-	-
	Mpio.sys	6.1.7601.234 03 - KB3125574	6.2.9200.211 90 - KB3046101	6.3.9600.186 16 - KB4022726	10.0.14393.1 198 - KB4022715	-
	Fveapi.dll	6.1.7601.233 11 - KB3125574	6.2.9200.209 30 - KB2930244	6.3.9600.182 94 - KB3172614	10.0.14393.5 76 - KB4022715	-
	Fveapibase.dll	6.1.7601.234 03 - KB3125574	6.2.9200.209 30 - KB2930244	6.3.9600.174 15 - KB3172614	10.0.14393.2 06 - KB4022715	-
Network	netvsc.sys	-	-	-	10.0.14393.1 198 - KB4022715	10.0.15063.2 50 - KB4020001

			MINIMUM FILE VERSION X64			
	mrxsmb10.sys	6.1.7601.238 16 - KB4022722	6.2.9200.221 08 - KB4022724	6.3.9600.186 03 - KB4022726	10.0.14393.4 79 - KB4022715	10.0.15063.4 83
	mrxsmb20.sys	6.1.7601.238 16 - KB4022722	6.2.9200.215 48 - KB4022724	6.3.9600.185 86 - KB4022726	10.0.14393.9 53 - KB4022715	10.0.15063.4 83
	mrxsmb.sys	6.1.7601.238 16 - KB4022722	6.2.9200.220 74 - KB4022724	6.3.9600.185 86 - KB4022726	10.0.14393.9 53 - KB4022715	10.0.15063.0
	tcpip.sys	6.1.7601.237 61 - KB4022722	6.2.9200.220 70 - KB4022724	6.3.9600.184 78 - KB4022726	10.0.14393.1 358 - KB4022715	10.0.15063.4 47
	http.sys	6.1.7601.234 03 - KB3125574	6.2.9200.172 85 - KB3042553	6.3.9600.185 74 - KB4022726	10.0.14393.2 51 - KB4022715	10.0.15063.4 83
	vmswitch.sys	6.1.7601.237 27 - KB4022719	6.2.9200.221 17 - KB4022724	6.3.9600.186 54 - KB4022726	10.0.14393.1 358 - KB4022715	10.0.15063.1 38
Core	ntoskrnl.exe	6.1.7601.238 07 - KB4022719	6.2.9200.221 70 - KB4022718	6.3.9600.186 96 - KB4022726	10.0.14393.1 358 - KB4022715	10.0.15063.4 83
Remote Desktop Services	rdpcorets.dll	6.2.9200.215 06 - KB4022719	6.2.9200.221 04 - KB4022724	6.3.9600.186 19 - KB4022726	10.0.14393.1 198 - KB4022715	10.0.15063.0
	termsrv.dll	6.1.7601.234 03 - KB3125574	6.2.9200.170 48 - KB2973501	6.3.9600.174 15 - KB3000850	10.0.14393.0 - KB4022715	10.0.15063.0
	termdd.sys	6.1.7601.234 03 - KB3125574	-	-	-	-
	win32k.sys	6.1.7601.238 07 - KB4022719	6.2.9200.221 68 - KB4022718	6.3.9600.186 98 - KB4022726	10.0.14393.5 94 - KB4022715	-
	rdpdd.dll	6.1.7601.234 03 - KB3125574	-	-	-	-
	rdpwd.sys	6.1.7601.234 03 - KB3125574	-	-	-	-
Security	Due to WannaCrypt	KB4012212	KB4012213	KB4012213	KB4012606	KB4012606

			MINIMUM FILE VERSION X64			
			KB4012216		KB4013198	KB4013198
		KB4012215	KB4012214	KB4012216	KB4013429	KB4013429
			KB4012217		KB4013429	KB4013429

When to use sysprep

Sysprep is a process that you could run into a windows installation that will reset the installation of the system and will provide an "out of the box experience" by removing all personal data and resetting several components. You typically do this if you want to create a template from which you can deploy several other VMs that have a specific configuration. This is called a **generalized image**.

If, instead, you want only to create one VM from one disk, you don't have to use sysprep. In this situation, you can just create the VM from what is known as a **specialized image**.

For more information about how to create a VM from a specialized disk, see:

- [Create a VM from a specialized disk](#)
- [Create a VM from a specialized VHD disk](#)

If you want to create a generalized image, you need to run sysprep. For more information about Sysprep, see [How to Use Sysprep: An Introduction](#).

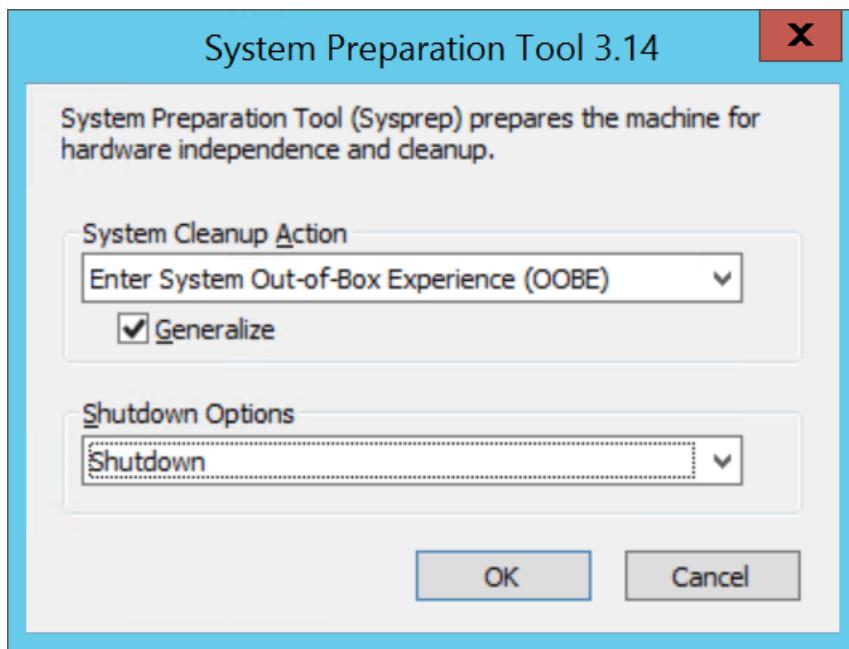
Not every role or application that's installed on a Windows-based computer supports this generalization. So before you run this procedure, refer to the following article to make sure that the role of that computer is supported by sysprep. For more information, [Sysprep Support for Server Roles](#).

Steps to generalize a VHD

NOTE

After you run sysprep.exe as specified in the following steps, turn off the VM, and do not turn it back on until you create an image from it in Azure.

1. Sign in to the Windows VM.
2. Run **Command Prompt** as an administrator.
3. Change the directory to: **%windir%\system32\sysprep**, and then run **sysprep.exe**.
4. In the **System Preparation Tool** dialog box, select **Enter System Out-of-Box Experience (OOBE)**, and make sure that the **Generalize** check box is selected.



5. In **Shutdown Options**, select **Shutdown**.
6. Click **OK**.
7. When Sysprep completes, shut down the VM. Do not use **Restart** to shut down the VM.
8. Now the VHD is ready to be uploaded. For more information about how to create a VM from a generalized disk, see [Upload a generalized VHD and use it to create a new VMs in Azure](#).

Complete recommended configurations

The following settings do not affect VHD uploading. However, we strongly recommend that you configured them.

- Install the [Azure VMs Agent](#). Then you can enable VM extensions. The VM extensions implement most of the critical functionality that you might want to use with your VMs such as resetting passwords, configuring RDP, and so on. For more information, see:
 - [VM Agent and Extensions – Part 1](#)
 - [VM Agent and Extensions – Part 2](#)
- The Dump log can be helpful in troubleshooting Windows crash issues. Enable the Dump log collection:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\CrashControl' -name "CrashDumpEnable" -Value "2" -Type DWord  
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\CrashControl' -name "DumpFile" -Value "%SystemRoot%\MEMORY.DMP"  
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\CrashControl' -name "AutoReboot" -Value 0 -Type DWord  
New-Item -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps'  
New-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps' -name "DumpFolder" -Value "c:\CrashDumps"  
New-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps' -name "DumpCount" -Value 10 -Type DWord  
New-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps' -name "DumpType" -Value 2 -Type DWord  
Set-Service -Name WerSvc -StartupType Manual
```

If you receive any errors during any of the procedural steps in this article, this means that the registry keys already exists. In this situation, use the following commands instead:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\CrashControl' -name "CrashDumpEnable" -Value "2" -Type DWord  
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\CrashControl' -name "DumpFile" -Value "%SystemRoot%\MEMORY.DMP"  
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps' -name "DumpFolder" -Value "c:\CrashDumps"  
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps' -name "DumpCount" -Value 10 -Type DWord  
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps' -name "DumpType" -Value 2 -Type DWord  
Set-Service -Name WerSvc -StartupType Manual
```

- After the VM is created in Azure, we recommend that you put the pagefile on the "Temporal drive" volume to improve performance. You can set up this as follows:

```
Set-ItemProperty -Path 'HKLM:\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management' -name "PagingFiles" -Value "D:\pagefile"
```

If there's any data disk that is attached to the VM, the Temporal drive volume's drive letter is typically "D." This designation could be different, depending on the number of available drives and the settings that you make.

Next steps

- [Upload a Windows VM image to Azure for Resource Manager deployments](#)
- [Troubleshoot Azure Windows virtual machine activation problems](#)

Create a managed image of a generalized VM in Azure

11/27/2017 • 4 min to read • [Edit Online](#)

A managed image resource can be created from a generalized VM that is stored as either a managed disk or an unmanaged disk in a storage account. The image can then be used to create multiple VMs.

Generalize the Windows VM using Sysprep

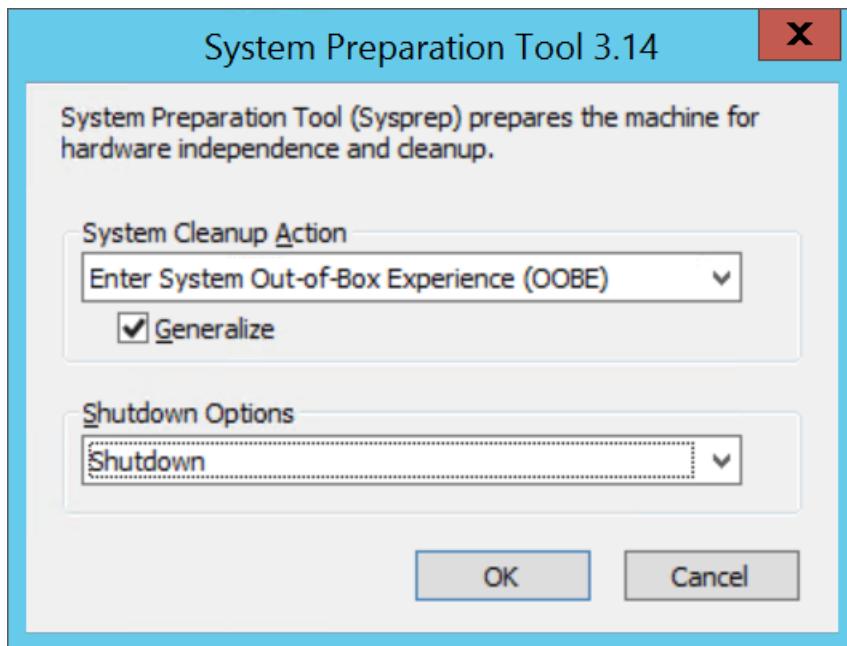
Sysprep removes all your personal account information, among other things, and prepares the machine to be used as an image. For details about Sysprep, see [How to Use Sysprep: An Introduction](#).

Make sure the server roles running on the machine are supported by Sysprep. For more information, see [Sysprep Support for Server Roles](#)

IMPORTANT

If you are running Sysprep before uploading your VHD to Azure for the first time, make sure you have [prepared your VM](#) before running Sysprep.

1. Sign in to the Windows virtual machine.
2. Open the Command Prompt window as an administrator. Change the directory to `%windir%\system32\sysprep`, and then run `sysprep.exe`.
3. In the **System Preparation Tool** dialog box, select **Enter System Out-of-Box Experience (OOBE)**, and make sure that the **Generalize** check box is selected.
4. In **Shutdown Options**, select **Shutdown**.
5. Click **OK**.



6. When Sysprep completes, it shuts down the virtual machine. Do not restart the VM.

Create a managed image in the portal

1. Open the [portal](#).
2. In the menu on the left, click Virtual Machines and then select the VM from the list.
3. In the page for the VM, on the upper menu, click **Capture**.
4. In **Name**, type the name that you would like to use for the image.
5. In **Resource group** either select **Create new** and type in a name, or select **Use existing** and select a resource group to use from the drop-down list.
6. If you want to delete the source VM after the image has been created, select **Automatically delete this virtual machine after creating the image**.
7. When you are done, click **Create**.
8. After the image is created, you will see it as an **Image** resource in the list of resources in the resource group.

Create an image of a VM using Powershell

Creating an image directly from the VM ensures that the image includes all of the disks associated with the VM, including the OS Disk and any data disks. This example shows how to create a managed image from a VM that uses managed disks.

Before you begin, make sure that you have the latest version of the AzureRM.Compute PowerShell module. Run the following command to install it.

```
Install-Module AzureRM.Compute -RequiredVersion 2.6.0
```

For more information, see [Azure PowerShell Versioning](#).

1. Create some variables.

```
$vmName = "myVM"  
$rgName = "myResourceGroup"  
$location = "EastUS"  
$imageName = "myImage"
```

2. Make sure the VM has been deallocated.

```
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName -Force
```

3. Set the status of the virtual machine to **Generalized**.

```
Set-AzureRmVm -ResourceGroupName $rgName -Name $vmName -Generalized
```

4. Get the virtual machine.

```
$vm = Get-AzureRmVM -Name $vmName -ResourceGroupName $rgName
```

5. Create the image configuration.

```
$image = New-AzureRmImageConfig -Location $location -SourceVirtualMachineId $vm.ID
```

6. Create the image.

```
New-AzureRmImage -Image $image -ImageName $imageName -ResourceGroupName $rgName
```

Create an image from a managed disk using PowerShell

If you only want to create an image of the OS disk, you can also create an image by specifying the managed disk ID as the OS disk.

1. Create some variables.

```
$vmName = "myVM"  
$rgName = "myResourceGroup"  
$location = "EastUS"  
$snapshotName = "mySnapshot"  
$imageName = "myImage"
```

2. Get the VM.

```
$vm = Get-AzureRmVm -Name myVM -ResourceGroupName $rgName
```

3. Get the ID of the managed disk.

```
$diskID = $vm.StorageProfile.OsDisk.ManagedDisk.Id
```

4. Create the image configuration.

```
$imageConfig = New-AzureRmImageConfig -Location $location  
$imageConfig = Set-AzureRmImageOsDisk -Image $imageConfig -OsState Generalized -OsType Windows -  
ManagedDiskId $diskID
```

5. Create the image.

```
New-AzureRmImage -ImageName $imageName -ResourceGroupName $rgName -Image $imageConfig
```

Create an image from a snapshot using Powershell

You can create a managed image from a snapshot of a generalized VM.

1. Create some variables.

```
$rgName = "myResourceGroup"  
$location = "EastUS"  
$snapshotName = "mySnapshot"  
$imageName = "myImage"
```

2. Get the snapshot.

```
$snapshot = Get-AzureRmSnapshot -ResourceGroupName $rgName -SnapshotName $snapshotName
```

3. Create the image configuration.

```
$imageConfig = New-AzureRmImageConfig -Location $location  
$imageConfig = Set-AzureRmImageOsDisk -Image $imageConfig -OsState Generalized -OsType Windows -  
SnapshotId $snapshot.Id
```

4. Create the image.

```
New-AzureRmImage -ImageName $imageName -ResourceGroupName $rgName -Image $imageConfig
```

Create image from a VHD in a storage account

Create a managed image from a generalized OS VHD in a storage account. You need the URI of the VHD in the storage account, which is in the format

https://mystorageaccount.blob.core.windows.net/container/vhd_filename.vhd. In this example, the VHD that we are using is in *mystorageaccount* in a container named *vhdcontainer* and the VHD filename is *osdisk.vhd*.

1. First, set the common parameters:

```
$vmName = "myVM"  
$rgName = "myResourceGroup"  
$location = "EastUS"  
$imageName = "myImage"  
$osVhdUri = "https://mystorageaccount.blob.core.windows.net/vhdcontainer/osdisk.vhd"
```

2. Step\deallocate the VM.

```
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName -Force
```

3. Mark the VM as generalized.

```
Set-AzureRmVm -ResourceGroupName $rgName -Name $vmName -Generalized
```

4. Create the image using your generalized OS VHD.

```
$imageConfig = New-AzureRmImageConfig -Location $location  
$imageConfig = Set-AzureRmImageOsDisk -Image $imageConfig -OsType Windows -OsState Generalized -BlobUri  
$osVhdUri  
$image = New-AzureRmImage -ImageName $imageName -ResourceGroupName $rgName -Image $imageConfig
```

Next steps

- Now you can [create a VM from the generalized managed image](#).

Create a VM from a managed image

12/6/2017 • 4 min to read • [Edit Online](#)

You can create multiple VMs from a managed VM image using PowerShell or the Azure portal. A managed VM image contains the information necessary to create a VM, including the OS and data disks. The VHDs that make up the image, including both the OS disks and any data disks, are stored as managed disks.

You need to have already [created a managed VM image](#) to use for creating the new VM.

Use the portal

1. Open the [Azure portal](#).
2. In the left menu, select **All resources**. You can sort the resources by **Type** to easily find your images.
3. Select the image you want to use from the list. The image **Overview** page opens.
4. Click **+ Create VM** from the menu.
5. Enter the virtual machine information. The user name and password entered here is used to log in to the virtual machine. When complete, click **OK**. You can create the new VM in an existing Resource Group, or choose **Create new** to create a new resource group to store the VM.
6. Select a size for the VM. To see more sizes, select **View all** or change the **Supported disk type** filter.
7. Under **Settings**, make changes as necessary and click **OK**.
8. On the summary page, you should see your image name listed for **Private image**. Click **Ok** to start the virtual machine deployment.

Use PowerShell

Prerequisites

Make sure that you have the latest versions of the AzureRM.Compute and AzureRM.Network PowerShell modules. Open a PowerShell prompt as an Administrator and run the following command to install them.

```
Install-Module AzureRM.Compute,AzureRM.Network
```

For more information, see [Azure PowerShell Versioning](#).

Collect information about the image

First we need to gather basic information about the image and create a variable for the image. This example uses a managed VM image named **myImage** that is in the **myResourceGroup** resource group in the **West Central US** location.

```
$rgName = "myResourceGroup"
.setLocation = "West Central US"
$imageName = "myImage"
$image = Get-AzureRMImage -ImageName $imageName -ResourceGroupName $rgName
```

Create a virtual network

Create the vNet and subnet of the [virtual network](#).

Create the subnet. This example creates a subnet named **mySubnet** with the address prefix of **10.0.0.0/24**.

```
$subnetName = "mySubnet"
$singleSubnet = New-AzureRmVirtualNetworkSubnetConfig ` 
    -Name $subnetName -AddressPrefix 10.0.0.0/24
```

Create the virtual network. This example creates a virtual network named **myVnet** with the address prefix of **10.0.0.0/16**.

```
$vnetName = "myVnet"
$vnet = New-AzureRmVirtualNetwork ` 
    -Name $vnetName ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -AddressPrefix 10.0.0.0/16 ` 
    -Subnet $singleSubnet
```

Create a public IP address and network interface

To enable communication with the virtual machine in the virtual network, you need a [public IP address](#) and a network interface.

Create a public IP address. This example creates a public IP address named **myPip**.

```
$ipName = "myPip"
$pip = New-AzureRmPublicIpAddress ` 
    -Name $ipName ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -AllocationMethod Dynamic
```

Create the NIC. This example creates a NIC named **myNic**.

```
$nicName = "myNic"
$nic = New-AzureRmNetworkInterface ` 
    -Name $nicName ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -SubnetId $vnet.Subnets[0].Id ` 
    -PublicIpAddressId $pip.Id
```

Create the network security group and an RDP rule

To be able to log in to your VM using RDP, you need to have a network security rule (NSG) that allows RDP access on port 3389.

This example creates an NSG named **myNsg** that contains a rule called **myRdpRule** that allows RDP traffic over port 3389. For more information about NSGs, see [Opening ports to a VM in Azure using PowerShell](#).

```
$nsgName = "myNsg"
$ruleName = "myRdpRule"
$rdpRule = New-AzureRmNetworkSecurityRuleConfig -Name $ruleName -Description "Allow RDP" ` 
    -Access Allow -Protocol Tcp -Direction Inbound -Priority 110 ` 
    -SourceAddressPrefix Internet -SourcePortRange * ` 
    -DestinationAddressPrefix * -DestinationPortRange 3389

$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $rgName -Location $location ` 
    -Name $nsgName -SecurityRules $rdpRule
```

Create a variable for the virtual network

Create a variable for the completed virtual network.

```
$vnet = Get-AzureRmVirtualNetwork -ResourceGroupName $rgName -Name $vnetName
```

Get the credentials for the VM

The following cmdlet will open a window where you will enter a new user name and password to use as the local administrator account for remotely accessing the VM.

```
$cred = Get-Credential
```

Set variables for the VM name, computer name and the size of the VM

Create variables for the VM name and computer name. This example sets the VM name as **myVM** and the computer name as **myComputer**.

```
$vmName = "myVM"  
$computerName = "myComputer"
```

Set the size of the virtual machine. This example creates **Standard_DS1_v2** sized VM. See the [VM sizes](#) documentation for more information.

```
$vmSize = "Standard_DS1_v2"
```

Add the VM name and size to the VM configuration.

```
$vm = New-AzureRmVMConfig -VMName $vmName -VMSize $vmSize
```

Set the VM image as source image for the new VM

Set the source image using the ID of the managed VM image.

```
$vm = Set-AzureRmVMSourceImage -VM $vm -Id $image.Id
```

Set the OS configuration and add the NIC.

Enter the storage type (PremiumLRS or StandardLRS) and the size of the OS disk. This example sets the account type to **PremiumLRS**, the disk size to **128 GB** and disk caching to **ReadWrite**.

```
$vm = Set-AzureRmVMOSDisk -VM $vm `  
-StorageAccountType PremiumLRS `  
-DiskSizeInGB 128 `  
-CreateOption FromImage `  
-Caching ReadWrite  
  
$vm = Set-AzureRmVMOperatingSystem -VM $vm -Windows -ComputerName $computerName `  
-Credential $cred -ProvisionVMAgent -EnableAutoUpdate  
  
$vm = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id
```

Create the VM

Create the new Vm using the configuration that we have built and stored in the **\$vm** variable.

```
New-AzureRmVM -VM $vm -ResourceGroupName $rgName -Location $location
```

Verify that the VM was created

When complete, you should see the newly created VM in the [Azure portal](#) under **Browse > Virtual machines**, or by using the following PowerShell commands:

```
$vmList = Get-AzureRmVM -ResourceGroupName $rgName  
$vmList.Name
```

Next steps

To manage your new virtual machine with Azure PowerShell, see [Create and manage Windows VMs with the Azure PowerShell module](#).

How to use Packer to create Windows virtual machine images in Azure

12/19/2017 • 6 min to read • [Edit Online](#)

Each virtual machine (VM) in Azure is created from an image that defines the Windows distribution and OS version. Images can include pre-installed applications and configurations. The Azure Marketplace provides many first and third-party images for most common OS' and application environments, or you can create your own custom images tailored to your needs. This article details how to use the open source tool [Packer](#) to define and build custom images in Azure.

Create Azure resource group

During the build process, Packer creates temporary Azure resources as it builds the source VM. To capture that source VM for use as an image, you must define a resource group. The output from the Packer build process is stored in this resource group.

Create a resource group with [New-AzureRmResourceGroup](#). The following example creates a resource group named *myResourceGroup* in the *eastus* location:

```
$rgName = "myResourceGroup"  
$location = "East US"  
New-AzureRmResourceGroup -Name $rgName -Location $location
```

Create Azure credentials

Packer authenticates with Azure using a service principal. An Azure service principal is a security identity that you can use with apps, services, and automation tools like Packer. You control and define the permissions as to what operations the service principal can perform in Azure.

Create a service principal with [New-AzureRmADServicePrincipal](#) and assign permissions for the service principal to create and manage resources with [New-AzureRmRoleAssignment](#):

```
$sp = New-AzureRmADServicePrincipal -DisplayName "Azure Packer" `  
-Password (ConvertTo-SecureString "P@ssw0rd!" -AsPlainText -Force)  
Sleep 20  
New-AzureRmRoleAssignment -RoleDefinitionName Contributor -ServicePrincipalName $sp.ApplicationId
```

To authenticate to Azure, you also need to obtain your Azure tenant and subscription IDs with [Get-AzureRmSubscription](#):

```
$sub = Get-AzureRmSubscription  
$sub.TenantId  
$sub.SubscriptionId
```

You use these two IDs in the next step.

Define Packer template

To build images, you create a template as a JSON file. In the template, you define builders and provisioners that

carry out the actual build process. Packer has a [provisioner for Azure](#) that allows you to define Azure resources, such as the service principal credentials created in the preceding step.

Create a file named `windows.json` and paste the following content. Enter your own values for the following:

PARAMETER	WHERE TO OBTAIN
<code>client_id</code>	View service principal ID with <code>\$sp.applicationId</code>
<code>client_secret</code>	Password you specified in <code>\$securePassword</code>
<code>tenant_id</code>	Output from <code>\$sub.TenantId</code> command
<code>subscription_id</code>	Output from <code>\$sub.SubscriptionId</code> command
<code>object_id</code>	View service principal object ID with <code>\$sp.Id</code>
<code>managed_image_resource_group_name</code>	Name of resource group you created in the first step
<code>managed_image_name</code>	Name for the managed disk image that is created

```
{
  "builders": [
    {
      "type": "azure-arm",
      "client_id": "0831b578-8ab6-40b9-a581-9a880a94aab1",
      "client_secret": "P@ssw0rd!",
      "tenant_id": "72f988bf-86f1-41af-91ab-2d7cd011db47",
      "subscription_id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx",
      "object_id": "a7dfb070-0d5b-47ac-b9a5-cf214fff0ae2",

      "managed_image_resource_group_name": "myResourceGroup",
      "managed_image_name": "myPackerImage",

      "os_type": "Windows",
      "image_publisher": "MicrosoftWindowsServer",
      "image_offer": "WindowsServer",
      "image_sku": "2016-Datacenter",

      "communicator": "winrm",
      "winrm_use_ssl": "true",
      "winrm_insecure": "true",
      "winrm_timeout": "3m",
      "winrm_username": "packer",

      "azure_tags": {
        "dept": "Engineering",
        "task": "Image deployment"
      },
      "location": "East US",
      "vm_size": "Standard_DS2_v2"
    }],
  "provisioners": [
    {
      "type": "powershell",
      "inline": [
        "Add-WindowsFeature Web-Server",
        "if( Test-Path $Env:SystemRoot\\windows\\system32\\Sysprep\\unattend.xml ){ rm $Env:SystemRoot\\windows\\system32\\Sysprep\\unattend.xml -Force}",
        "& $Env:SystemRoot\\System32\\Sysprep\\Sysprep.exe /oobe /generalize /shutdown /quiet"
      ]
    }
  ]
}
```

This template builds a Windows Server 2016 VM, installs IIS, then generalizes the VM with Sysprep.

Build Packer image

If you don't already have Packer installed on your local machine, [follow the Packer installation instructions](#).

Build the image by specifying your Packer template file as follows:

```
./packer build windows.json
```

An example of the output from the preceding commands is as follows:

```
azure-arm output will be in this color.

==> azure-arm: Running builder ...
    azure-arm: Creating Azure Resource Manager (ARM) client ...
==> azure-arm: Creating resource group ...
==> azure-arm:  -> ResourceGroupName : 'packer-Resource-Group-pq0mthtbtt'
==> azure-arm:  -> Location          : 'East US'
==> azure-arm:  -> Tags            :
```

```

==> azure-arm: -> task : Image deployment
==> azure-arm: -> dept : Engineering
==> azure-arm: Validating deployment template ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> DeploymentName : 'pkrdppq0mthbt'
==> azure-arm: Deploying deployment template ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> DeploymentName : 'pkrdppq0mthbt'
==> azure-arm: Getting the certificate's URL ...
==> azure-arm: -> Key Vault Name : 'pkrkvpq0mthbt'
==> azure-arm: -> Key Vault Secret Name : 'packerKeyVaultSecret'
==> azure-arm: -> Certificate URL :
'https://pkrkvpq0mthbt.vault.azure.net/secrets/packerKeyVaultSecret/8c7bd823e4fa44e1abb747636128adbb'
==> azure-arm: Setting the certificate's URL ...
==> azure-arm: Validating deployment template ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> DeploymentName : 'pkrdppq0mthbt'
==> azure-arm: Deploying deployment template ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> DeploymentName : 'pkrdppq0mthbt'
==> azure-arm: Getting the VM's IP address ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> PublicIPAddressName : 'packerPublicIP'
==> azure-arm: -> NicName : 'packerNic'
==> azure-arm: -> Network Connection : 'PublicEndpoint'
==> azure-arm: -> IP Address : '40.76.55.35'
==> azure-arm: Waiting for WinRM to become available...
==> azure-arm: Connected to WinRM!
==> azure-arm: Provisioning with Powershell...
==> azure-arm: Provisioning with shell script: /var/folders/h1/ymh5bxdx15wgdn5hvgj1wc0zh0000gn/T/packer-
powershell-provisioner902510110
    azure-arm: #< CLIXML
    azure-arm:
    azure-arm: Success Restart Needed Exit Code      Feature Result
    azure-arm: ----- ----- ----- -----
    azure-arm: True   No        Success      {Common HTTP Features, Default Document, D...
    azure-arm: <Objs Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04"><Obj
S="progress" RefId="0"><TN RefId="0"><T>System.Management.Automation.PSCustomObject</T><T>System.Object</T>
</TN><MS><I64 N="SourceId">1</I64><PR N="Record"><AV>Preparing modules for first use.</AV><AI>0</AI><Nil />
<PI>-1</PI><PC>-1</PC><T>Completed</T><SR>-1</SR><SD> </SD></PR></MS></Obj></Objs>
==> azure-arm: Querying the machine's properties ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> ComputeName : 'pkrvmpq0mthbt'
==> azure-arm: -> Managed OS Disk : '/subscriptions/guid/resourceGroups/packer-Resource-Group-
pq0mthbt/providers/Microsoft.Compute/disks/osdisk'
==> azure-arm: Powering off machine ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> ComputeName : 'pkrvmpq0mthbt'
==> azure-arm: Capturing image ...
==> azure-arm: -> Compute ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: -> Compute Name : 'pkrvmpq0mthbt'
==> azure-arm: -> Compute Location : 'East US'
==> azure-arm: -> Image ResourceGroupName : 'myResourceGroup'
==> azure-arm: -> Image Name : 'myPackerImage'
==> azure-arm: -> Image Location : 'eastus'
==> azure-arm: Deleting resource group ...
==> azure-arm: -> ResourceGroupName : 'packer-Resource-Group-pq0mthbt'
==> azure-arm: Deleting the temporary OS disk ...
==> azure-arm: -> OS Disk : skipping, managed disk was used...
Build 'azure-arm' finished.

==> Builds finished. The artifacts of successful builds are:
--> azure-arm: Azure.ResourceManagement.VMImage:

ManagedImageResourceGroupName: myResourceGroup
ManagedImageName: myPackerImage
ManagedImageLocation: eastus

```

It takes a few minutes for Packer to build the VM, run the provisioners, and clean up the deployment.

Create VM from Azure Image

You can now create a VM from your Image with [New-AzureRmVM](#). First, set an administrator username and password for the VM with [Get-Credential](#).

```
$cred = Get-Credential
```

The following example creates a VM named *myVM* from *myPackerImage*.

```

# Create a subnet configuration
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig ` 
    -Name mySubnet ` 
    -AddressPrefix 192.168.1.0/24

# Create a virtual network
$vnet = New-AzureRmVirtualNetwork ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -Name myVnet ` 
    -AddressPrefix 192.168.0.0/16 ` 
    -Subnet $subnetConfig

# Create a public IP address and specify a DNS name
$publicIP = New-AzureRmPublicIpAddress ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -AllocationMethod "Static" ` 
    -IdleTimeoutInMinutes 4 ` 
    -Name "myPublicIP"

# Create an inbound network security group rule for port 80
$nsgRuleWeb = New-AzureRmNetworkSecurityRuleConfig ` 
    -Name myNetworkSecurityGroupRuleWeb ` 
    -Protocol Tcp ` 
    -Direction Inbound ` 
    -Priority 1001 ` 
    -SourceAddressPrefix * ` 
    -SourcePortRange * ` 
    -DestinationAddressPrefix * ` 
    -DestinationPortRange 80 ` 
    -Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -Name myNetworkSecurityGroup ` 
    -SecurityRules $nsgRuleWeb

# Create a virtual network card and associate with public IP address and NSG
$nic = New-AzureRmNetworkInterface ` 
    -Name myNic ` 
    -ResourceGroupName $rgName ` 
    -Location $location ` 
    -SubnetId $vnet.Subnets[0].Id ` 
    -PublicIpAddressId $publicIP.Id ` 
    -NetworkSecurityGroupId $nsg.Id

# Define the image created by Packer
$image = Get-AzureRMImage -ImageName myPackerImage -ResourceGroupName $rgName

# Create a virtual machine configuration
$vmConfig = New-AzureRmVMConfig -VMName myVM -VMSize Standard_DS2 | ` 
    Set-AzureRmVMOperatingSystem -Windows -ComputerName myVM -Credential $cred | ` 
    Set-AzureRmVMSourceImage -Id $image.Id | ` 
    Add-AzureRmVMNetworkInterface -Id $nic.Id

New-AzureRmVM -ResourceGroupName $rgName -Location $location -VM $vmConfig

```

It takes a few minutes to create the VM from your Packer image.

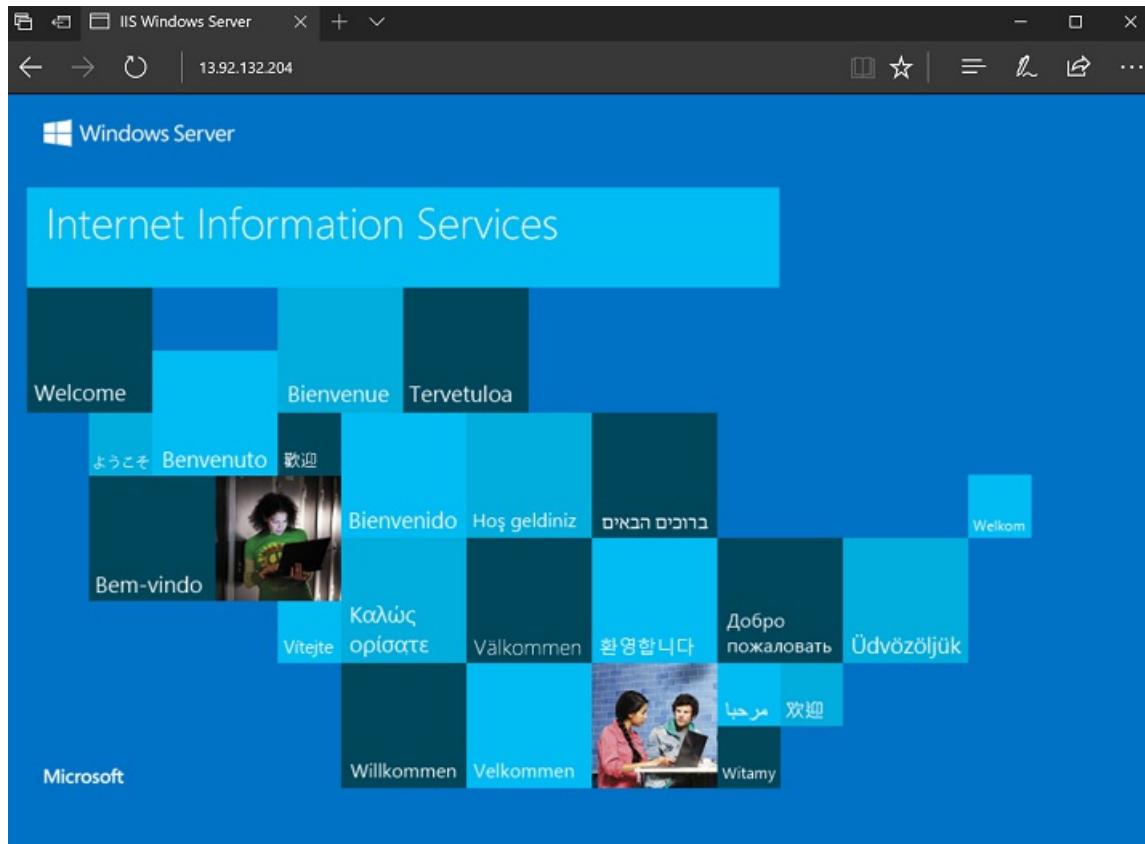
Test VM and IIS

Obtain the public IP address of your VM with [Get-AzureRmPublicIPAddress](#). The following example obtains the IP

address for *myPublicIP* created earlier:

```
Get-AzureRmPublicIPAddress `  
-ResourceGroupName $rgName `  
-Name "myPublicIP" | select "IpAddress"
```

You can then enter the public IP address in to a web browser.



Next steps

In this example, you used Packer to create a VM image with IIS already installed. You can use this VM image alongside existing deployment workflows, such as to deploy your app to VMs created from the Image with Team Services, Ansible, Chef, or Puppet.

For additional example Packer templates for other Windows distros, see [this GitHub repo](#).

Use Windows client in Azure for dev/test scenarios

12/15/2017 • 1 min to read • [Edit Online](#)

You can use Windows 7, Windows 8, or Windows 10 in Azure for dev/test scenarios provided you have an appropriate Visual Studio (formerly MSDN) subscription. This article outlines the eligibility requirements for running Windows client in Azure and use of the Azure Gallery images.

Subscription eligibility

Active Visual Studio subscribers (people who have acquired a Visual Studio subscription license) can use Windows client for development and testing purposes. Windows client can be used on your own hardware and Azure virtual machines running in any type of Azure subscription. Windows client may not be deployed to or used on Azure for normal production use, or used by people who are not active Visual Studio subscribers.

For your convenience, certain Windows 10 images are available from the Azure Gallery within [eligible dev/test offers](#). Visual Studio subscribers within any type of offer can also [adequately prepare and create](#) a 64-bit Windows 7, Windows 8, or Windows 10 image and then [upload to Azure](#). The use remains limited to dev/test by active Visual Studio subscribers.

Eligible offers

The following table details the offer IDs that are eligible to deploy Windows 10 through the Azure Gallery. The Windows 10 images are only visible to the following offers. Visual Studio subscribers who need to run Windows client in a different offer type require you to [adequately prepare and create](#) a 64-bit Windows 7, Windows 8, or Windows 10 image and [then upload to Azure](#).

Offer Name	Offer Number	Available Client Images
Pay-As-You-Go Dev/Test	0023P	Windows 10
Visual Studio Enterprise (MPN) subscribers	0029P	Windows 10
Visual Studio Professional subscribers	0059P	Windows 10
Visual Studio Test Professional subscribers	0060P	Windows 10
Visual Studio Premium with MSDN (benefit)	0061P	Windows 10
Visual Studio Enterprise subscribers	0063P	Windows 10
Visual Studio Enterprise (BizSpark) subscribers	0064P	Windows 10
Enterprise Dev/Test	0148P	Windows 10

Check your Azure subscription

If you do not know your offer ID, you can obtain it through the Azure portal in one of these two ways:

- On the *Subscriptions* window:

The screenshot shows the Azure Subscriptions window for a tenant named 'Contoso'. On the left, there's a search bar and a sidebar with filters for Role (All) and Status (All). Below that is a button labeled 'Apply' and a search input field. At the bottom of the sidebar are 'SUBS...' and 'SUBSCRIP...' dropdowns, followed by a list item 'Contoso... b031470d-04... ...'. The main area has tabs for 'Overview' (which is selected and highlighted in blue), 'Access control (IAM)', 'Diagnose and solv...', and 'BILLING'. Under 'BILLING', there's a 'Partner information' section. On the right, there's a 'Manage' button, a 'Transfer' button, and a trash bin icon. A yellow warning box says 'For billing information, v...'. Below that is a 'Essentials' section with a collapsed arrow. The 'Offer ID' row is highlighted with a red box. The table data is as follows:

Subscription ID	Subscript...
b031470d-04xx-xxxx...	Contoso
My Role	Current...
Owner	5/1/17 -
Offer	Currency
Microsoft Azure In...	USD
Offer ID	Status
MS-AZR-0044P	Active

- Or, click **Billing** and then click your subscription ID. The offer ID appears in the *Billing* window.

You can also view the offer ID from the '[Subscriptions](#)' tab of the Azure Account portal:

The screenshot shows the 'ACCOUNT ADMINISTRATOR' page. It displays various subscription details:

- SUBSCRIPTION ID:** Visual Studio Enterprise
- ORDER ID:** (not visible)
- OFFER:** Visual Studio Enterprise
- OFFER ID:** MS-AZR-0063P (this field is highlighted with a red box)
- CURRENCY:** USD
- STATUS:** Active

Next steps

You can now deploy your VMs using [PowerShell](#), [Resource Manager templates](#), or [Visual Studio](#).

Download a Windows VHD from Azure

8/21/2017 • 2 min to read • [Edit Online](#)

In this article, you learn how to download a [Windows virtual hard disk \(VHD\)](#) file from Azure using the Azure portal.

Virtual machines (VMs) in Azure use [disks](#) as a place to store an operating system, applications, and data. All Azure VMs have at least two disks – a Windows operating system disk and a temporary disk. The operating system disk is initially created from an image, and both the operating system disk and the image are VHDs stored in an Azure storage account. Virtual machines also can have one or more data disks, that are also stored as VHDs.

Stop the VM

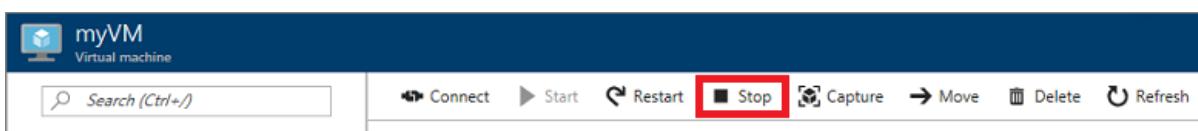
A VHD can't be downloaded from Azure if it's attached to a running VM. You need to stop the VM to download a VHD. If you want to use a VHD as an [image](#) to create other VMs with new disks, you use [Sysprep](#) to generalize the operating system contained in the file and then stop the VM. To use the VHD as a disk for a new instance of an existing VM or data disk, you only need to stop and deallocate the VM.

To use the VHD as an image to create other VMs, complete these steps:

1. If you haven't already done so, sign in to the [Azure portal](#).
2. [Connect to the VM](#).
3. On the VM, open the Command Prompt window as an administrator.
4. Change the directory to `%windir%\system32\sysprep` and run `sysprep.exe`.
5. In the System Preparation Tool dialog box, select **Enter System Out-of-Box Experience (OOBE)**, and make sure that **Generalize** is selected.
6. In Shutdown Options, select **Shutdown**, and then click **OK**.

To use the VHD as a disk for a new instance of an existing VM or data disk, complete these steps:

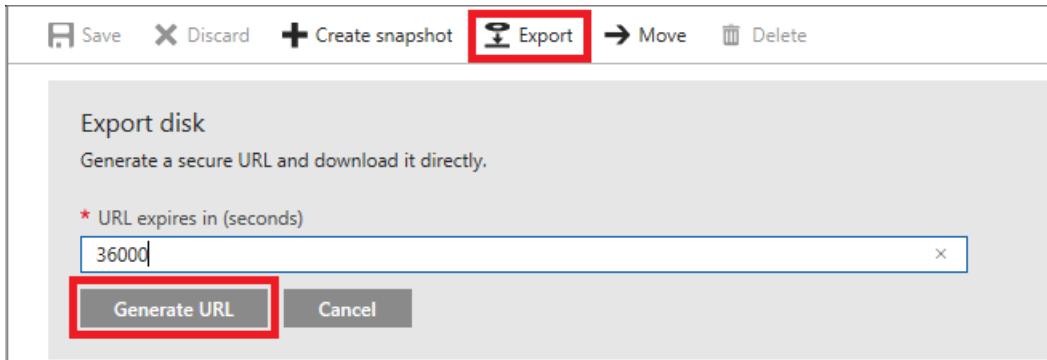
1. On the Hub menu in the Azure portal, click **Virtual Machines**.
2. Select the VM from the list.
3. On the blade for the VM, click **Stop**.



Generate SAS URL

To download the VHD file, you need to generate a [shared access signature \(SAS\)](#) URL. When the URL is generated, an expiration time is assigned to the URL.

1. On the menu of the blade for the VM, click **Disks**.
2. Select the operating system disk for the VM, and then click **Export**.
3. Set the expiration time of the URL to 36000.
4. Click **Generate URL**.

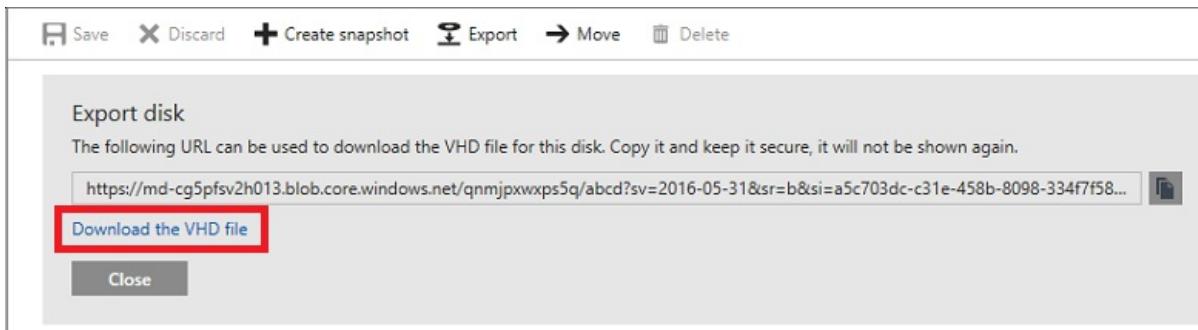


NOTE

The expiration time is increased from the default to provide enough time to download the large VHD file for a Windows Server operating system. You can expect a VHD file that contains the Windows Server operating system to take several hours to download depending on your connection. If you are downloading a VHD for a data disk, the default time is sufficient.

Download VHD

1. Under the URL that was generated, click Download the VHD file.



2. You may need to click **Save** in the browser to start the download. The default name for the VHD file is *abcd*.



Next steps

- Learn how to [upload a VHD file to Azure](#).
- [Create managed disks from unmanaged disks in a storage account](#).
- [Manage Azure disks with PowerShell](#).

What are virtual machine scale sets in Azure?

12/20/2017 • 10 min to read • [Edit Online](#)

Virtual machine scale sets are an Azure compute resource that you can use to deploy and manage a set of identical VMs. With all VMs configured the same, scale sets are designed to support true autoscale, and no pre-provisioning of VMs is required. So it's easier to build large-scale services that target big compute, large data, and containerized workloads.

For applications that need to scale compute resources out and in, scale operations are implicitly balanced across fault and update domains. For a further introduction to scale sets, refer to the [Azure blog announcement](#).

For more information about scale sets, watch these videos:

- [Mark Russinovich talks Azure scale sets](#)
- [Virtual Machine Scale Sets with Guy Bowerman](#)

Creating and managing scale sets

You can create a scale set in the [Azure portal](#) by selecting **new** and typing **scale** on the search bar. **Virtual machine scale set** is listed in the results. From there, you can fill in the required fields to customize and deploy your scale set. You also have options to set up basic autoscale rules based on CPU usage in the portal. To manage your scale set, you can use the Azure portal, [Azure PowerShell cmdlets](#), or the Azure CLI 2.0.

Scale sets can be deployed to an [availability zone](#).

NOTE

Currently virtual machine scale sets only supports deploying to a single availability zone. Multi-zone deployment will be supported in the future.

You can define and deploy scale sets by using JSON templates and [REST APIs](#), just like individual Azure Resource Manager VMs. Therefore, you can use any standard Azure Resource Manager deployment methods. For more information about templates, see [Authoring Azure Resource Manager templates](#).

You can find a set of example templates for virtual machine scale sets in the [Azure Quickstart templates GitHub repository](#). (Look for templates with **vmss** in the title.)

For the Quickstart template examples, a "Deploy to Azure" button in the readme for each template links to the portal deployment feature. To deploy the scale set, click the button and then fill in any parameters that are required in the portal.

Autoscale

To maintain consistent application performance, you can automatically increase or decrease the number of VM instances in your scale set. This autoscale ability reduces the management overhead to monitor and tune your scale set as customer demand changes over time. You define rules based on performance metrics, application response, or a fixed schedule, and your scale set autoscales as needed.

For basic autoscale rules, you can use host-based performance metrics such as CPU usage or disk I/O. These host-based metrics are available automatically, with no additional agents or extensions to install and configure.

Autoscale rules that use host-based metrics can be created with one of the following tools:

- [Azure portal](#)
- [Azure PowerShell](#)
- [Azure CLI 2.0](#)

To use more granular performance metrics, you can install and configure the Azure diagnostic extension on VM instances in your scale set. The Azure diagnostic extension allows you to collect additional performance metrics, such as memory consumption, from inside of each VM instance. These performance metrics are streamed to an Azure storage account, and you create autoscale rules to consume this data. For more information, see the articles for how to enable the Azure diagnostics extension on a [Linux VM](#) or [Windows VM](#).

To monitor the application performance itself, you can install and configure a small instrumentation package in to your application for App Insights. Detailed performance metrics for the application response time or number of sessions can then be streamed back from your app. You can then create autoscale rules with defined thresholds for the application-level performance itself. For more information about App Insights, see [What is Application Insights](#).

Manually scaling a scale set out and in

You can manually change the capacity of a scale set in the Azure portal by clicking the **Scaling** section under **Settings**.

To change scale set capacity on the command line, use the **scale** command in [Azure CLI](#). For example, use this command to set a scale set to a capacity of 10 VMs:

```
az vmss scale -g resourcegroupname -n scalesetname --new-capacity 10
```

To set the number of VMs in a scale set by using PowerShell, use the **Update-AzureRmVmss** command:

```
$vmss = Get-AzureRmVmss -ResourceGroupName resourcegroupname -VMScaleSetName scalesetname
$vmss.Sku.Capacity = 10
Update-AzureRmVmss -ResourceGroupName resourcegroupname -Name scalesetname -VirtualMachineScaleSet $vmss
```

To increase or decrease the number of virtual machines in a scale set by using an Azure Resource Manager template, change the **capacity** property and redeploy the template. This simplicity makes it easy to integrate scale sets with Azure Autoscale, or to write your own custom scaling layer if you need to define custom scale events that Azure Autoscale does not support.

If you are redeploying an Azure Resource Manager template to change the capacity, you can define a much smaller template that includes only the **SKU** property packet with the updated capacity. [Here's an example](#).

Monitoring your scale set

The [Azure portal](#) lists scale sets and shows their properties. The portal also supports management operations. You can perform management operations on both scale sets and individual VMs within a scale set. The portal also provides a customizable resource usage graph.

If you need to see or edit the underlying JSON definition of an Azure resource, you can also use [Azure Resource Explorer](#). Scale sets are a resource under the Microsoft.Compute Azure resource provider. From this site, you can see them by expanding the following links:

Subscriptions > your subscription > resourceGroups > providers > Microsoft.Compute > virtualMachineScaleSets > your scale set > etc.

Scale set scenarios

This section lists some typical scale set scenarios. Some higher-level Azure services (like Batch, Service Fabric, and

Container Service) use these scenarios.

- **Use RDP or SSH to connect to scale set instances:** A scale set is created inside a virtual network, and individual VMs in the scale set are not allocated public IP addresses by default. This policy avoids the expense and management overhead of allocating separate public IP addresses to all the nodes in your compute grid. If you do need direct external connections to scale set VMs, you can configure a scale set to automatically assign public IP addresses to new VMs. Alternatively you can connect to VMs from other resources in your virtual network that can be allocated public IP addresses, for example, load balancers and standalone virtual machines.
- **Connect to VMs by using NAT rules:** You can create a public IP address, assign it to a load balancer, and define an inbound NAT pool. These actions map ports on the IP address to a port on a VM in the scale set. For example:

SOURCE	SOURCE PORT	DESTINATION	DESTINATION PORT
Public IP	Port 50000	vmss_0	Port 22
Public IP	Port 50001	vmss_1	Port 22
Public IP	Port 50002	vmss_2	Port 22

In [this example](#), NAT rules are defined to enable an SSH connection to every VM in a scale set, by using a single public IP address.

[This example](#) does the same with RDP and Windows.

- **Connect to VMs by using a "jumpbox":** If you create a scale set and a standalone VM in the same virtual network, the standalone VM and the scale set VM can connect to one another by using their internal IP addresses, as defined by the virtual network or subnet. If you create a public IP address and assign it to the standalone VM, you can use RDP or SSH to connect to the standalone VM. You can then connect from that machine to your scale set instances. You might notice at this point that a simple scale set is inherently more secure than a simple standalone VM with a public IP address in its default configuration.

For example, [this template](#) deploys a simple scale set with a standalone VM.

- **Load balancing to scale set instances:** If you want to deliver work to a compute cluster of VMs by using a round-robin approach, you can configure an Azure load balancer with layer-4 load-balancing rules accordingly. You can define probes to verify that your application is running by pinging ports with a specified protocol, interval, and request path. [Azure Application Gateway](#) also supports scale sets, along with layer-7 and more sophisticated load-balancing scenarios.

[This example](#) creates a scale set that runs Apache web servers, and it uses a load balancer to balance the load that each VM receives. (Look at the Microsoft.Network/loadBalancers resource type and networkProfile and extensionProfile in virtualMachineScaleSet.)

[This Linux example](#) and [this Windows example](#) use Application Gateway.

- **Deploying a scale set as a compute cluster in a PaaS cluster manager:** Scale sets are sometimes described as a next-generation worker role. Though a valid description, it does run the risk of confusing scale set features with Azure Cloud Services features. In a sense, scale sets provide a true worker role or worker resource. They are a generalized compute resource that is platform/runtime independent, is customizable, and integrates into Azure Resource Manager IaaS.

A Cloud Services worker role is limited in terms of platform/runtime support (Windows platform images only). But it also includes services such as VIP swap, configurable upgrade settings, and runtime/app deployment-specific settings. These services are not yet available in scale sets, or they're delivered by other higher-level PaaS services like Azure Service Fabric. You can look at scale sets as an infrastructure that

supports PaaS. PaaS solutions like [Service Fabric](#) build on this infrastructure.

In [this example](#) of this approach, [Azure Container Service](#) deploys a cluster based on scale sets with a container orchestrator.

Scale set performance and scale guidance

- A scale set supports up to 1,000 VMs. If you create and upload your own custom VM images, the limit is 300. For considerations in using large scale sets, see [Working with large virtual machine scale sets](#).
- You do not have to pre-create Azure storage accounts to use scale sets. Scale sets support Azure managed disks, which negate performance concerns about the number of disks per storage account. For more information, see [Azure virtual machine scale sets and managed disks](#).
- Consider using Azure Premium Storage instead of Azure Storage for faster, more predictable VM provisioning times, and improved I/O performance.
- The vCPU quota in the region in which you are deploying limits the number of VMs you can create. You might need to contact Customer Support to increase your compute quota limit, even if you have a high limit of vCPUs for use with Azure Cloud Services today. To query your quota, run this Azure CLI command: `az vm list-usage`. Or, run this PowerShell command: `Get-AzureRmVMUsage`.

Frequently asked questions for scale sets

Q. How many VMs can I have in a scale set?

A. A scale set can have 0 to 1,000 VMs based on platform images, or 0 to 300 VMs based on custom images.

Q. Are data disks supported within scale sets?

A. Yes. A scale set can define an attached data disks configuration that applies to all VMs in the set. For more information, see [Azure scale sets and attached data disks](#). Other options for storing data include:

- Azure files (SMB shared drives)
- OS drive
- Temp drive (local, not backed by Azure Storage)
- Azure data service (for example, Azure tables, Azure blobs)
- External data service (for example, remote database)

Q. Which Azure regions support scale sets?

A. All regions support scale sets.

Q. How do I create a scale set by using a custom image?

A. Create a managed disk based on your custom image VHD and reference it in your scale set template. [Here's an example](#).

Q. If I reduce my scale set capacity from 20 to 15, which VMs are removed?

A. Virtual machines are removed from the scale set evenly across update domains and fault domains to maximize availability. VMs with the highest IDs are removed first.

Q. What if I then increase the capacity from 15 to 18?

A. If you increase capacity to 18, then 3 new VMs are created. Each time, the VM instance ID is incremented from the previous highest value (for example, 20, 21, 22). VMs are balanced across fault domains and update domains.

Q. When I'm using multiple extensions in a scale set, can I enforce an execution sequence?

A. Not directly, but for the customScript extension, your script can wait for another extension to finish. You can get

additional guidance on extension sequencing in the blog post [Extension Sequencing in Azure virtual machine scale sets](#).

Q. Do scale sets work with Azure availability sets?

A. Yes. A scale set is an implicit availability set with five fault domains and five update domains. Scale sets of more than 100 VMs span multiple *placement groups*, which are equivalent to multiple availability sets. For more information about placement groups, see [Working with large virtual machine scale sets](#). An availability set of VMs can exist in the same virtual network as a scale set of VMs. A common configuration is to put control node VMs (which often require unique configuration) in an availability set and put data nodes in the scale set.

You can find more answers to questions about scale sets in the [Azure virtual machine scale sets FAQ](#).

Manage the availability of Windows virtual machines in Azure

12/8/2017 • 8 min to read • [Edit Online](#)

Learn ways to set up and manage multiple virtual machines to ensure high availability for your Windows application in Azure. You can also [manage the availability of Linux virtual machines](#).

For instructions on creating and using availability sets when using the classic deployment model, see [How to Configure an Availability Set](#).

Understand VM Reboots - maintenance vs. downtime

There are three scenarios that can lead to virtual machine in Azure being impacted: unplanned hardware maintenance, unexpected downtime, and planned maintenance.

- **Unplanned Hardware Maintenance Event** occurs when the Azure platform predicts that the hardware or any platform component associated to a physical machine, is about to fail. When the platform predicts a failure, it will issue an unplanned hardware maintenance event to reduce the impact to the virtual machines hosted on that hardware. Azure uses Live Migration technology to migrate the Virtual Machines from the failing hardware to a healthy physical machine. Live Migration is a VM preserving operation that only pauses the Virtual Machine for a short time. Memory, open files, and network connections are maintained, but performance might be reduced before and/or after the event. In cases where Live Migration cannot be used, the VM will experience Unexpected Downtime, as described below.
- **An Unexpected Downtime** rarely occurs when the hardware or the physical infrastructure underlying your virtual machine has faulted in some way. This may include local network failures, local disk failures, or other rack level failures. When such a failure is detected, the Azure platform automatically migrates (heals) your virtual machine to a healthy physical machine in the same datacenter. During the healing procedure, virtual machines experience downtime (reboot) and in some cases loss of the temporary drive. The attached OS and data disks are always preserved.

Virtual machines can also experience downtime in the unlikely event of an outage or disaster that affects an entire datacenter, or even an entire region. For these scenarios, Azure provides protection options including [availability zones](#) and [paired regions](#).

- **Planned Maintenance events** are periodic updates made by Microsoft to the underlying Azure platform to improve overall reliability, performance, and security of the platform infrastructure that your virtual machines run on. Most of these updates are performed without any impact upon your Virtual Machines or Cloud Services (see [VM Preserving Maintenance](#)). While the Azure platform attempts to use VM Preserving Maintenance in all possible occasions, there are rare instances when these updates require a reboot of your virtual machine to apply the required updates to the underlying infrastructure. In this case, you can perform Azure Planned Maintenance with Maintenance-Redeploy operation by initiating the maintenance for their VMs in the suitable time window. For more information, see [Planned Maintenance for Virtual Machines](#).

To reduce the impact of downtime due to one or more of these events, we recommend the following high availability best practices for your virtual machines:

- [Configure multiple virtual machines in an availability set for redundancy](#)
- [Use managed disks for VMs in an availability set](#)
- [Use Scheduled Events to proactively response to VM impacting events](#)

- [Configure each application tier into separate availability sets](#)
- [Combine a Load Balancer with availability sets](#)
- [Use availability zones to protect from datacenter level failures](#)

Configure multiple virtual machines in an availability set for redundancy

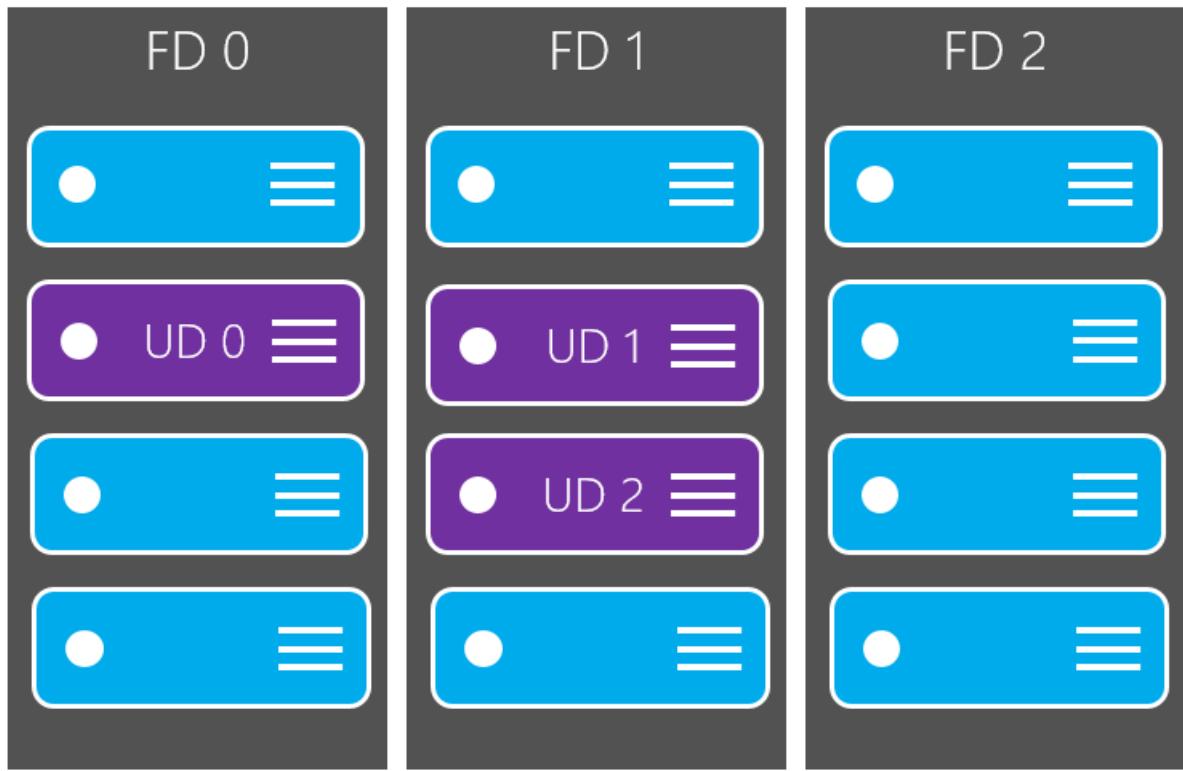
To provide redundancy to your application, we recommend that you group two or more virtual machines in an availability set. This configuration within a datacenter ensures that during either a planned or unplanned maintenance event, at least one virtual machine is available and meets the 99.95% Azure SLA. For more information, see the [SLA for Virtual Machines](#).

IMPORTANT

Avoid leaving a single instance virtual machine in an availability set by itself. VMs in this configuration do not qualify for a SLA guarantee and face downtime during Azure planned maintenance events, except when a single VM is using [Azure Premium Storage](#). For single VMs using premium storage, the Azure SLA applies.

Each virtual machine in your availability set is assigned an **update domain** and a **fault domain** by the underlying Azure platform. For a given availability set, five non-user-configurable update domains are assigned by default (Resource Manager deployments can then be increased to provide up to 20 update domains) to indicate groups of virtual machines and underlying physical hardware that can be rebooted at the same time. When more than five virtual machines are configured within a single availability set, the sixth virtual machine is placed into the same update domain as the first virtual machine, the seventh in the same update domain as the second virtual machine, and so on. The order of update domains being rebooted may not proceed sequentially during planned maintenance, but only one update domain is rebooted at a time. A rebooted update domain is given 30 minutes to recover before maintenance is initiated on a different update domain.

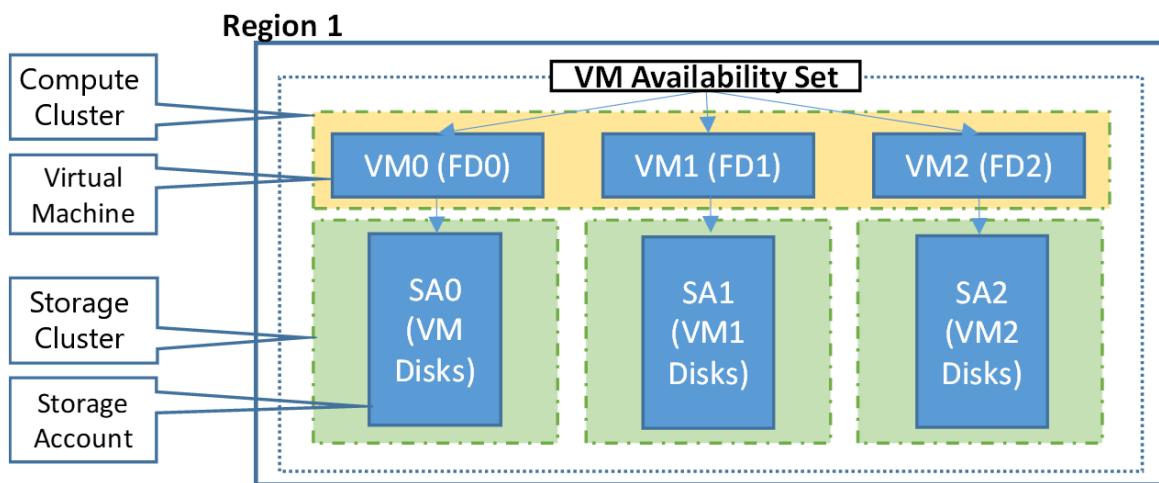
Fault domains define the group of virtual machines that share a common power source and network switch. By default, the virtual machines configured within your availability set are separated across up to three fault domains for Resource Manager deployments (two fault domains for Classic). While placing your virtual machines into an availability set does not protect your application from operating system or application-specific failures, it does limit the impact of potential physical hardware failures, network outages, or power interruptions.



Use managed disks for VMs in an availability set

If you are currently using VMs with unmanaged disks, we highly recommend you [convert VMs in Availability Set to use Managed Disks](#).

Managed disks provide better reliability for Availability Sets by ensuring that the disks of VMs in an Availability Set are sufficiently isolated from each other to avoid single points of failure. It does this by automatically placing the disks in different storage clusters. If a storage cluster fails due to hardware or software failure, only the VM instances with disks on those stamps fail.



IMPORTANT

The number of fault domains for managed availability sets varies by region - either two or three per region. The following table shows the number per region

Number of Fault Domains per region

REGION	MAX # OF FAULT DOMAINS
East US	3
East US 2	3
West US	3
West US 2	2
Central US	3
North Central US	3
South Central US	3
West Central US	2
Canada Central	2
Canada East	2
North Europe	3
West Europe	3
UK South	2
UK West	2
East Asia	2
South East Asia	2
Japan East	2
Japan West	2
South India	2
Central India	2
West India	2
Korea Central	2
Korea South	2
Australia East	2

REGION	MAX # OF FAULT DOMAINS
Australia Southeast	2
Brazil South	2
US Gov Virginia	2
US Gov Texas	2
US Gov Arizona	2
US DoD Central	2
US DoD East	2

If you plan to use VMs with [unmanaged disks](#), follow below best practices for Storage accounts where virtual hard disks (VHDs) of VMs are stored as [page blobs](#).

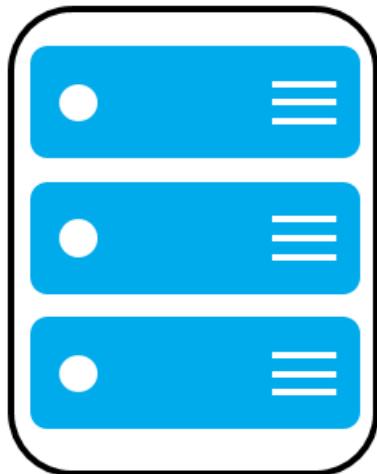
- 1. Keep all disks (OS and data) associated with a VM in the same storage account**
- 2. Review the limits on the number of unmanaged disks in a Storage account** before adding more VHDs to a storage account
- 3. Use separate storage account for each VM in an Availability Set.** Do not share Storage accounts with multiple VMs in the same Availability Set. It is acceptable for VMs across different Availability Sets to share storage accounts if above best practices are followed

Configure each application tier into separate availability sets

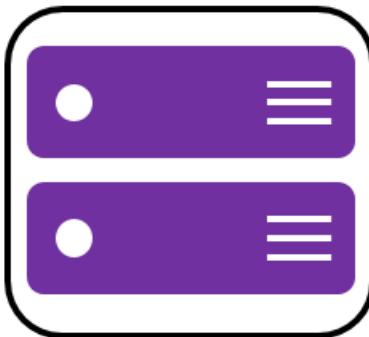
If your virtual machines are all nearly identical and serve the same purpose for your application, we recommend that you configure an availability set for each tier of your application. If you place two different tiers in the same availability set, all virtual machines in the same application tier can be rebooted at once. By configuring at least two virtual machines in an availability set for each tier, you guarantee that at least one virtual machine in each tier is available.

For example, you could put all the virtual machines in the front end of your application running IIS, Apache, Nginx in a single availability set. Make sure that only front-end virtual machines are placed in the same availability set. Similarly, make sure that only data-tier virtual machines are placed in their own availability set, like your replicated SQL Server virtual machines, or your MySQL virtual machines.

Web Tier Availability Set



Data Tier Availability Set



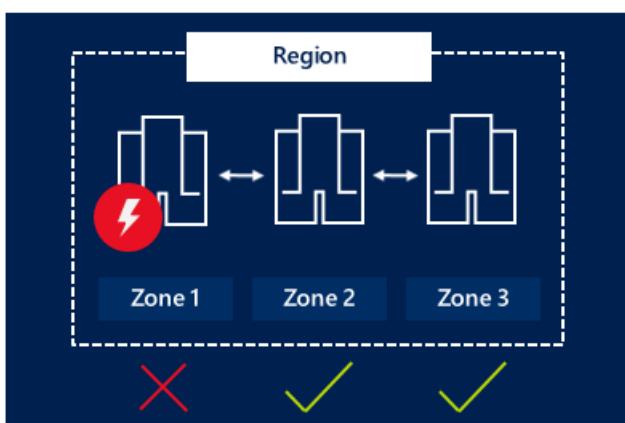
Combine a load balancer with availability sets

Combine the [Azure Load Balancer](#) with an availability set to get the most application resiliency. The Azure Load Balancer distributes traffic between multiple virtual machines. For our Standard tier virtual machines, the Azure Load Balancer is included. Not all virtual machine tiers include the Azure Load Balancer. For more information about load balancing your virtual machines, see [Load Balancing virtual machines](#).

If the load balancer is not configured to balance traffic across multiple virtual machines, then any planned maintenance event affects the only traffic-serving virtual machine, causing an outage to your application tier. Placing multiple virtual machines of the same tier under the same load balancer and availability set enables traffic to be continuously served by at least one instance.

Use availability zones to protect from datacenter level failures

[Availability zones](#) (preview), an alternative to availability sets, expand the level of control you have to maintain the availability of the applications and data on your VMs. An Availability Zone is a physically separate zone within an Azure region. There are three Availability Zones per supported Azure region. Each Availability Zone has a distinct power source, network, and cooling, and is logically separate from the other Availability Zones within the Azure region. By architecting your solutions to use replicated VMs in zones, you can protect your apps and data from the loss of a datacenter. If one zone is compromised, then replicated apps and data are instantly available in another zone.



NOTE

Azure Availability Zones is in preview and is ready for your development and test scenarios. Support is available for select Azure resources, regions, and size families. For more information on how to get started, and which Azure resources, regions, and size families you can try with Availability Zones, see [Overview of Availability Zones](#). You can [provide feedback](#) on the Azure website. For support, contact [StackOverflow](#) or open an [Azure support ticket](#).

Learn more about deploying a [Windows](#) or [Linux](#) VM in an Availability Zone.

Next steps

To learn more about load balancing your virtual machines, see [Load Balancing virtual machines](#).

Vertically scale Windows VMs with Azure Automation

6/27/2017 • 2 min to read • [Edit Online](#)

Vertical scaling is the process of increasing or decreasing the resources of a machine in response to the workload. In Azure this can be accomplished by changing the size of the Virtual Machine. This can help in the following scenarios

- If the Virtual Machine is not being used frequently, you can resize it down to a smaller size to reduce your monthly costs
- If the Virtual Machine is seeing a peak load, it can be resized to a larger size to increase its capacity

The outline for the steps to accomplish this is as below

1. Setup Azure Automation to access your Virtual Machines
2. Import the Azure Automation Vertical Scale runbooks into your subscription
3. Add a webhook to your runbook
4. Add an alert to your Virtual Machine

NOTE

Because of the size of the first Virtual Machine, the sizes it can be scaled to, may be limited due to the availability of the other sizes in the cluster current Virtual Machine is deployed in. In the published automation runbooks used in this article we take care of this case and only scale within the below VM size pairs. This means that a Standard_D1v2 Virtual Machine will not suddenly be scaled up to Standard_G5 or scaled down to Basic_A0.

VM SIZES SCALING PAIR	
Basic_A0	Basic_A4
Standard_A0	Standard_A4
Standard_A5	Standard_A7
Standard_A8	Standard_A9
Standard_A10	Standard_A11
Standard_D1	Standard_D4
Standard_D11	Standard_D14
Standard_DS1	Standard_DS4
Standard_DS11	Standard_DS14
Standard_D1v2	Standard_D5v2
Standard_D11v2	Standard_D14v2
Standard_G1	Standard_G5
Standard_GS1	Standard_GS5

Setup Azure Automation to access your Virtual Machines

The first thing you need to do is create an Azure Automation account that will host the runbooks used to scale a Virtual Machine. Recently the Automation service introduced the "Run As account" feature which makes setting up the Service Principal for automatically running the runbooks on the user's behalf very easy. You can read more about this in the article below:

- [Authenticate Runbooks with Azure Run As account](#)

Import the Azure Automation Vertical Scale runbooks into your subscription

The runbooks that are needed for Vertically Scaling your Virtual Machine are already published in the Azure Automation Runbook Gallery. You will need to import them into your subscription. You can learn how to import runbooks by reading the following article.

- [Runbook and module galleries for Azure Automation](#)

The runbooks that need to be imported are shown in the image below

The screenshot shows the 'Browse Gallery' interface for Azure Automation. It displays a list of runbooks with their details and creation metadata.

Runbook Title	Description	Created by	Ratings	Downloads	Last update
PowerShell Runbook	Script will walk you through checking if your database has been granted Premium database quota. Followed by how to upgrade reservations on a database to premium.	Windows Azure Product Team	0 of 5	1,122	10/16/2014
Create Availability Group Listener in Windows Azure VMs (Cloud-Only)	**Note:** Premium database quota must be requested for your server via the Windows Azure Management Portal	Cephas Lin	5 of 5	1,813	11/22/2013
Create Availability Group Listener in Hybrid IT	This script configures an availability group listener for an availability group that is running in hybrid IT. It automatically configures the Windows Azure settings and also configures each cluster node on-premise and in Windows Azure remotely using PowerShell remoting.	Cephas Lin	0 of 5	1,095	11/22/2013
Vertically scale up an Azure Resource Manager VM with Azure Automation	This Azure Automation runbook can help you vertically scale up an Azure Resource Manager VM in response to an alert.	Kay Singh	0 of 5	0	3/29/2016
Vertically scale down Azure Resource Manager VM with Azure Automation	This Azure Automation runbook can help you vertically scale down an Azure Resource Manager VM in response to an alert.	Kay Singh	0 of 5	0	3/29/2016

Add a webhook to your runbook

Once you've imported the runbooks you'll need to add a webhook to the runbook so it can be triggered by an alert from a Virtual Machine. The details of creating a webhook for your Runbook can be read here

- Azure Automation webhooks

Make sure you copy the webhook before closing the webhook dialog as you will need this in the next section.

Add an alert to your Virtual Machine

1. Select Virtual Machine settings
2. Select "Alert rules"
3. Select "Add alert"
4. Select a metric to fire the alert on
5. Select a condition, which when fulfilled will cause the alert to fire
6. Select a threshold for the condition in Step 5. to be fulfilled
7. Select a period over which the monitoring service will check for the condition and threshold in Steps 5 & 6
8. Paste in the webhook you copied from the previous section.

The screenshot shows two windows side-by-side. The left window is the 'armvm' Virtual machine blade, displaying basic details like Resource group (armrg), Computer name (armvm), Status (Running), Location (Southeast Asia), Subscription name (Visual Studio Ultimate with MSDN), and more. It also shows a monitoring chart for CPU percentage with a value of 0.41% and a note 'No available data.' The right window is the 'Settings' blade for the same VM, listing various support and troubleshooting options. A red circle with the number 1 highlights the 'All settings' button at the bottom of the left blade. A red circle with the number 2 highlights the 'Alert rules' link in the 'MONITORING' section of the right blade.

Alert rules

armvm

Add alert

3

NAME	CONDITION	LAST ACTIVE
ARMVM (VIRTUALMACHINES)		
downCpuLT40	CPU percentage guest OS > 40... Never	
upCpuGT75	CPU percentage guest OS > 60... Never	

Add an alert rule

4 * Metric ⓘ CPU percentage guest OS

1%
0.8%
0.6%
0.4%

Mar 29 6 AM 12 PM 6 PM

5 * Condition greater than

6 * Threshold ⓘ 1 %

7 * Period ⓘ Over the last 5 minutes

Email owners, contributors, and readers

Additional administrator email(s) Add email addresses separated by semicolons

8 Webhook ⓘ HTTP or HTTPS endpoint to route alerts to [Learn more about configuring webhooks](#)

Automation Runbook ⓘ Not configured

OK

Create a Windows virtual machine in an availability zone with PowerShell

9/22/2017 • 4 min to read • [Edit Online](#)

This article details using Azure PowerShell to create an Azure virtual machine running Windows Server 2016 in an Azure availability zone. An [availability zone](#) is a physically separate zone in an Azure region. Use availability zones to protect your apps and data from an unlikely failure or loss of an entire datacenter.

NOTE

Azure Availability Zones is in preview and is ready for your development and test scenarios. Support is available for select Azure resources, regions, and size families. For more information on how to get started, and which Azure resources, regions, and size families you can try with Availability Zones, see [Overview of Availability Zones](#). You can [provide feedback](#) on the Azure website. For support, contact [StackOverflow](#) or [open an Azure support ticket](#).

Make sure that you have installed the latest Azure PowerShell module. If you need to install or upgrade, see [Install Azure PowerShell module](#).

Log in to Azure

Log in to your Azure subscription with the `Login-AzureRmAccount` command and follow the on-screen directions.

```
Login-AzureRmAccount
```

Create resource group

Create an Azure resource group with [New-AzureRmResourceGroup](#). A resource group is a logical container into which Azure resources are deployed and managed. In this example, a resource group named *myResourceGroup* is created in the *eastus2* region. East US 2 is one of the Azure regions that supports availability zones in preview.

```
New-AzureRmResourceGroup -Name myResourceGroup -Location eastus2
```

Create networking resources

Create a virtual network, subnet, and a public IP address

These resources are used to provide network connectivity to the virtual machine and connect it to the internet. Create the IP address in an availability zone, 2 in this example. To create the VM in an availability zone (shown in a later step), you specify the same zone used to create the IP address.

```

# Create a subnet configuration
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig -Name mySubnet -AddressPrefix 192.168.1.0/24

# Create a virtual network
$vnet = New-AzureRmVirtualNetwork -ResourceGroupName myResourceGroup -Location eastus2 ` 
    -Name MYvNET -AddressPrefix 192.168.0.0/16 -Subnet $subnetConfig

# Create a public IP address in an availability zone and specify a DNS name
$pip = New-AzureRmPublicIpAddress -ResourceGroupName myResourceGroup -Location eastus2 -Zone 2 ` 
    -AllocationMethod Static -IdleTimeoutInMinutes 4 -Name "mypublicdns$(Get-Random)"

```

Create a network security group and a network security group rule

The network security group secures the virtual machine using inbound and outbound rules. In this case, an inbound rule is created for port 3389, which allows incoming remote desktop connections. We also want to create an inbound rule for port 80, which allows incoming web traffic.

```

# Create an inbound network security group rule for port 3389
$nsgRuleRDP = New-AzureRmNetworkSecurityRuleConfig -Name myNetworkSecurityGroupRuleRDP -Protocol Tcp ` 
    -Direction Inbound -Priority 1000 -SourceAddressPrefix * -SourcePortRange * -DestinationAddressPrefix * ` 
    -DestinationPortRange 3389 -Access Allow

# Create an inbound network security group rule for port 80
$nsgRuleWeb = New-AzureRmNetworkSecurityRuleConfig -Name myNetworkSecurityGroupRuleWWW -Protocol Tcp ` 
    -Direction Inbound -Priority 1001 -SourceAddressPrefix * -SourcePortRange * -DestinationAddressPrefix * ` 
    -DestinationPortRange 80 -Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName myResourceGroup -Location eastus2 ` 
    -Name myNetworkSecurityGroup -SecurityRules $nsgRuleRDP,$nsgRuleWeb

```

Create a network card for the virtual machine

Create a network card with [New-AzureRmNetworkInterface](#) for the virtual machine. The network card connects the virtual machine to a subnet, network security group, and public IP address.

```

# Create a virtual network card and associate with public IP address and NSG
$nic = New-AzureRmNetworkInterface -Name myNic -ResourceGroupName myResourceGroup -Location eastus2 ` 
    -SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $pip.Id -NetworkSecurityGroupId $nsg.Id

```

Create virtual machine

Create a virtual machine configuration. This configuration includes the settings that are used when deploying the virtual machine such as a virtual machine image, size, and authentication configuration. The *Standard_DS1_v2* size in this example is supported in the availability zones preview. This configuration also specifies the availability zone you set when creating the IP address. When running this step, you are prompted for credentials. The values that you enter are configured as the user name and password for the virtual machine.

```

# Define a credential object
$cred = Get-Credential

# Create a virtual machine configuration
$vmConfig = New-AzureRmVMConfig -VMName myVM -VMSize Standard_DS1_v2 -Zone 2 | ` 
    Set-AzureRmVMOperatingSystem -Windows -ComputerName myVM -Credential $cred | ` 
    Set-AzureRmVMSourceImage -PublisherName MicrosoftWindowsServer -Offer WindowsServer ` 
    -Skus 2016-Datacenter -Version latest | Add-AzureRmVMNetworkInterface -Id $nic.Id

```

Create the virtual machine with [New-AzureRmVM](#).

```
New-AzureRmVM -ResourceGroupName myResourceGroup -Location eastus2 -VM $vmConfig
```

Zone for IP address and managed disk

You created the VM's IP address resource in the same availability zone as the VM. The managed disk resource for the VM is also created in the same availability zone. You can verify this with [Get-AzureRmDisk](#):

```
Get-AzureRmDisk -ResourceGroupName myResourceGroup
```

The output shows that the managed disk is in the same availability zone as the VM:

```
ResourceGroupName : myResourceGroup
AccountType      : PremiumLRS
OwnerId          : /subscriptions/d5b9d4b7-6fc1-0000-0000-
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.
                           Compute/virtualMachines/myVM
ManagedBy        : /subscriptions/d5b9d4b7-6fc1-0000-0000-
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.
                           Compute/virtualMachines/myVM
Sku              : Microsoft.Azure.Management.Compute.Models.DiskSku
Zones            : {2}
TimeCreated       : 9/7/2017 6:57:26 PM
OsType           : Windows
CreationData     : Microsoft.Azure.Management.Compute.Models.CreationData
DiskSizeGB       : 127
EncryptionSettings :
ProvisioningState : Succeeded
Id               : /subscriptions/d5b9d4b7-6fc1-0000-0000-
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.
                           Compute/disks/myVM_OsDisk_1_bd921920bb0a4650becfc2d83000000
Name             : myVM_OsDisk_1_bd921920bb0a4650becfc2d83000000
Type             : Microsoft.Compute/disks
Location         : eastus2
Tags             : {}
```

Next steps

In this article, you learned how to create a VM in an availability zone. Learn more about [regions and availability](#) for Azure VMs.

Create a Windows virtual machine in an availability zone with the Azure portal

9/22/2017 • 2 min to read • [Edit Online](#)

This article steps through using the Azure portal to create a virtual machine in an Azure availability zone. An [availability zone](#) is a physically separate zone in an Azure region. Use availability zones to protect your apps and data from an unlikely failure or loss of an entire datacenter.

NOTE

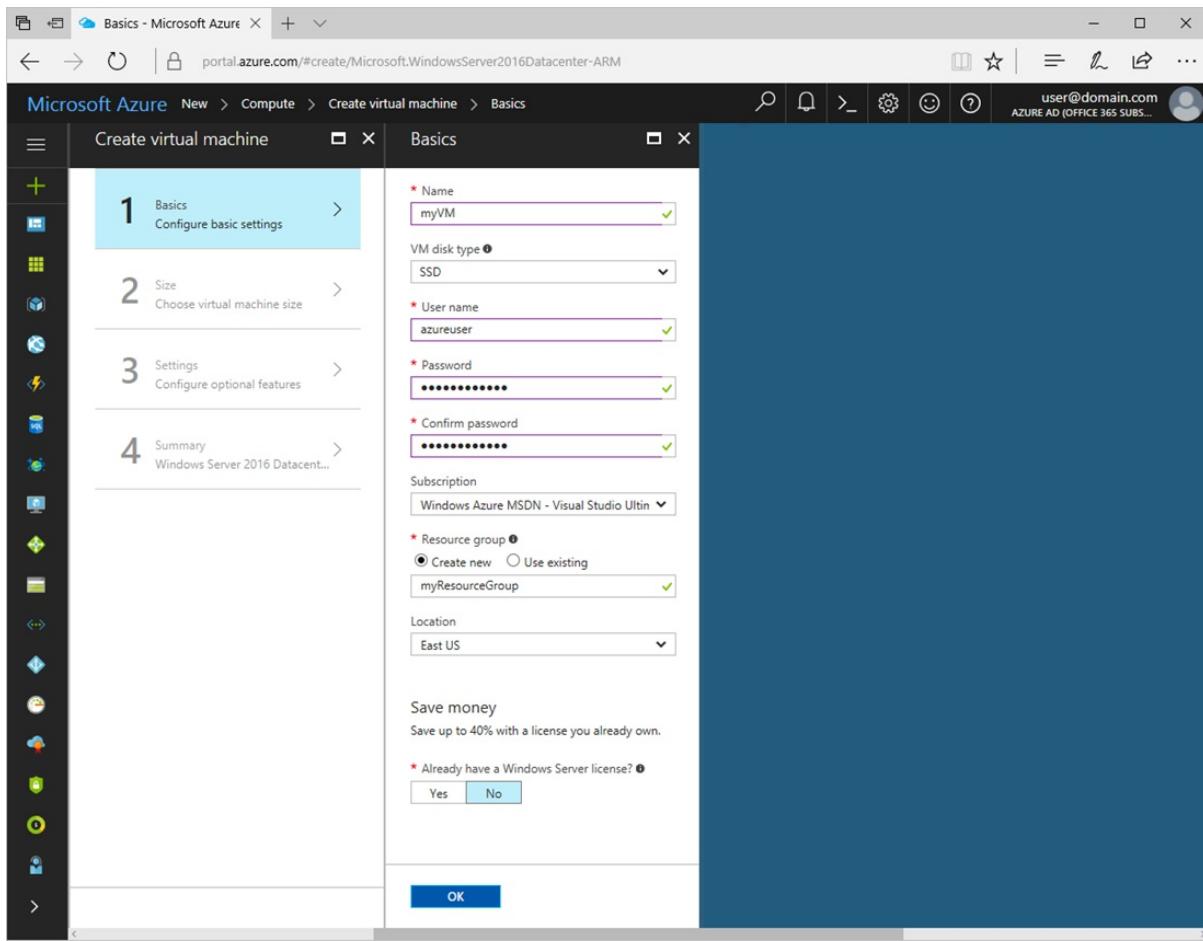
Azure Availability Zones is in preview and is ready for your development and test scenarios. Support is available for select Azure resources, regions, and size families. For more information on how to get started, and which Azure resources, regions, and size families you can try with Availability Zones, see [Overview of Availability Zones](#). You can [provide feedback](#) on the Azure website. For support, contact [StackOverflow](#) or [open an Azure support ticket](#).

Log in to Azure

Log in to the Azure portal at <https://portal.azure.com>.

Create virtual machine

1. Click the **New** button found on the upper left-hand corner of the Azure portal.
2. Select **Compute**, and then select **Windows Server 2016 Datacenter**.
3. Enter the virtual machine information. The user name and password entered here is used to log in to the virtual machine. When complete, click **OK**.



4. Select a size for the VM. To see more sizes, select **View all** or change the **Supported disk type** filter. Take care to select one of the sizes supported in the availability zones preview, such as *DS1_v2 Standard*.

Choose a size
Browse the available sizes and their features

Prices presented are estimates in your local currency that include only Azure infrastructure costs and any discounts for the subscription and location. The prices don't include any applicable software costs. Recommended sizes are determined by the publisher of the selected image based on hardware and software requirements.

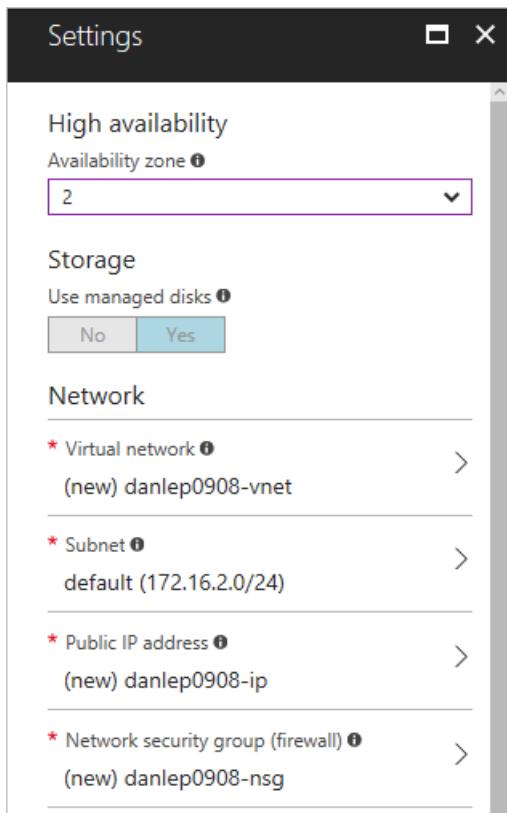
Supported disk type	Minimum memory (GiB)	Minimum cores
SSD	0	1

★ Recommended | [View all](#)

Size	Cores	Data disks	Max IOPS	Local SSD	Premium disk support	Price (USD/MONTH ESTIMATED)
DS1_V2 Standard	2	2	3200	7 GB	Load balancing	96.72
DS2_V2 Standard	7	4	6400	14 GB	Load balancing	193.44
DS11_V2 Standard	14	4	6400	28 GB	Load balancing	223.20

5. Under **Settings > High availability**, select one of the numbered zones from the **Availability zone**

dropdown, keep the remaining defaults, and click **OK**.



6. On the summary page, click **Purchase** to start the virtual machine deployment.
7. The VM will be pinned to the Azure portal dashboard. Once the deployment has completed, the VM summary automatically opens.

Zone for IP address and managed disk

When the VM is deployed in an availability zone, the IP address and managed disk resources are deployed in the same availability zone. You can confirm the zone settings using Azure PowerShell. If you need to install or upgrade, see [Install Azure PowerShell module](#).

The following examples get information about the resources in a resource group named *myResourceGroup*. Substitute the name of the resource group you used to create the VM.

Find the zone of the public IP address with [Get-AzureRmPublicIpAddress](#):

```
Get-AzureRmPublicIpAddress -ResourceGroupName myResourceGroup
```

The `Zones` setting in the output shows that the public IP address is in the same availability zone as the VM:

```

Name          : myVM-ip
ResourceGroupName   : myResourceGroup
Location       : eastus2
Id            : /subscriptions/e44f251c-c67e-4760-9ed6-
bf99a306ecff/resourceGroups/danlep0911/providers/Micr
                osoft.Network/publicIPAddresses/myVM-ip
Etag          : W/"b67e14c0-7e8a-4d12-91c5-da2a5dfad132"
ResourceGuid    : 314bf57d-9b25-4474-9282-db3561d536aa
ProvisioningState : Succeeded
Tags          :
PublicIpAllocationMethod : Dynamic
IpAddress      : 13.68.16.25
PublicIpAddressVersion : IPv4
IdleTimeoutInMinutes : 4
IpConfiguration : {
    "Id": "/subscriptions/e44f251c-c67e-4760-9ed6-
bf99a306ecff/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myVM11842/ipConfigura
tions/ipconfig1"
}
DnsSettings    : null
Zones         : {2}

```

The managed disk resource for the VM is also created in the same availability zone. You can verify this with [Get-AzureRmDisk](#):

```
Get-AzureRmDisk -ResourceGroupName myResourceGroup
```

The output shows that the managed disk is in the same availability zone as the VM:

```

ResourceGroupName  : myResourceGroup
AccountType       : PremiumLRS
OwnerId          : /subscriptions/d5b9d4b7-6fc1-0000-0000-
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.
                    Compute/virtualMachines/myVM
ManagedBy        : /subscriptions/d5b9d4b7-6fc1-0000-0000-
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.
                    Compute/virtualMachines/myVM
Sku              : Microsoft.Azure.Management.Compute.Models.DiskSku
Zones            : {2}
TimeCreated       : 9/7/2017 6:57:26 PM
OsType           : Windows
CreationData      : Microsoft.Azure.Management.Compute.Models.CreationData
DiskSizeGB        : 127
EncryptionSettings :
ProvisioningState : Succeeded
Id               : /subscriptions/d5b9d4b7-6fc1-0000-0000-
000000000000/resourceGroups/myResourceGroup/providers/Microsoft.
                    Compute/disks/myVM_OsDisk_1_bd921920bb0a4650becfc2d830000000
Name             : myVM_OsDisk_1_bd921920bb0a4650becfc2d830000000
Type             : Microsoft.Compute/disks
Location         : eastus2
Tags             : {}

```

Next steps

In this article, you learned how to create a VM in an availability zone. Learn more about [regions and availability](#) for Azure VMs.

Automating Azure virtual machine deployment with Chef

12/11/2017 • 6 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

Chef is a great tool for delivering automation and desired state configurations.

With the latest cloud api release, Chef provides seamless integration with Azure, giving you the ability to provision and deploy configuration states through a single command.

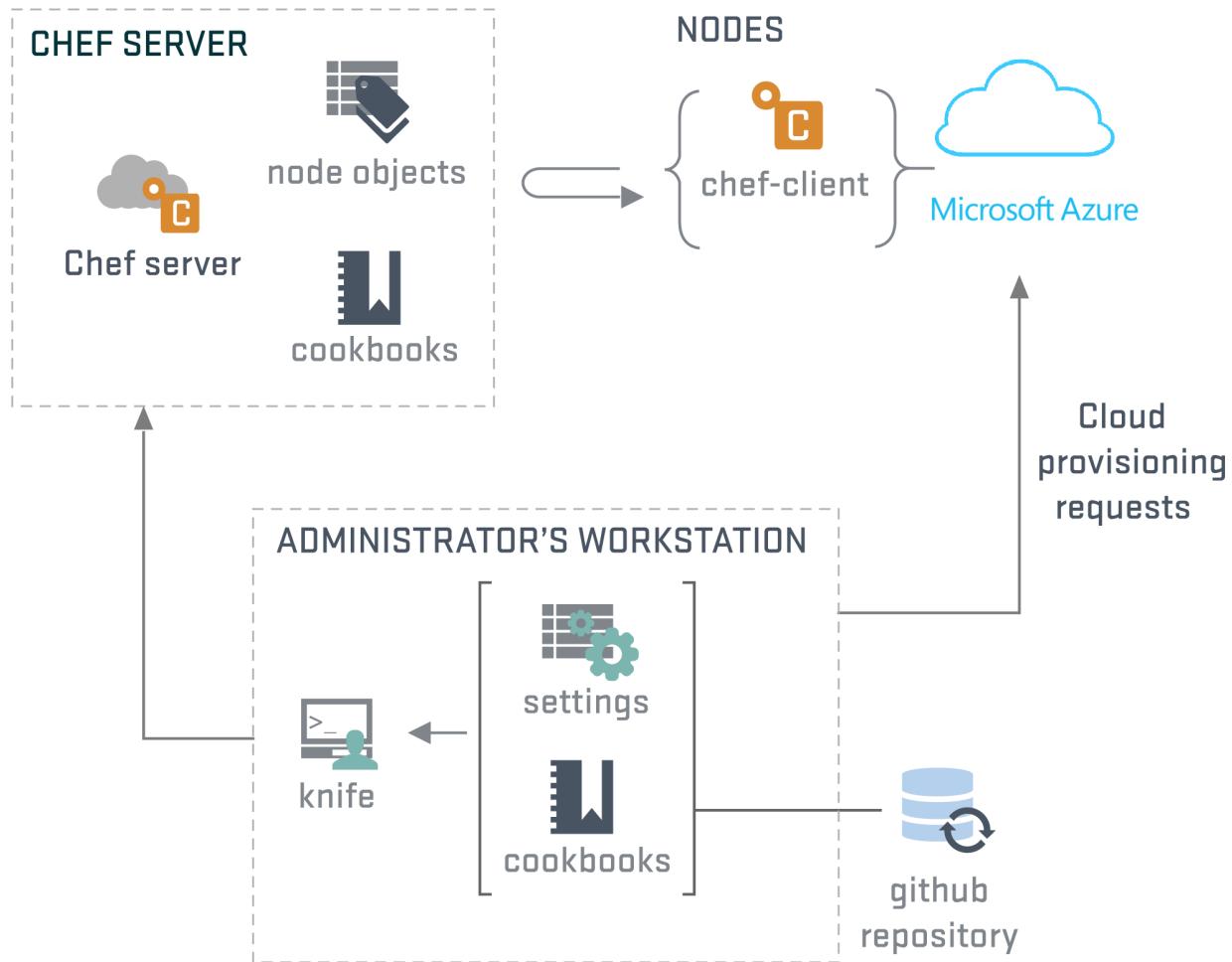
In this article, you set up your Chef environment to provision Azure virtual machines and walk through creating a policy or "CookBook" and then deploying this cookbook to an Azure virtual machine.

Let's begin!

Chef basics

Before you begin, [review the basic concepts of Chef](#).

The following diagram depicts the high-level Chef architecture.



Chef has three main architectural components: Chef Server, Chef Client (node), and Chef Workstation.

The Chef Server is the management point and there are two options for the Chef Server: a hosted solution or an on-premises solution. We will be using a hosted solution.

The Chef Client (node) is the agent that sits on the servers you are managing.

The Chef Workstation is the admin workstation where we create policies and execute management commands. We run the **knife** command from the Chef Workstation to manage the infrastructure.

There is also the concept of "Cookbooks" and "Recipes". These are effectively the policies we define and apply to the servers.

Preparing the workstation

First, let's prep the workstation. I'm using a standard Windows workstation. We need to create a directory to store the config files and cookbooks.

First create a directory called C:\chef.

Then create a second directory called c:\chef\cookbooks.

We now need to download the Azure settings file so Chef can communicate with the Azure subscription.

Download your publish settings using the PowerShell Azure [Get-AzurePublishSettingsFile](#) command.

Save the publish settings file in C:\chef.

Creating a managed Chef account

Sign up for a hosted Chef account [here](#).

During the signup process, you will be asked to create a new organization.

The screenshot shows a 'Create Organization' dialog box. At the top left is a grid icon, followed by the text 'Create Organization'. At the top right is a close button (an 'X'). Below the title, there is a field labeled 'Full Name (example: Chef, Inc.)' containing the value 'Azure'. Below this is a field labeled 'Short Name (example: chef)' which is currently empty. A red error message 'Short name is required' is displayed below the empty field. At the bottom right of the dialog are two buttons: 'Cancel' and 'Create Organization'.

Once your organization is created, download the starter kit.

The screenshot shows the Chef Manage dashboard. The top navigation bar includes 'Nodes', 'Reports', 'Policy', and 'Administration' tabs, with 'Administration' being the active tab. On the left sidebar, under the 'Organizations' heading, there are links for 'Create', 'Reset Validation Key', 'Generate Knife Config', 'Invite User', 'Leave Organization', and 'Starter Kit'. Under 'Users', there is a link for 'Chef Documentation'. Under 'Groups', there is a link for 'Browse Community Cookbooks'. A note at the bottom of the sidebar states: 'NOTE If you receive a prompt warning you that your keys will be reset, it's ok to proceed as we have no existing infrastructure configured as yet.' The main content area features a large 'Thank you for choosing hosted Chef!' message, followed by 'Follow these steps to be on your way to using hosted Chef' and two buttons: 'Download Starter Kit' and 'Learn Chef'.

This starter kit zip file contains your organization config files and keys.

Configuring the Chef workstation

Extract the content of the chef-starter.zip to C:\chef.

Copy all files under chef-starter\chef-repo.chef to your c:\chef directory.

Your directory should now look something like the following example.

Name	Date modified	Type
chef-starter	11/12/2014 11:29 A...	File folder
cookbooks	11/12/2014 7:12 AM	File folder
a[REDACTED].validator.pem	11/12/2014 11:29 A...	PEM File
diego.publishsettings	11/12/2014 7:46 PM	PUBLISHSETTINGS F...
[REDACTED].pem	11/12/2014 11:29 A...	PEM File
knife.rb	11/12/2014 7:54 PM	RB File

You should now have four files including the Azure publishing file in the root of c:\chef.

The PEM files contain your organization and admin private keys for communication while the knife.rb file contains your knife configuration. We will need to edit the knife.rb file.

Open the file in your editor of choice and modify the "cookbook_path" by removing the ../../ from the path so it appears as shown next.

```
cookbook_path ["#{current_dir}/cookbooks"]
```

Also add the following line reflecting the name of your Azure publish settings file.

```
knife[:azure_publish_settings_file] = "yourfilename.publishsettings"
```

Your knife.rb file should now look similar to the following example.

```
current_dir = File.dirname(__FILE__)
log_level :info
log_location STDOUT
node_name '[REDACTED]'
client_key "#{current_dir}/[REDACTED].pem"
validation_client_name "a[REDACTED]-validator"
validation_key "#{current_dir}/a[REDACTED]-validator.pem"
chef_server_url "https://api.opscode.com/organizations/a[REDACTED]"
cache_type 'BasicFile'
cache_options( :path => "#{ENV['HOME']}/.chef/checksums" )
cookbook_path ["#{current_dir}/cookbooks"]
knife[:azure_publish_settings_file] = "[REDACTED].publishsettings"
```

These lines will ensure that Knife references the cookbooks directory under c:\chef\cookbooks, and also uses our Azure Publish Settings file during Azure operations.

Installing the Chef Development Kit

Next [download and install](#) the ChefDK (Chef Development Kit) to set up your Chef Workstation.



Custom Setup

Select the way you want features to be installed.



Click the icons in the tree below to change the way features will be installed.

Chef Development Kit

Compiling cost for this feature...

Location: C:\opscode\

[Browse...](#)

[Reset](#)

[Disk Usage](#)

[Back](#)

[Next](#)

[Cancel](#)

Install in the default location of c:\opscode. This install will take around 10 minutes.

Confirm your PATH variable contains entries for

C:\opscode\chefdk\bin;C:\opscode\chefdk\embedded\bin;c:\users\yourusername.chefdk\gem\ruby\2.0.0\bin

If they are not there, make sure you add these paths!

NOTE THE ORDER OF THE PATH IS IMPORTANT! If your opscode paths are not in the correct order you will have issues.

Reboot your workstation before you continue.

Next, we will install the Knife Azure extension. This provides Knife with the "Azure Plugin".

Run the following command.

```
chef gem install knife-azure --pre
```

NOTE

The --pre argument ensures you are receiving the latest RC version of the Knife Azure Plugin which provides access to the latest set of APIs.

It's likely that a number of dependencies will also be installed at the same time.

Command Prompt

```
c:\Chef>chef gem install knife-azure --pre

Temporarily enhancing PATH to include DevKit...
Building native extensions. This could take a while...
Successfully installed eventmachine-1.0.4
Successfully installed em-winrm-0.6.0
Successfully installed winrm-s-0.2.2
Successfully installed knife-windows-0.8.2
Fetching: knife-azure-1.4.0.rc.0.gem (100%)
Successfully installed knife-azure-1.4.0.rc.0
Parsing documentation for eventmachine-1.0.4
Installing ri documentation for eventmachine-1.0.4
Parsing documentation for em-winrm-0.6.0
Installing ri documentation for em-winrm-0.6.0
Parsing documentation for winrm-s-0.2.2
Installing ri documentation for winrm-s-0.2.2
Parsing documentation for knife-windows-0.8.2
Installing ri documentation for knife-windows-0.8.2
Parsing documentation for knife-azure-1.4.0.rc.0
Installing ri documentation for knife-azure-1.4.0.rc.0
Done installing documentation for eventmachine, em-winrm, winrm-s, knife-windows, knife-azure after 16 seconds
5 gems installed
```

To ensure everything is configured correctly, run the following command.

```
knife azure image list
```

If everything is configured correctly, you will see a list of available Azure images scroll through.

Congratulations. The workstation is set up!

Creating a Cookbook

A Cookbook is used by Chef to define a set of commands that you wish to execute on your managed client. Creating a Cookbook is straightforward and we use the **chef generate cookbook** command to generate the Cookbook template. I will be calling my Cookbook web server as I would like a policy that automatically deploys IIS.

Under your C:\Chef directory run the following command.

```
chef generate cookbook webserver
```

This will generate a set of files under the directory C:\Chef\cookbooks\webserver. We now need to define the set of commands we would like the Chef client to execute on the managed virtual machine.

The commands are stored in the file default.rb. In this file, I'll be defining a set of commands that installs IIS, starts IIS and copies a template file to the wwwroot folder.

Modify the C:\chef\cookbooks\webserver\recipes\default.rb file and add the following lines.

```
powershell_script 'Install IIS' do
  action :run
  code 'add-windowsfeature Web-Server'
end

service 'w3svc' do
  action [ :enable, :start ]
end

template 'c:\inetpub\wwwroot\Default.htm' do
  source 'Default.htm.erb'
  rights :read, 'Everyone'
end
```

Save the file once you are done.

Creating a template

As we mentioned previously, we need to generate a template file which will be used as the default.html page.

Run the following command to generate the template.

```
chef generate template webserver Default.htm
```

Now navigate to the C:\chef\cookbooks\webserver\templates\default\Default.htm.erb file. Edit the file by adding some simple "Hello World" HTML code, and then save the file.

Upload the Cookbook to the Chef Server

In this step, we are taking a copy of the Cookbook that we have created on the local machine and uploading it to the Chef Hosted Server. Once uploaded, the Cookbook will appear under the **Policy** tab.

```
knife cookbook upload webserver
```

Cookbook	Current Version
webserver	0.1.2

Deploy a virtual machine with Knife Azure

We will now deploy an Azure virtual machine and apply the "Webserver" Cookbook which will install the IIS web service and default web page.

In order to do this, use the **knife azure server create** command.

An example of the command appears next.

```
knife azure server create --azure-dns-name 'diegotest01' --azure-vm-name 'testserver01' --azure-vm-size 'Small' --azure-storage-account 'portalvhdsxxx' --bootstrap-protocol 'cloud-api' --azure-source-image 'a699494373c04fc0bc8f2bb1389d6106__Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd' --azure-service-location 'Southeast Asia' --winrm-user azureuser --winrm-password 'myPassword123' --tcp-endpoints 80,3389 --r 'recipe[webserver]'
```

The parameters are self-explanatory. Substitute your particular variables and run.

NOTE

Through the the command line, I'm also automating my endpoint network filter rules by using the -tcp-endpoints parameter. I've opened up ports 80 and 3389 to provide access to my web page and RDP session.

Once you run the command, go to the Azure portal and you will see your machine begin to provision.

```
testserver01      ** Starting          Visual Studio Ultimate with MSDN   Southeast Asia    diegotest01.cloudapp.net
```

The command prompt appears next.

```
c:\Chef>knife azure server create --azure-dns-name 'diegotest01' --azure-vm-name 'testserver01' --azure-vm-size 'Small' --azure-storage-account 'portals' --boot-strap-protocol 'cloud-api' --azure-source-image 'a699494373c04fc0bc8f2bb1389d6106' --azure-service-location 'Southeast Asia' --winrm-user azureuser --winrm-password 'myPassword123' --tcp-endpoints 80,3389 --r 'recipe[webserver]'

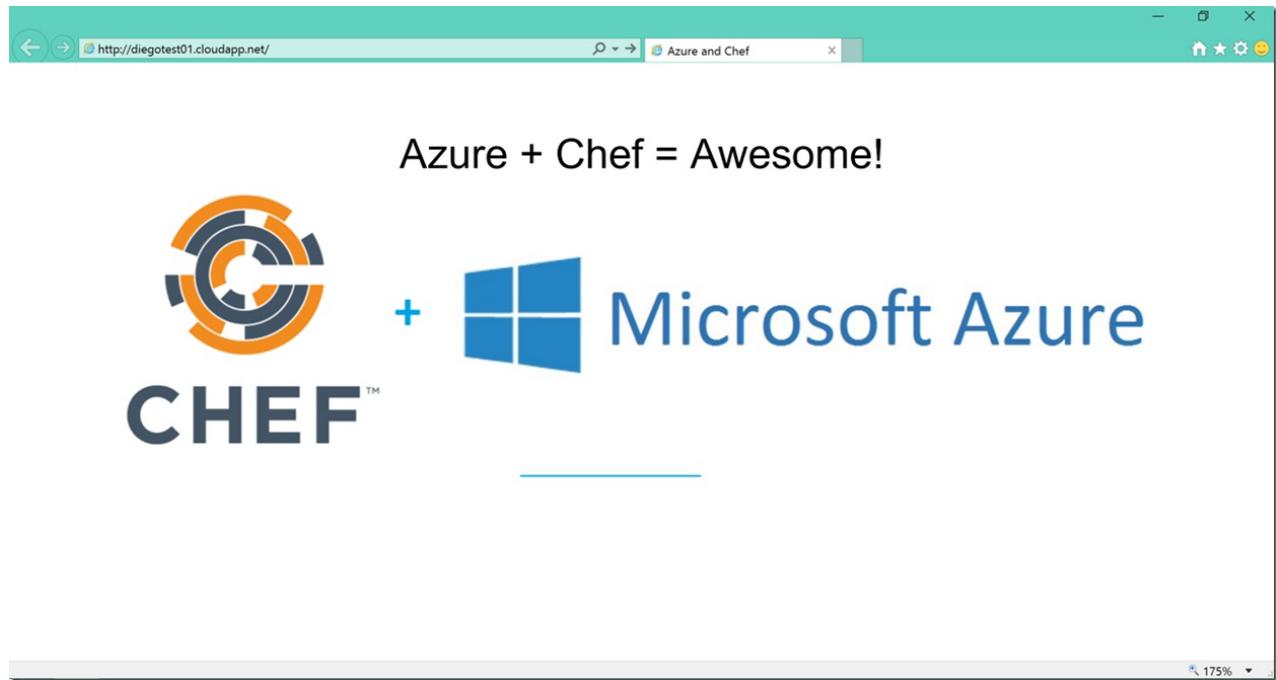
Waiting for virtual machine to reach status 'provisioning'.....vm state 'provisioning' reached after 2.79 minutes.
Waiting for virtual machine to reach status 'ready'.....vm state 'ready' reached after 2.23 minutes.

DNS Name: diegotest01.cloudapp.net
VM Name: testserver01
Size: Small
Azure Source Image: a699494373c04fc0bc8f2bb1389d6106__Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd
Azure Service Location: Southeast Asia
Public Ip Address: 104.43.9.88
Private Ip Address: 100.72.52.27
WinRM Port: 5985
TCP Ports: [{"Name": "tcpport_3389_testserver01", "Vip": "104.43.9.88", "PublicPort": "3389", "LocalPort": "3389"}, {"Name": "tcpport_80_testserver01", "Vip": "104.43.9.88", "PublicPort": "80", "LocalPort": "80"}]
Environment: _default
Runlist: ["recipe[webserver]"]

Waiting for Resource Extension to reach status 'wagent provisioning'....Resource extension state 'wagent provisioning' reached after 0.09 minutes.
Waiting for Resource Extension to reach status 'provisioning'.....Resource extension state 'provisioning' reached after 2.02 minutes.
Waiting for Resource Extension to reach status 'ready'.....Resource extension state 'ready' reached after 4.86 minutes.

DNS Name: diegotest01.cloudapp.net
VM Name: testserver01
Size: Small
Azure Source Image: a699494373c04fc0bc8f2bb1389d6106__Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd
Azure Service Location: Southeast Asia
Public Ip Address: 104.43.9.88
Private Ip Address: 100.72.52.27
WinRM Port: 5985
TCP Ports: [{"Name": "tcpport_3389_testserver01", "Vip": "104.43.9.88", "PublicPort": "3389", "LocalPort": "3389"}, {"Name": "tcpport_80_testserver01", "Vip": "104.43.9.88", "PublicPort": "80", "LocalPort": "80"}]
Environment: _default
Runlist: ["recipe[webserver]"]
```

Once the deployment is complete, we should be able to connect to the web service over port 80 as we had opened the port when we provisioned the virtual machine with the Knife Azure command. As this virtual machine is the only virtual machine in my cloud service, I'll connect it with the cloud service url.



As you can see, I got creative with my HTML code.

Don't forget we can also connect through an RDP session from the Azure portal via port 3389.

I hope this has been helpful! Go and start your infrastructure as code journey with Azure today!

Publish an ASP.NET Web App to an Azure VM from Visual Studio

12/7/2017 • 2 min to read • [Edit Online](#)

This document describes how to publish an ASP.NET web application to an Azure virtual machine (VM) using the **Microsoft Azure Virtual Machines** publishing feature in Visual Studio 2017.

Prerequisites

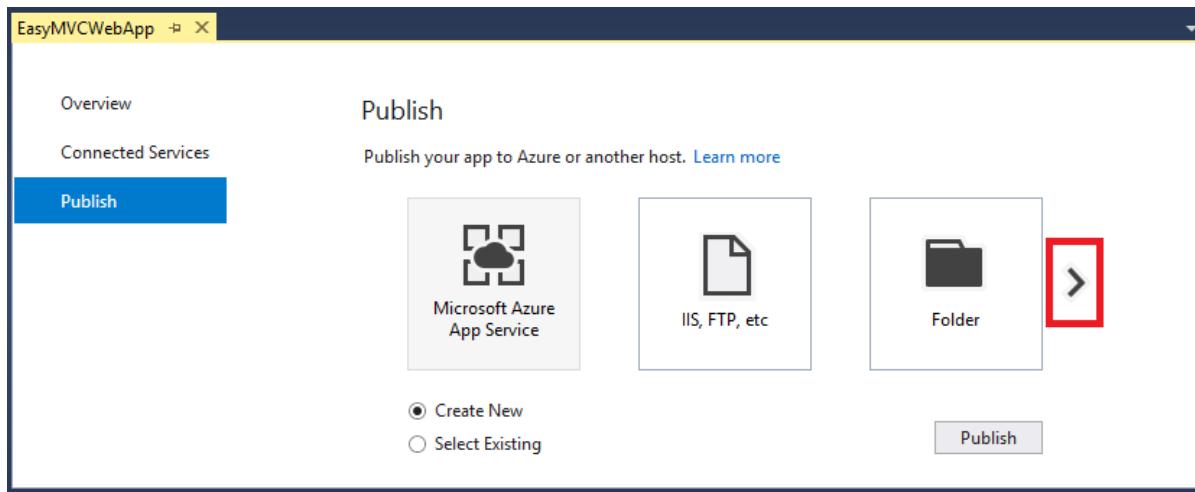
In order to use Visual Studio to publish an ASP.NET project to an Azure VM, the VM must be correctly set up.

- Machine must be configured to run an ASP.NET web application and have WebDeploy installed.
- The VM must have a DNS name configured. For more information, see [Create a fully qualified domain name in the Azure portal for a Windows VM](#).

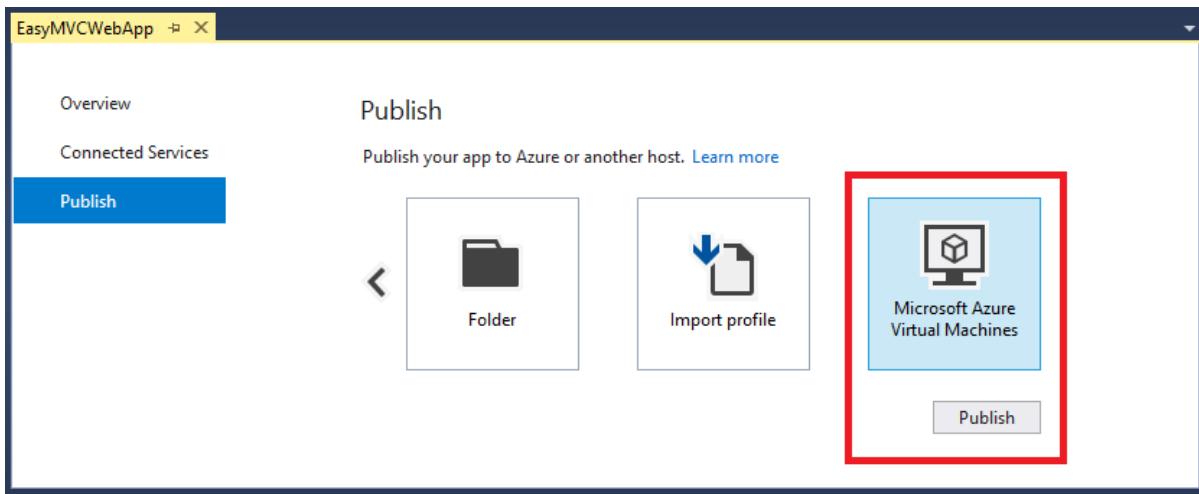
Publish your ASP.NET web app to the Azure VM using Visual Studio

The following section describes how to publish an existing ASP.NET web application to an Azure virtual machine.

1. Open your web app solution in Visual Studio 2017.
2. Right-click the project in Solution Explorer and choose **Publish...**
3. Use the arrow on the right of the page to scroll through the publishing options until you find **Microsoft Azure Virtual Machines**.

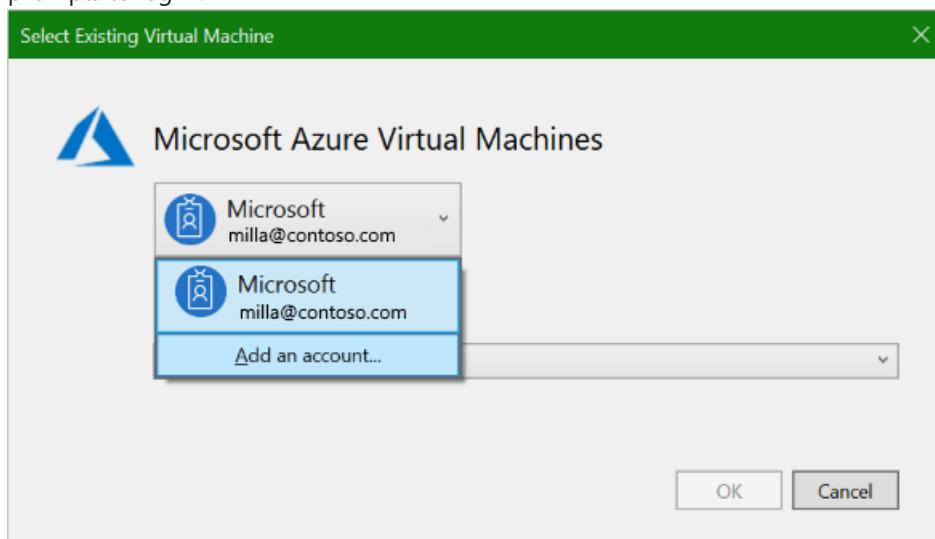


4. Select the **Microsoft Azure Virtual Machines** icon and select **Publish**.



5. Choose the appropriate account (with Azure subscription connected to your virtual machine).

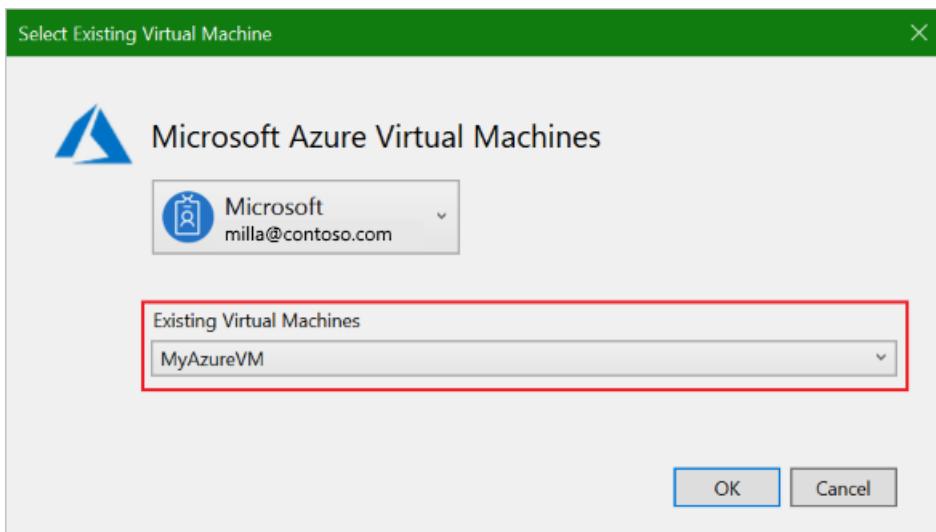
- If you're signed in to Visual Studio, the account list is populated with all your authenticated accounts.
- If you are not signed in, or if the account you need is not listed, choose "Add an account..." and follow the prompts to log in.



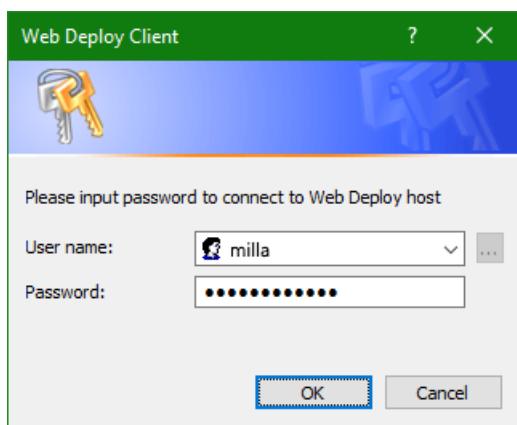
6. Select the appropriate VM from the list of Existing Virtual Machines.

NOTE

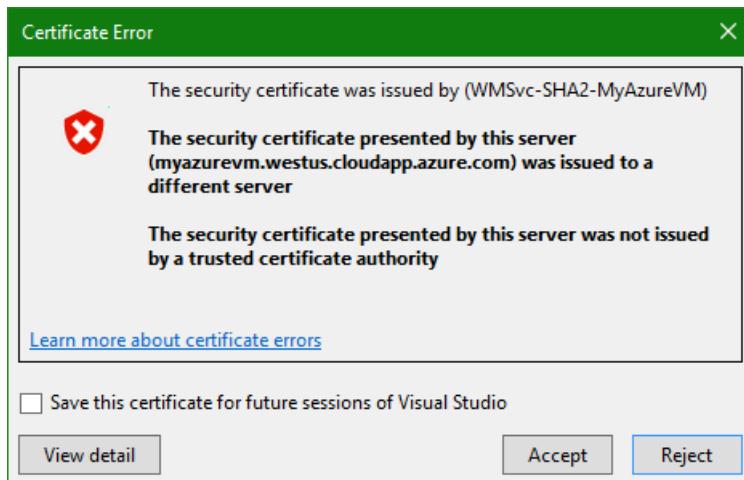
Populating this list can take some time.



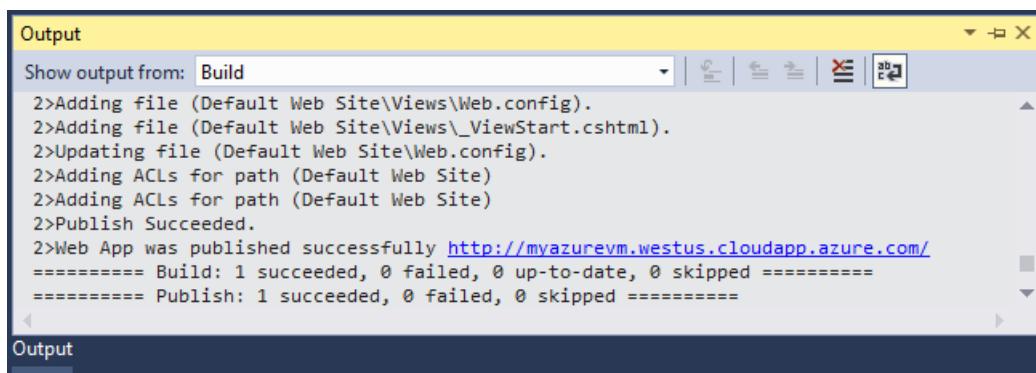
7. Click OK to begin publishing.
8. When prompted for credentials, supply the username and password of a user account on the target VM that is configured with publishing rights (typically the admin username and password used when creating the VM).



9. Accept the security certificate.



10. Watch the Output window to check the progress of the publish operation.



11. If publishing is successful, a browser launches to open the URL of the newly published site.

Success!

You have now successfully published your web app to an Azure virtual machine.

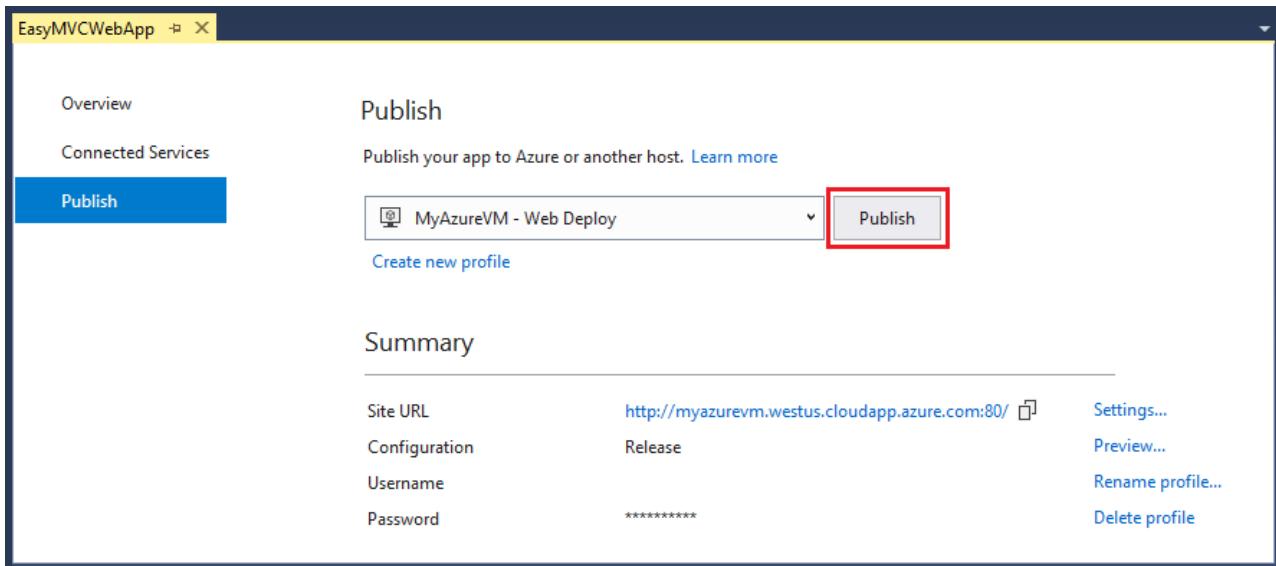
Publish Page Options

After completing the publish wizard, the Publish page is opened in the document well with the new publishing profile selected.

Re-publish

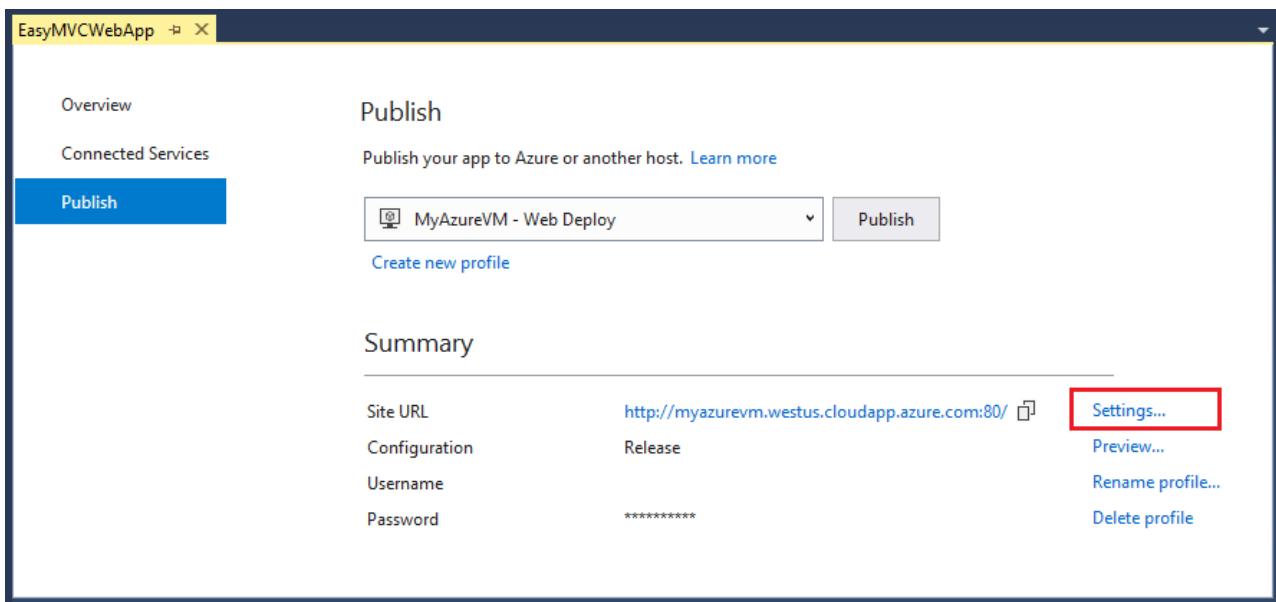
To publish updates to your web application, select the **Publish** button on the Publish page.

- If prompted, enter username and password.
- Publishing begins immediately.

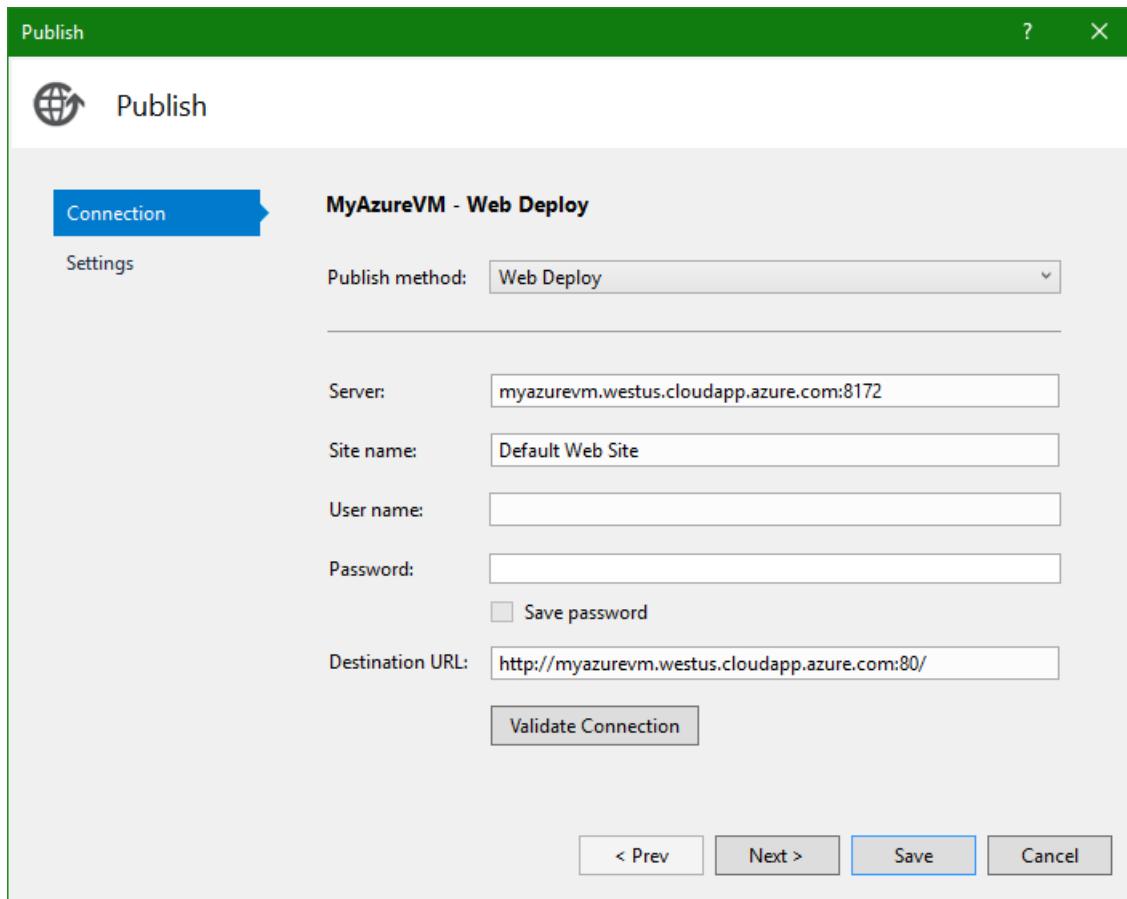


Modify publish profile settings

To view and modify the publish profile settings, select **Settings....**



Your settings should look something like this:

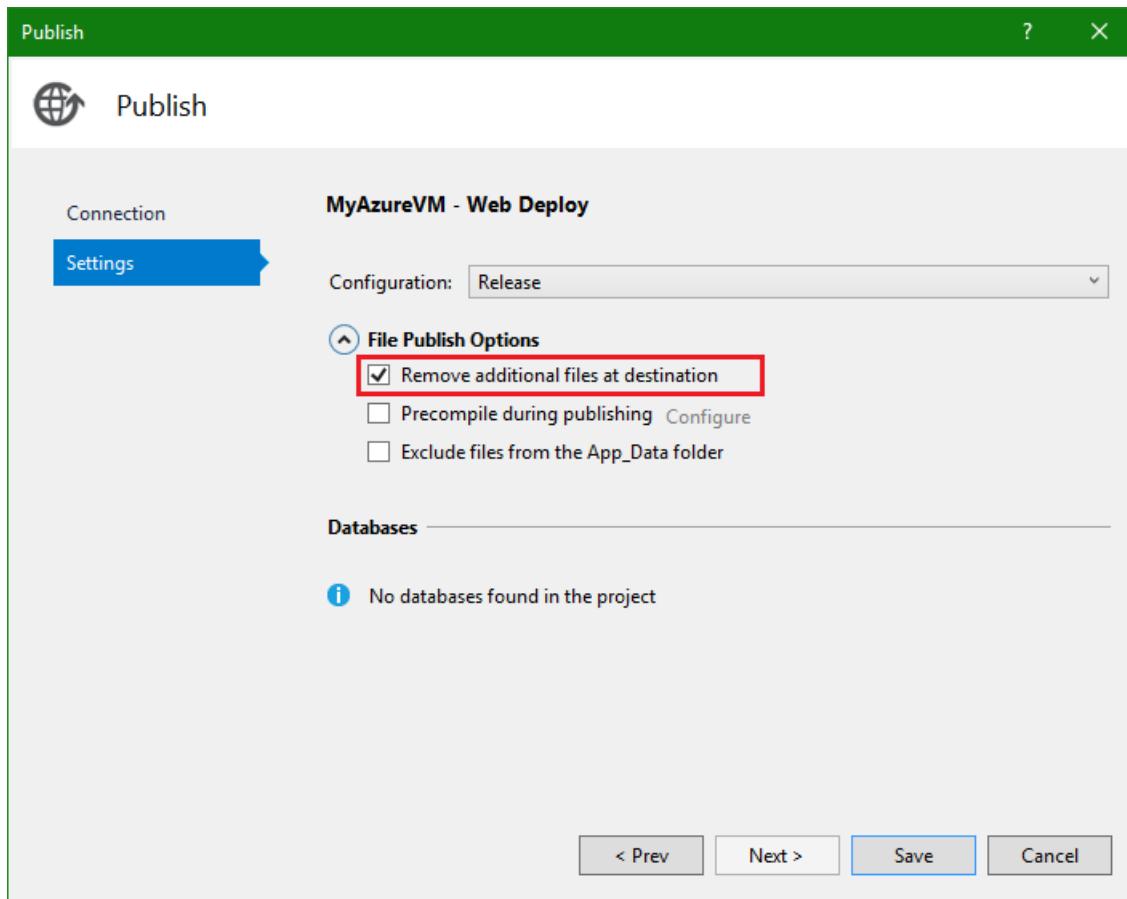


Save User name and Password

- To avoid providing authentication information every time you publish, you can populate the **User name** and **Password** fields and select the **Save password** box.
- Use the **Validate Connection** button to confirm that you have entered the right information.

Deploy to clean web server

- If you want to ensure that the web server has a clean copy of the web application after each upload (and that no other files are left hanging around from a previous deployment), you can check the **Remove additional files at destination** checkbox in the **Settings** tab.
- Warning: Publishing with this setting deletes all files that exist on the web server (wwwroot directory). Be sure you know the state of the machine before publishing with this option enabled.



Next steps

Set up CI/CD for automated deployment to Azure VM

To set up a continuous delivery pipeline with Visual Studio Team Service, see [Deploy to a Windows Virtual Machine](#).

What is SQL Server on Azure Virtual Machines? (Windows)

12/15/2017 • 4 min to read • [Edit Online](#)

SQL Server on Azure virtual machines enables you to use full versions of SQL Server in the Cloud without having to manage any on-premises hardware. SQL Server VMs also simplify licensing costs when you pay as you go.

Azure virtual machines run in many different [geographic regions](#) around the world. They also offer a variety of [machine sizes](#). The virtual machine image gallery allows you to create a SQL Server VM with the right version, edition, and operating system. This makes virtual machines a good option for a many different SQL Server workloads.

Automated updates

SQL Server Azure VMs can use [Automated Patching](#) to schedule a maintenance window for installing important windows and SQL Server updates automatically.

Automated backups

SQL Server Azure VMs can take advantage of [Automated Backup](#), which regularly creates backups of your database to blob storage. You can also manually use this technique. For more information, see [Use Azure Storage for SQL Server Backup and Restore](#).

High availability

If you require high availability, consider configuring SQL Server Availability Groups. This involves multiple SQL Server Azure VMs in a virtual network. You can configure your high availability solution manually, or you can use templates in the Azure portal for automatic configuration. For an overview of all high availability options, see [High Availability and Disaster Recovery for SQL Server in Azure Virtual Machines](#).

Performance

Azure virtual machines offer different machine sizes to meet various workload demands. SQL VMs also provide automated storage configuration, which is optimized for your performance requirements. For more information about configuring storage for SQL VMs, see [Storage configuration for SQL Server VMs](#). To fine-tune performance, see the [Performance best practices for SQL Server in Azure Virtual Machines](#).

Get started with SQL VMs

To get started, choose a SQL Server virtual machine image with your required version, edition, and operating system. The following sections provide direct links to the Azure portal for the SQL Server virtual machine gallery images.

TIP

To understand the VM and SQL pricing for these images, see [Pricing guidance for SQL Server Azure VMs](#).

TIP

To understand the update and lifecycle policy for SQL Server virtual machine gallery images, see the [SQL Server VMs FAQ](#).

Pay as you go

The following table provides a matrix of pay-as-you-go SQL Server images.

VERSION	OPERATING SYSTEM	EDITION
SQL Server 2017	Windows Server 2016	Enterprise, Standard, Web, Express, Developer
SQL Server 2016 SP1	Windows Server 2016	Enterprise, Standard, Web, Express, Developer
SQL Server 2014 SP2	Windows Server 2012 R2	Enterprise, Standard, Web, Express
SQL Server 2012 SP3	Windows Server 2012 R2	Enterprise, Standard, Web, Express
SQL Server 2008 R2 SP3	Windows Server 2008 R2	Enterprise, Standard, Web, Express

To see the available Linux SQL Server virtual machine images, see [Overview of SQL Server on Azure Virtual Machines \(Linux\)](#).

Bring your own license

You can also bring your own license (BYOL). In this scenario, you only pay for the VM without any additional charges for SQL Server licensing. Bringing your own license can save you money over time for continuous production workloads. For requirements to use this option, see [Pricing guidance for SQL Server Azure VMs](#).

VERSION	OPERATING SYSTEM	EDITION
SQL Server 2017	Windows Server 2016	Enterprise BYOL, Standard BYOL
SQL Server 2016 SP1	Windows Server 2016	Enterprise BYOL, Standard BYOL
SQL Server 2014 SP2	Windows Server 2012 R2	Enterprise BYOL, Standard BYOL
SQL Server 2012 SP2	Windows Server 2012 R2	Enterprise BYOL, Standard BYOL

In the portal, these image names are prefixed with **{BYOL}**.

Connect to the VM

After creating your SQL Server VM, connect to it from applications or tools, such as SQL Server Management Studio (SSMS). For instructions, see [Connect to a SQL Server Virtual Machine on Azure](#).

Migrate your data

If you have an existing database, you'll want to move that to the newly provisioned SQL VM. For a list of migration options and guidance, see [Migrating a Database to SQL Server on an Azure VM](#).

Customer experience improvement program (CEIP)

The Customer Experience Improvement Program (CEIP) is enabled by default. This periodically sends reports to Microsoft to help improve SQL Server. There is no management task required with CEIP unless you want to disable

it after provisioning. You can customize or disable the CEIP by connecting to the VM with remote desktop. Then run the **SQL Server Error and Usage Reporting** utility. Follow the instructions to disable reporting. For more information about data collection, see the [SQL Server Privacy Statement](#).

Related products and services

Windows Virtual Machines

- [Virtual Machines overview](#)

Storage

- [Introduction to Microsoft Azure Storage](#)

Networking

- [Virtual Network overview](#)
- [IP addresses in Azure](#)
- [Create a Fully Qualified Domain Name in the Azure portal](#)

SQL

- [SQL Server documentation](#)
- [Azure SQL Database comparison](#)

Next steps

Get started with SQL Server on Azure virtual machines:

- [Create a SQL Server VM in the Azure portal](#)

Get answers to commonly asked questions about SQL VMs:

- [SQL Server on Azure Virtual Machines FAQ](#)

Install and configure MongoDB on a Windows VM in Azure

12/15/2017 • 5 min to read • [Edit Online](#)

MongoDB is a popular open-source, high-performance NoSQL database. This article guides you through installing and configuring MongoDB on a Windows Server 2016 virtual machine (VM) in Azure. You can also [install MongoDB on a Linux VM in Azure](#).

Prerequisites

Before you install and configure MongoDB, you need to create a VM and, ideally, add a data disk to it. See the following articles to create a VM and add a data disk:

- Create a Windows Server VM using [the Azure portal](#) or [Azure PowerShell](#).
- Attach a data disk to a Windows Server VM using [the Azure portal](#) or [Azure PowerShell](#).

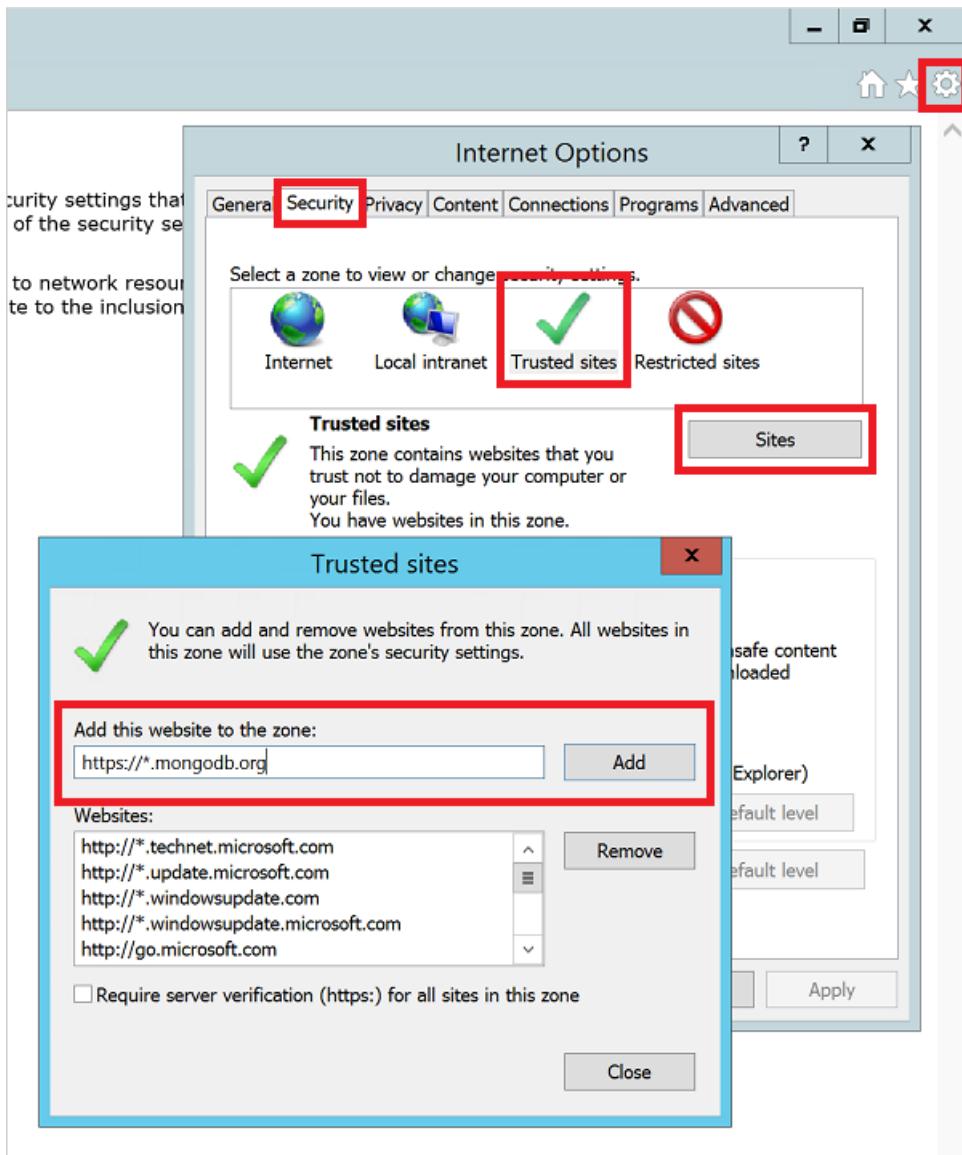
To begin installing and configuring MongoDB, [log on to your Windows Server VM](#) by using Remote Desktop.

Install MongoDB

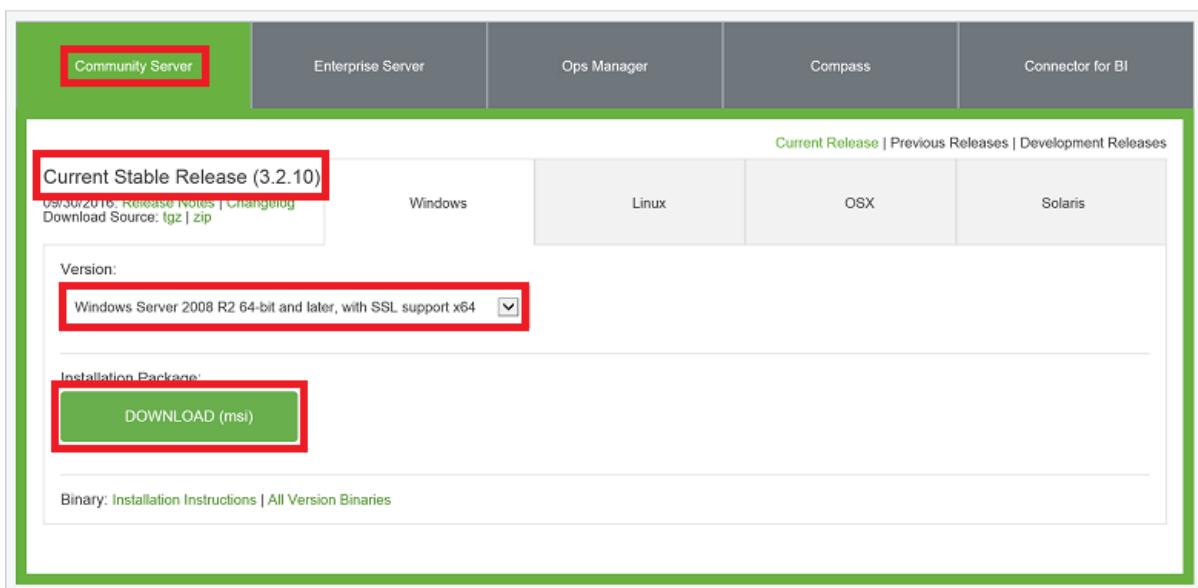
IMPORTANT

MongoDB security features, such as authentication and IP address binding, are not enabled by default. Security features should be enabled before deploying MongoDB to a production environment. For more information, see [MongoDB Security and Authentication](#).

1. After you've connected to your VM using Remote Desktop, open Internet Explorer from the taskbar.
2. Select **Use recommended security, privacy, and compatibility settings** when Internet Explorer first opens, and click **OK**.
3. Internet Explorer enhanced security configuration is enabled by default. Add the MongoDB website to the list of allowed sites:
 - Select the **Tools** icon in the upper-right corner.
 - In **Internet Options**, select the **Security** tab, and then select the **Trusted Sites** icon.
 - Click the **Sites** button. Add https://*.mongodb.com to the list of trusted sites, and then close the dialog box.



4. Browse to the [MongoDB - Downloads](http://www.mongodb.com/downloads) page (<http://www.mongodb.com/downloads>).
5. If needed, select the **Community Server** edition and then select the latest current stable release for *Windows Server 2008 R2 64-bit and later*. To download the installer, click **DOWNLOAD (msi)**.



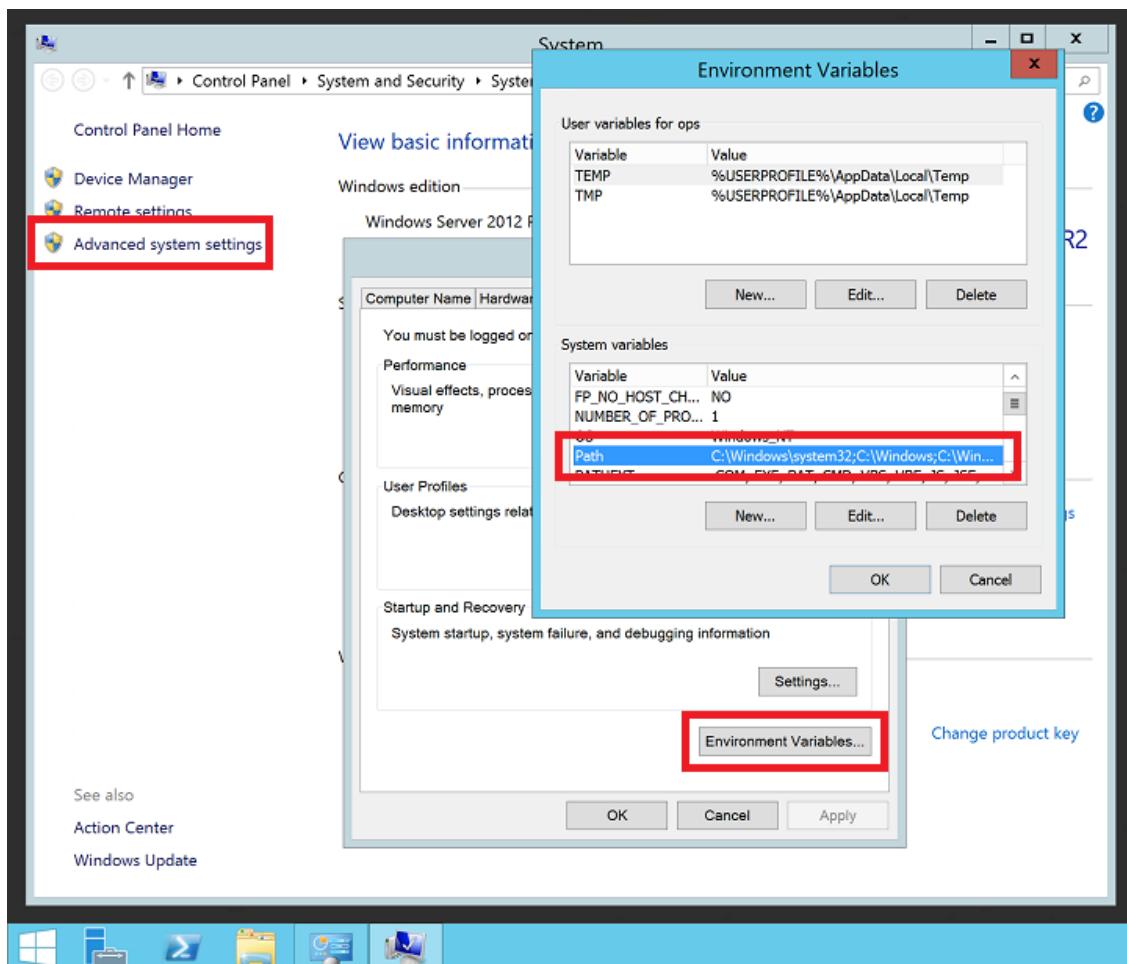
Run the installer after the download is complete.

6. Read and accept the license agreement. When you're prompted, select **Complete** install.

7. If desired, you can choose to also install Compass, a graphical interface for MongoDB.
 8. On the final screen, click **Install**.

Configure the VM and MongoDB

1. The path variables are not updated by the MongoDB installer. Without the MongoDB `bin` location in your path variable, you need to specify the full path each time you use a MongoDB executable. To add the location to your path variable:
 - Right-click the **Start** menu, and select **System**.
 - Click **Advanced system settings**, and then click **Environment Variables**.
 - Under **System variables**, select **Path**, and then click **Edit**.



Add the path to your MongoDB `bin` folder. MongoDB is typically installed in `C:\Program Files\MongoDB`. Verify the installation path on your VM. The following example adds the default MongoDB install location to the `PATH` variable:

`;C:\Program Files\MongoDB\Server\3.6\bin`

NOTE

Be sure to add the leading semicolon (;) to indicate that you are adding a location to your PATH variable.

2. Create MongoDB data and log directories on your data disk. From the **Start** menu, select **Command Prompt**. The following examples create the directories on drive F:

```
mkdir F:\MongoData  
mkdir F:\MongoLogs
```

3. Start a MongoDB instance with the following command, adjusting the path to your data and log directories accordingly:

```
mongod --dbpath F:\MongoData\ --logpath F:\MongoLogs\mongolog.log
```

It may take several minutes for MongoDB to allocate the journal files and start listening for connections. All log messages are directed to the *F:\MongoLogs\mongolog.log* file as `mongod.exe` server starts and allocates journal files.

NOTE

The command prompt stays focused on this task while your MongoDB instance is running. Leave the command prompt window open to continue running MongoDB. Or, install MongoDB as service, as detailed in the next step.

4. For a more robust MongoDB experience, install the `mongod.exe` as a service. Creating a service means you don't need to leave a command prompt running each time you want to use MongoDB. Create the service as follows, adjusting the path to your data and log directories accordingly:

```
mongod --dbpath F:\MongoData\ --logpath F:\MongoLogs\mongolog.log --logappend --install
```

The preceding command creates a service named MongoDB, with a description of "Mongo DB". The following parameters are also specified:

- The `--dbpath` option specifies the location of the data directory.
- The `--logpath` option must be used to specify a log file, because the running service does not have a command window to display output.
- The `--logappend` option specifies that a restart of the service causes output to append to the existing log file.

To start the MongoDB service, run the following command:

```
net start MongoDB
```

For more information about creating the MongoDB service, see [Configure a Windows Service for MongoDB](#).

Test the MongoDB instance

With MongoDB running as a single instance or installed as a service, you can now start creating and using your databases. To start the MongoDB administrative shell, open another command prompt window from the **Start** menu, and enter the following command:

```
mongo
```

You can list the databases with the `db` command. Insert some data as follows:

```
db.foo.insert( { a : 1 } )
```

Search for data as follows:

```
db.foo.find()
```

The output is similar to the following example:

```
{ "_id" : "ObjectId("57f6a86cee873a6232d74842")", "a" : 1 }
```

Exit the `mongo` console as follows:

```
exit
```

Configure firewall and Network Security Group rules

Now that MongoDB is installed and running, open a port in Windows Firewall so you can remotely connect to MongoDB. To create a new inbound rule to allow TCP port 27017, open an administrative PowerShell prompt and enter the following command:

```
New-NetFirewallRule `  
-DisplayName "Allow MongoDB" `  
-Direction Inbound `  
-Protocol TCP `  
-LocalPort 27017 `  
-Action Allow
```

You can also create the rule by using the **Windows Firewall with Advanced Security** graphical management tool. Create a new inbound rule to allow TCP port 27017.

If needed, create a Network Security Group rule to allow access to MongoDB from outside of the existing Azure virtual network subnet. You can create the Network Security Group rules by using the [Azure portal](#) or [Azure PowerShell](#). As with the Windows Firewall rules, allow TCP port 27017 to the virtual network interface of your MongoDB VM.

NOTE

TCP port 27017 is the default port used by MongoDB. You can change this port by using the `--port` parameter when starting `mongod.exe` manually or from a service. If you change the port, make sure to update the Windows Firewall and Network Security Group rules in the preceding steps.

Next steps

In this tutorial, you learned how to install and configure MongoDB on your Windows VM. You can now access MongoDB on your Windows VM, by following the advanced topics in the [MongoDB documentation](#).

Using Azure for hosting and running SAP workload scenarios

11/1/2017 • 10 min to read • [Edit Online](#)

By choosing Microsoft Azure as your SAP ready cloud partner, you are able to reliably run your mission critical SAP workloads and scenarios on a scalable, compliant, and enterprise-proven platform. Get the scalability, flexibility, and cost savings of Azure. With the expanded partnership between Microsoft and SAP, you can run SAP applications across dev/test and production scenarios in Azure - and be fully supported. From SAP NetWeaver to SAP S4/HANA, SAP BI, Linux to Windows, SAP HANA to SQL, we have you covered.

Besides hosting SAP NetWeaver scenarios with the different DBMS on Azure, you can host different other SAP workload scenarios, like SAP BI on Azure. Documentation regarding SAP NetWeaver deployments on Azure native Virtual Machines can be found in the section "SAP NetWeaver on Azure Virtual Machines."

Azure has native Azure Virtual Machine offers that are ever growing in size of CPU and memory resources to cover SAP workload that leverages SAP HANA. For more information on this topic, look up the documents under the section SAP HANA on Azure Virtual Machines."

The uniqueness of Azure for SAP HANA is a unique offer that sets Azure apart from competition. In order to enable hosting more memory and CPU resource demanding SAP scenarios involving SAP HANA, Azure offers the usage of customer dedicated bare-metal hardware for the purpose of running SAP HANA deployments that require up to 20 TB (60 TB scale-out) of memory for S/4HANA or other SAP HANA workload. This unique Azure solution of SAP HANA on Azure (Large Instances) allows you to run SAP HANA on the dedicated bare-metal hardware with the SAP application layer or workload middle-ware layer hosted in native Azure Virtual Machines. This solution is documented in several documents in the section "SAP HANA on Azure (Large Instances)."

Hosting SAP workload scenarios in Azure also can create requirements of Identity integration and Single-Sign-On using Azure Activity Directory to different SAP components and SAP SaaS or PaaS offers. A list of such integration and Single-Sign-On scenarios with Azure Active Directory (AAD) and SAP entities is described and documented in the section "AAD SAP Identity Integration and Single-Sign-On."

SAP HANA on SAP HANA on Azure (Large Instances)

Overview and architecture of SAP HANA on Azure (Large Instances)

Title: Overview and Architecture of SAP HANA on Azure (Large Instances)

Summary: This Architecture and Technical Deployment Guide provides information to help you deploy SAP on the new SAP HANA on Azure (Large Instances) in Azure. It is not intended to be a comprehensive guide covering specific setup of SAP solutions, but rather useful information in your initial deployment and ongoing operations. It should not replace SAP documentation related to the installation of SAP HANA (or the many SAP Support Notes that cover the topic). It gives you an overview and provides the additional detail of installing SAP HANA on Azure (Large Instances).

Updated: October 2017

[This guide can be found here](#)

Infrastructure and connectivity to SAP HANA on Azure (Large Instances)

Title: Infrastructure and Connectivity to SAP HANA on Azure (Large Instances)

Summary: After the purchase of SAP HANA on Azure (Large Instances) is finalized between you and the Microsoft

enterprise account team, various network configurations are required in order to ensure proper connectivity. This document outlines the information that has to be shared with the following information is required. This document outlines what information has to be collected and what configuration scripts have to be run.

Updated: October 2017

[This guide can be found here](#)

Install SAP HANA in SAP HANA on Azure (Large Instances)

Title: Install SAP HANA on SAP HANA on Azure (Large Instances)

Summary: This document outlines the setup procedures for installing SAP HANA on your Azure Large Instance.

Updated: July 2017

[This guide can be found here](#)

High availability and disaster recovery of SAP HANA on Azure (Large Instances)

Title: High Availability and Disaster Recovery of SAP HANA on Azure (Large Instances)

Summary: High Availability (HA) and Disaster Recovery (DR) are very important aspects of running your mission-critical SAP HANA on Azure (Large Instances) server(s). It's import to work with SAP, your system integrator, and/or Microsoft to properly architect and implement the right HA/DR strategy for you. Important considerations like Recovery Point Objective (RPO) and Recovery Time Objective (RTO), specific to your environment, must be considered. This document explains your options for enabling your preferred level of HA and DR.

Updated: October 2017

[This document can be found here](#)

Troubleshooting and monitoring of SAP HANA on Azure (Large Instances)

Title: Troubleshooting and Monitoring of SAP HANA on Azure (Large Instances)

Summary: This guide covers information that is useful in establishing monitoring of your SAP HANA on Azure environment as well as additional troubleshooting information.

Updated: October 2017

[This document can be found here](#)

SAP HANA on Azure Virtual Machines

Getting started with SAP HANA on Azure

Title: Quickstart guide for manual installation of SAP HANA on Azure VMs

Summary: This quickstart guide helps to set up a single-instance SAP HANA system on Azure VMs by a manual installation of SAP NetWeaver 7.5 and SAP HANA SP12. The guide presumes that the reader is familiar with Azure IaaS basics like how to deploy virtual machines or virtual networks either via the Azure portal or Powershell/CLI including the option to use json templates. Furthermore it's expected that the reader is familiar with SAP HANA, SAP NetWeaver and how to install it on-premises.

Updated: June 2017

[This guide can be found here](#)

S/4HANA SAP CAL deployment on Azure

Title: Deploy SAP S/4HANA or BW/4HANA on Azure

Summary: This guide helps to demonstrate the deployment of SAP S/4HANA on Azure using SAP Cloud Appliance

Library. SAP Cloud Appliance Library is a service by SAP that allows to deploy SAP applications on Azure. The guide describes step by step the deployment.

Updated: June 2017

[This guide can be found here](#)

High Availability of SAP HANA in Azure Virtual Machines

Title: High Availability of SAP HANA on Azure Virtual Machines

Summary: This guide leads you through the high availability configuration of the SUSE 12 OS and SAP HANA to accommodate HANA System replication with automatic failover. The guide is specific for SUSE and Azure Virtual Machines. The guide does not apply yet for Red Hat or bare-metal or private cloud or other non-Azure public cloud deployments.

Updated: June 2017

[This guide can be found here](#)

SAP HANA backup overview on Azure VMs

Title: Backup guide for SAP HANA on Azure Virtual Machines

Summary: This guide provides basic information about backup possibilities running SAP HANA on Azure Virtual Machines.

Updated: March 2017

[This guide can be found here](#)

SAP HANA file level backup on Azure VMs

Title: SAP HANA backup based on storage snapshots

Summary: This guide provides information about using snapshot-based backups on Azure VMs when running SAP HANA on Azure Virtual Machines.

Updated: March 2017

[This guide can be found here](#)

SAP HANA snapshot based backups on Azure VMs

Title: SAP HANA Azure Backup on file level

Summary: This guide provides information about using SAP HANA file level backup running SAP HANA on Azure Virtual Machines

Updated: March 2017

[This guide can be found here](#)

SAP NetWeaver deployed on Azure Virtual Machines

Deploy SAP IDES system on Windows and SQL Server through SAP CAL on Azure

Title: Testing SAP NetWeaver on Microsoft Azure SUSE Linux VMs

Summary: This document describes the deployment of an SAP IDES system based on Windows and SQL Server on Azure using SAP Cloud Appliance Library. SAP Cloud appliance Library is an SAP service that allows the deployment of SAP products on Azure. This document goes step by step through the deployment of an SAP IDES system. The IDES system is just an example for several other dozen applications that can be deployed through SAP Cloud appliance on Microsoft Azure.

Updated: June 2017

[This guide can be found here](#)

Quickstart guide for NetWeaver on SUSE Linux on Azure

Title: Testing SAP NetWeaver on Microsoft Azure SUSE Linux VMs

Summary: This article describes various things to consider when you're running SAP NetWeaver on Microsoft Azure SUSE Linux virtual machines (VMs). SAP NetWeaver is officially supported on SUSE Linux VMs on Azure. All details regarding Linux versions, SAP kernel versions, and other details can be found in SAP Note 1928533 "SAP Applications on Azure: Supported Products and Azure VM types".

Updated: September 2016

[This guide can be found here](#)

Planning and implementation

Title: Azure Virtual Machines planning and implementation for SAP NetWeaver

Summary: This document is the guide to start with if you are thinking about running SAP NetWeaver in Azure Virtual Machines. This planning and implementation guide helps you evaluate whether an existing or planned SAP NetWeaver-based system can be deployed to an Azure Virtual Machines environment. It covers multiple SAP NetWeaver deployment scenarios, and includes SAP configurations that are specific to Azure. The paper lists and describes all the necessary configuration information you'll need on the SAP/Azure side to run a hybrid SAP landscape. Measures you can take to ensure high availability of SAP NetWeaver-based systems on IaaS are also covered.

Updated: June 2017

[This guide can be found here](#)

High Availability configurations of SAP NetWeaver in Azure VMs

Title: Azure Virtual Machines High Availability for SAP NetWeaver

Summary: In this document, we cover the steps that you can take to deploy high-availability SAP systems in Azure by using the Azure Resource Manager deployment model. We walk you through these major tasks. In the document, we describe how single-point-of-failure components like Advanced Business Application Programming (ABAP) SAP Central Services (ASCS)/SAP Central Services (SCS) and database management systems (DBMS), and redundant components like SAP Application Server are going to be protected when running in Azure VMs. A step-by-step example of an installation and configuration of a high-availability SAP system in a Windows Server Failover Clustering cluster and SUSE Linux Enterprise Server Cluster Framework in Azure is demonstrated and shown in this document.

Updated: October 2017

[This guide can be found here](#)

Realizing Multi-SID deployments of SAP NetWeaver in Azure VMs

Title: Create an SAP NetWeaver multi-SID configuration

Summary: This document is an addition to the document High availability for SAP NetWeaver on Azure VMs. Due to new functionality in Azure that got introduced in September 2016, it is possible to deploy multiple SAP NetWeaver ASCS/SCS instances in a pair of Azure VMs. With such a configuration, you can reduce the number of VMs necessary to deploy to realize highly available SAP NetWeaver configurations. The guide describes the setup of such multi-SID configurations.

Updated: December 2016

[This guide can be found here](#)

Deployment of SAP NetWeaver in Azure VMs

Title: Azure Virtual Machines deployment for SAP NetWeaver

Summary: This document provides procedural guidance for deploying SAP NetWeaver software to virtual machines in Azure. This paper focuses on three specific deployment scenarios, with an emphasis on enabling the Azure Monitoring Extensions for SAP, including troubleshooting recommendations for the Azure Monitoring Extensions for SAP. This paper assumes that you've read the planning and implementation guide.

Updated: June 2017

[This guide can be found here](#)

DBMS deployment guide

Title: Azure Virtual Machines DBMS deployment for SAP NetWeaver

Summary: This paper covers planning and implementation considerations for the DBMS systems that should run in conjunction with SAP. In the first part, general considerations are listed and presented. The following parts of the paper relate to deployments of different DBMS in Azure that are supported by SAP. Different DBMS presented are SQL Server, SAP ASE, and Oracle. In those specific parts, considerations you have to account for when you are running SAP systems on Azure in conjunction with those DBMS are discussed. Subjects like backup and high availability methods that are supported by the different DBMS on Azure are presented for the usage with SAP applications.

Updated: June 2017

[This guide can be found here](#)

Using Azure Site Recovery for SAP workload

Title: SAP NetWeaver: Building a Disaster Recovery Solution with Azure Site Recovery

Summary: This document describes the way how Azure Site Recovery services can be used for the purpose of handling disaster recovery scenarios. Cases where Azure is used as disaster recovery location for an on-premise SAP landscape using Azure Site Recovery Services. Another scenario described in the document is the Azure-to-Azure (A2A) disaster recovery case and how it is managed with Azure Site Recovery.

Updated: August 2017

[This guide can be found here](#)

Create MATLAB Distributed Computing Server clusters on Azure VMs

11/16/2017 • 3 min to read • [Edit Online](#)

Use Microsoft Azure virtual machines to create one or more MATLAB Distributed Computing Server clusters to run your compute-intensive parallel MATLAB workloads. Install your MATLAB Distributed Computing Server software on a VM to use as a base image and use an Azure quickstart template or Azure PowerShell script (available on [GitHub](#)) to deploy and manage the cluster. After deployment, connect to the cluster to run your workloads.

About MATLAB and MATLAB Distributed Computing Server

The [MATLAB](#) platform is optimized for solving engineering and scientific problems. MATLAB users with large-scale simulations and data processing tasks can use MathWorks parallel computing products to speed up their compute-intensive workloads by taking advantage of compute clusters and grid services. [Parallel Computing Toolbox](#) lets MATLAB users parallelize applications and take advantage of multi-core processors, GPUs, and compute clusters. [MATLAB Distributed Computing Server](#) enables MATLAB users to utilize many computers in a compute cluster.

By using Azure virtual machines, you can create MATLAB Distributed Computing Server clusters that have all the same mechanisms available to submit parallel work as on-premises clusters, such as interactive jobs, batch jobs, independent tasks, and communicating tasks. Using Azure in conjunction with the MATLAB platform has many benefits compared to provisioning and using traditional on-premises hardware: a range of virtual machine sizes, creation of clusters on-demand so you pay only for the compute resources you use, and the ability to test models at scale.

Prerequisites

- **Client computer** - You'll need a Windows-based client computer to communicate with Azure and the MATLAB Distributed Computing Server cluster after deployment.
- **Azure PowerShell** - See [How to install and configure Azure PowerShell](#) to install it on your client computer.
- **Azure subscription** - If you don't have a subscription, you can create a [free account](#) in just a couple of minutes. For larger clusters, consider a pay-as-you-go subscription or other purchase options.
- **vCPUs quota** - You might need to increase the vCPU quota to deploy a large cluster or more than one MATLAB Distributed Computing Server cluster. To increase a quota, [open an online customer support request](#) at no charge.
- **MATLAB, Parallel Computing Toolbox, and MATLAB Distributed Computing Server licenses** - The scripts assume that the [MathWorks Hosted License Manager](#) is used for all licenses.
- **MATLAB Distributed Computing Server software** - Will be installed on a VM that will be used as the base VM image for the cluster VMs.

High level steps

To use Azure virtual machines for your MATLAB Distributed Computing Server clusters, the following high-level steps are required. Detailed instructions are in the documentation accompanying the quickstart template and scripts on [GitHub](#).

1. Create a base VM image

- Download and install MATLAB Distributed Computing Server software onto this VM.

NOTE

This process can take a couple of hours, but you only have to do it once for each version of MATLAB you use.

2. Create one or more clusters

- Use the supplied PowerShell script or use the quickstart template to create a cluster from the base VM image.
- Manage the clusters using the supplied PowerShell script which allows you to list, pause, resume, and delete clusters.

Cluster configurations

Currently, the cluster creation script and template enable you to create a single MATLAB Distributed Computing Server topology. If you want, create one or more additional clusters, with each cluster having a different number of worker VMs, using different VM sizes, and so on.

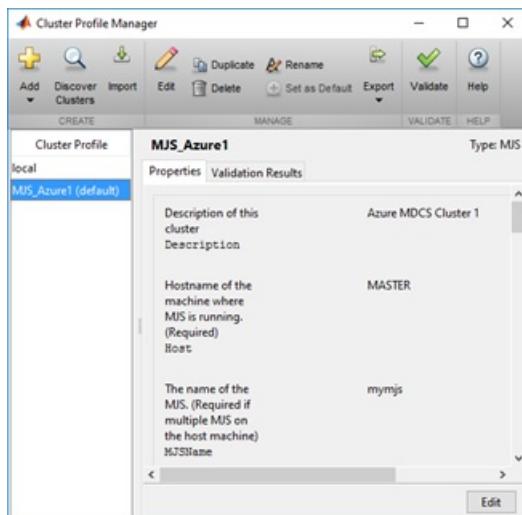
MATLAB client and cluster in Azure

The MATLAB client node, MATLAB Job Scheduler node, and MATLAB Distributed Computing Server "worker" nodes are all configured as Azure VMs in a virtual network, as shown in the following figure.

- To use the cluster, connect by Remote Desktop to the client node. The client node runs the MATLAB client.
- The client node has a file share that can be accessed by all workers.
- MathWorks Hosted License Manager is used for the license checks for all MATLAB software.
- By default, one MATLAB Distributed Computing Server worker per vCPU is created on the worker VMs, but you can specify any number.

Use an Azure-based Cluster

As with other types of MATLAB Distributed Computing Server clusters, you need to use the Cluster Profile Manager in the MATLAB client (on the client VM) to create a MATLAB Job Scheduler cluster profile.



Next steps

- For detailed instructions to deploy and manage MATLAB Distributed Computing Server clusters in Azure, see the [GitHub](#) repository containing the templates and scripts.
- Go to the [MathWorks site](#) for detailed documentation for MATLAB and MATLAB Distributed Computing Server.

Options with HPC Pack to create and manage a cluster for Windows HPC workloads in Azure

10/27/2017 • 1 min to read • [Edit Online](#)

Create and manage a cloud-based high-performance computing (HPC) cluster by taking advantage of [Microsoft HPC Pack](#) and Azure compute and infrastructure services. HPC Pack, available for free download, is built on Microsoft Azure and Windows Server technologies and supports a wide range of HPC workloads.

For more HPC options in Azure, see [Technical resources for batch and high-performance computing](#).

This article focuses on options to create HPC Pack clusters to run Windows workloads. There are also options for creating HPC Pack clusters to run [Linux HPC workloads](#).

HPC Pack cluster in Azure VMs and VM scale sets

Azure templates

- ([GitHub](#)) [HPC Pack 2016 cluster templates](#)
- ([GitHub](#)) [HPC Pack 2012 R2 cluster templates](#)
- ([Marketplace](#)) [HPC Pack cluster for Windows workloads](#)
- ([Marketplace](#)) [HPC Pack cluster for Excel workloads](#)
- ([Quickstart](#)) [Create an HPC cluster](#)
- ([Quickstart](#)) [Create an HPC cluster with custom compute node image](#)

Azure VM images

- [HPC Pack 2016 head node on Windows Server 2016](#)
- [HPC Pack 2016 compute node on Windows Server 2016](#)
- [HPC Pack 2016 head node on Windows Server 2012 R2](#)
- [HPC Pack 2016 compute node on Windows Server 2012 R2](#)
- [HPC Pack 2012 R2 head node on Windows Server 2012 R2](#)
- [HPC Pack 2012 R2 compute node on Windows Server 2012 R2](#)
- [HPC Pack compute node with Excel on Windows Server 2012 R2](#)

PowerShell deployment script for HPC Pack 2012 R2

- [Create an HPC cluster with the HPC Pack IaaS deployment script](#)

Tutorials

- [Tutorial: Deploy an HPC Pack 2016 cluster in Azure](#)
- [Tutorial: Get started with an HPC Pack cluster in Azure to run Excel and SOA workloads](#)

Manual deployment with the Azure portal

- [Set up the head node of an HPC Pack cluster in an Azure VM](#)

Cluster management

- [Manage compute nodes in an HPC Pack cluster in Azure](#)
- [Grow and shrink Azure compute resources in an HPC Pack cluster](#)
- [Submit jobs to an HPC Pack cluster in Azure](#)
- [Job management in HPC Pack](#)
- [Manage an HPC Pack 2016 cluster in Azure with Azure Active Directory](#)

Burst with worker role nodes

- [Burst to Azure worker instances with HPC Pack](#)
- [Tutorial: Set up a hybrid cluster with HPC Pack in Azure](#)
- [Add Azure "burst" nodes to an HPC Pack head node in Azure](#)

Burst with Azure Batch

- [Burst to Azure Batch with HPC Pack](#)

RDMA clusters for MPI workloads

- [Set up a Windows RDMA cluster with HPC Pack to run MPI applications](#)

Deploy an HPC Pack 2016 cluster in Azure

6/27/2017 • 4 min to read • [Edit Online](#)

Follow the steps in this article to deploy a [Microsoft HPC Pack 2016](#) cluster in Azure virtual machines. HPC Pack is Microsoft's free HPC solution built on Microsoft Azure and Windows Server technologies and supports a wide range of HPC workloads.

Use one of the [Azure Resource Manager templates](#) to deploy the HPC Pack 2016 cluster. You have several choices of cluster topology with different numbers of cluster head nodes, and with either Linux or Windows compute nodes.

Prerequisites

PFX certificate

A Microsoft HPC Pack 2016 cluster requires a Personal Information Exchange (PFX) certificate to secure the communication between the HPC nodes. The certificate must meet the following requirements:

- It must have a private key capable of key exchange
- Key usage includes Digital Signature and Key Encipherment
- Enhanced key usage includes Client Authentication and Server Authentication

If you don't already have a certificate that meets these requirements, you can request the certificate from a certification authority. Alternatively, you can use the following commands to generate the self-signed certificate based on the operating system on which you run the command, and export the PFX format certificate with private key.

- For Windows 10 or Windows Server 2016, run the built-in **New-SelfSignedCertificate** PowerShell cmdlet as follows:

```
New-SelfSignedCertificate -Subject "CN=HPC Pack 2016 Communication" -KeySpec KeyExchange -TextExtension @("2.5.29.37={text}1.3.6.1.5.5.7.3.1,1.3.6.1.5.5.7.3.2") -CertStoreLocation cert:\CurrentUser\My -KeyExportPolicy Exportable -NotAfter (Get-Date).AddYears(5)
```

- For operating systems earlier than Windows 10 or Windows Server 2016, download the [self-signed certificate generator](#) from the Microsoft Script Center. Extract its contents and run the following commands at a PowerShell prompt:

```
Import-Module -Name c:\ExtractedModule\New-SelfSignedCertificateEx.ps1

New-SelfSignedCertificateEx -Subject "CN=HPC Pack 2016 Communication" -KeySpec Exchange -KeyUsage "DigitalSignature,KeyEncipherment" -EnhancedKeyUsage "Server Authentication","Client Authentication" -StoreLocation CurrentUser -Exportable -NotAfter (Get-Date).AddYears(5)
```

Upload certificate to an Azure key vault

Before deploying the HPC cluster, upload the certificate to an [Azure key vault](#) as a secret, and record the following information for use during the deployment: **Vault name**, **Vault resource group**, **Certificate URL**, and **Certificate thumbprint**.

A sample PowerShell script to upload the certificate follows. For more information about uploading a certificate to an Azure key vault, see [Get started with Azure Key Vault](#).

```

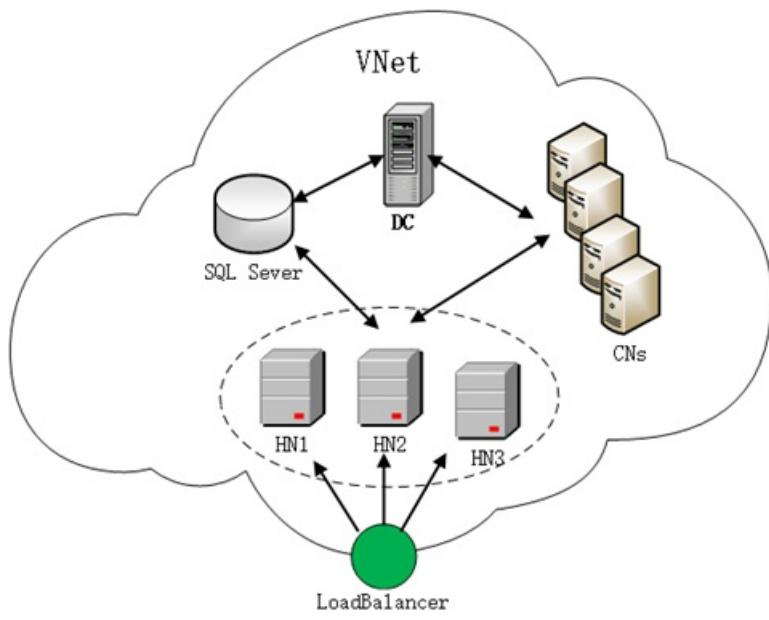
#Give the following values
$VaultName = "mytestvault"
$SecretName = "hpcpxcert"
$VaultRG = "myresourcegroup"
$location = "westus"
$PfxFile = "c:\Temp\mytest.pfx"
$Password = "yourpfxkeyprotectionpassword"
#Validate the pfx file
try {
    $pfxCert = New-Object System.Security.Cryptography.X509Certificates.X509Certificate2 -ArgumentList
$PfxFile, $Password
}
catch [System.Management.Automation.MethodInvocationException]
{
    throw $_.Exception.InnerException
}


```

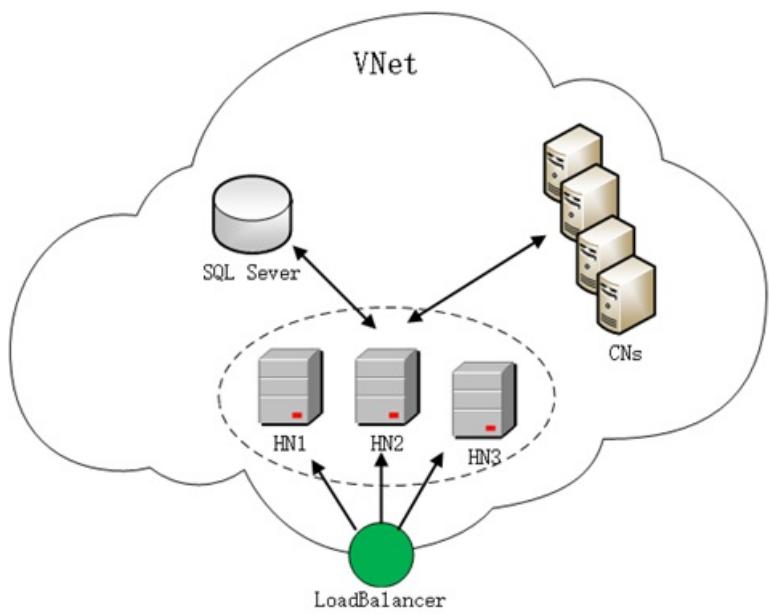
Supported topologies

Choose one of the [Azure Resource Manager templates](#) to deploy the HPC Pack 2016 cluster. Following are high-level architectures of three supported cluster topologies. High-availability topologies include multiple cluster head nodes.

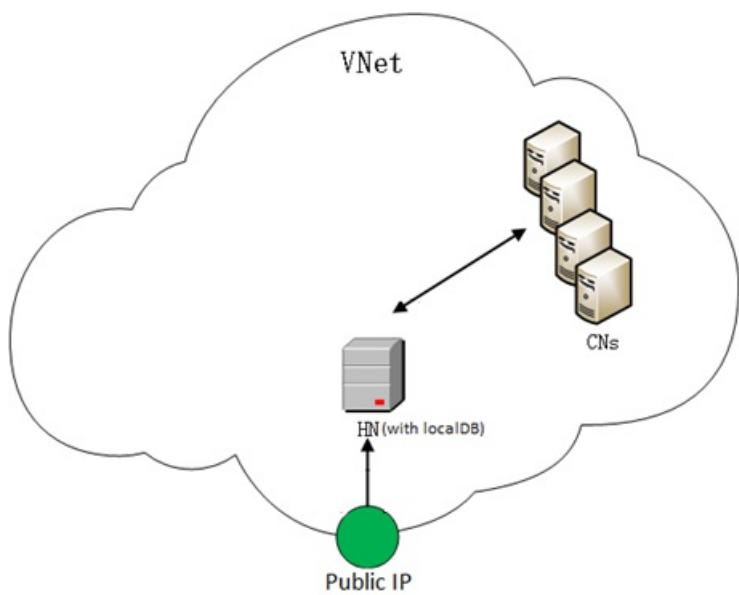
1. High-availability cluster with Active Directory domain



2. High-availability cluster without Active Directory domain



3. Cluster with a single head node



Deploy a cluster

To create the cluster, choose a template and click **Deploy to Azure**. In the [Azure portal](#), specify parameters for the template as described in the following steps. Each template creates all Azure resources required for the HPC cluster infrastructure. Resources include an Azure virtual network, public IP address, load balancer (only for a high-availability cluster), network interfaces, availability sets, storage accounts, and virtual machines.

Step 1: Select the subscription, location, and resource group

The **Subscription** and the **Location** must be same that you specified when you uploaded your PFX certificate (see Prerequisites). We recommend that you create a **Resource group** for the deployment.

Step 2: Specify the parameter settings

Enter or modify values for the template parameters. Click the icon next to each parameter for help information. Also see the guidance for [available VM sizes](#).

Specify the values you recorded in the Prerequisites for the following parameters: **Vault name**, **Vault resource group**, **Certificate URL**, and **Certificate thumbprint**.

Step 3. Review legal terms and create

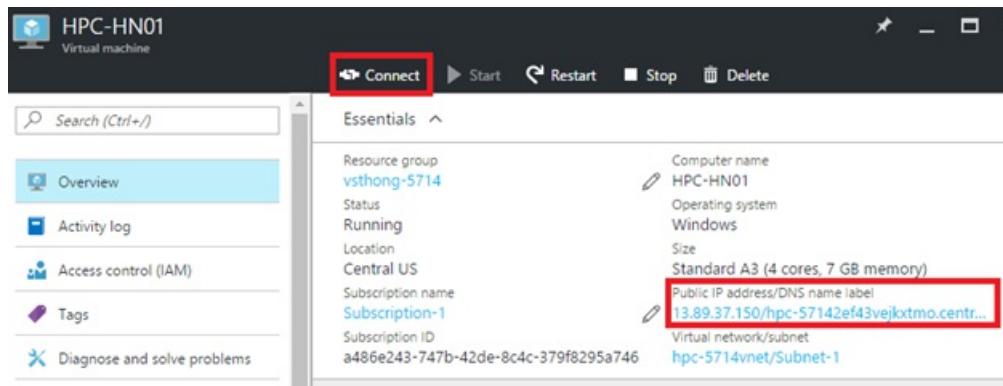
Click **Review legal terms** to review the terms. If you agree, click **Purchase**, and then click **Create** to start the deployment.

Connect to the cluster

- After the HPC Pack cluster is deployed, go to the [Azure portal](#). Click **Resource groups**, and find the resource group in which the cluster was deployed. You can find the head node virtual machines.

NAME	TYPE	LOCATION
hpc-5714-avset	Availability set	Central US
hpc-5714db	Virtual machine	Central US
hpc-5714dc	Virtual machine	Central US
HPC-HN01	Virtual machine	Central US
HPC-HN02	Virtual machine	Central US
HPC-HN03	Virtual machine	Central US
IaaSNC-000	Virtual machine	Central US
IaaSNC-001	Virtual machine	Central US
IaaSNC-002	Virtual machine	Central US
IaaSNC-003	Virtual machine	Central US

- Click one head node (in a high-availability cluster, click any of the head nodes). In **Essentials**, you can find the public IP address or full DNS name of the cluster.



3. Click **Connect** to log on to any of the head nodes using Remote Desktop with your specified administrator user name. If the cluster you deployed is in an Active Directory Domain, the user name is of the form <adminUsername> (for example, hpc.local\hpcadmin).

Next steps

- Submit jobs to your cluster. See [Submit jobs to HPC an HPC Pack cluster in Azure](#) and [Manage an HPC Pack 2016 cluster in Azure using Azure Active Directory](#).

Manage an HPC Pack cluster in Azure using Azure Active Directory

11/20/2017 • 7 min to read • [Edit Online](#)

Microsoft HPC Pack 2016 supports integration with [Azure Active Directory](#) (Azure AD) for administrators who deploy an HPC Pack cluster in Azure.

Follow the steps in this article for the following high level tasks:

- Manually integrate your HPC Pack cluster with your Azure AD tenant
- Manage and schedule jobs in your HPC Pack cluster in Azure

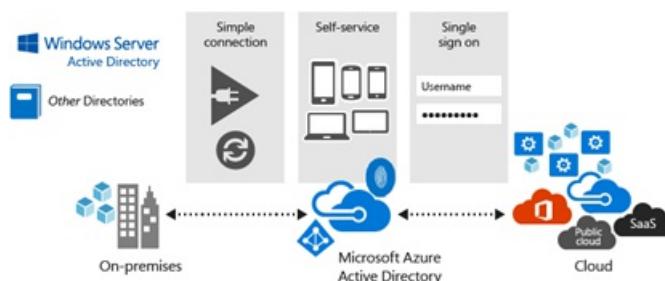
Integrating an HPC Pack cluster solution with Azure AD follows standard steps to integrate other applications and services. This article assumes you are familiar with basic user management in Azure AD. For more information and background, see the [Azure Active Directory documentation](#) and the following section.

Benefits of integration

Azure Active Directory (Azure AD) is a multi-tenant cloud-based directory and identity management service that provides single sign-on (SSO) access to cloud solutions.

Integration of an HPC Pack cluster with Azure AD can help you achieve the following goals:

- Remove the traditional Active Directory domain controller from the HPC Pack cluster. This can help reduce the costs of maintaining the cluster if this is not necessary for your business, and speed-up the deployment process.
- Leverage the following benefits that are brought by Azure AD:
 - Single sign-on
 - Using a local AD identity for the HPC Pack cluster in Azure



Prerequisites

- **HPC Pack 2016 cluster deployed in Azure virtual machines** - For steps, see [Deploy an HPC Pack 2016 cluster in Azure](#). You need the DNS name of the head node and the credentials of a cluster administrator to complete the steps in this article.

NOTE

Azure Active Directory integration is not supported in versions of HPC Pack before HPC Pack 2016.

- **Client computer** - You need a Windows or Windows Server client computer to run HPC Pack client utilities. If you only want to use the HPC Pack web portal or REST API to submit jobs, you can use any client

computer of your choice.

- **HPC Pack client utilities** - Install the HPC Pack client utilities on the client computer, using the free installation package available from the Microsoft Download Center.

Step 1: Register the HPC cluster server with your Azure AD tenant

1. Sign in to the [Azure portal](#).
2. If your account gives you access to more than one Azure AD tenant, click your account in the top right corner. Then set your portal session to the desired tenant. You must have permission to access resources in the directory.
3. Click **Azure Active Directory** in the left Services navigation pane, click **Users and groups**, and make sure there are user accounts already created or configured.
4. In **Azure Active Directory**, click **App registrations > New application registration**. Enter the following information:
 - **Name** - HPCPackClusterServer
 - **Application type** - Select **Web app / API**
 - **Sign-on URL**- The base URL for the sample, which is by default `https://hpcserver`
 - Click **Create**.
5. After the app is added, select it in the **App registrations** list. Then click **Settings > Properties**. Enter the following information:
 - Select **Yes** for **Multi-tenanted**.
 - Change **App ID URI** to `https://<Directory_name>/<application_name>`. Replace `<Directory_name>` with the full name of your Azure AD tenant, for example, `hpclocal.onmicrosoft.com`, and replace `<application_name>` with the name you chose previously.
6. Click **Save**. When saving completes, on the app page, click **Manifest**. Edit the manifest by locating the `appRoles` setting and adding the following application role, and then click **Save**:

```
"appRoles": [  
    {  
        "allowedMemberTypes": [  
            "User",  
            "Application"  
        ],  
        "displayName": "HpcAdminMirror",  
        "id": "61e10148-16a8-432a-b86d-ef620c3e48ef",  
        "isEnabled": true,  
        "description": "HpcAdminMirror",  
        "value": "HpcAdminMirror"  
    },  
    {  
        "allowedMemberTypes": [  
            "User",  
            "Application"  
        ],  
        "description": "HpcUsers",  
        "displayName": "HpcUsers",  
        "id": "91e10148-16a8-432a-b86d-ef620c3e48ef",  
        "isEnabled": true,  
        "value": "HpcUsers"  
    }  
,  
],
```

7. In **Azure Active Directory**, click **Enterprise applications > All applications**. Select **HPCPackClusterServer** from the list.
8. Click **Properties**, and change **User assignment required** to **Yes**. Click **Save**.

9. Click **Users and groups** > **Add user**. Select a user and select a role, and then click **Assign**. Assign one of the available roles (HpcUsers or HpcAdminMirror) to the user. Repeat this step with additional users in the directory. For background information about cluster users, see [Managing Cluster Users](#).

Step 2: Register the HPC cluster client with your Azure AD tenant

1. Sign in to the [Azure portal](#).
2. If your account gives you access to more than one Azure AD tenant, click your account in the top right corner. Then set your portal session to the desired tenant. You must have permission to access resources in the directory.
3. In **Azure Active Directory**, click **App registrations** > **New application registration**. Enter the following information:
 - **Name** - HPCPackClusterClient
 - **Application type** - Select **Native**
 - **Redirect URI** - `http://hpcclient`
 - Click **Create**
4. After the app is added, select it in the **App registrations** list. Copy the **Application ID** value and save it. You need this later when configuring your application.
5. Click **Settings** > **Required permissions** > **Add** > **Select an API**. Search and select the HpcPackClusterServer application (created in Step 1).
6. In the **Enable Access** page, select **Access HpcClusterServer**. Then click **Done**.

Step 3: Configure the HPC cluster

1. Connect to the HPC Pack 2016 head node in Azure.
2. Start HPC PowerShell.
3. Run the following command:

```
Set-HpcClusterRegistry -SupportAAD true -AADInstance https://login.microsoftonline.com/ -AADAppName HpcPackClusterServer -AADTenant <your AAD tenant name> -AADClientAppId <client ID> -AADClientAppRedirectUri http://hpcclient
```

where

- **AADTenant** specifies the Azure AD tenant name, such as `hpclocal.onmicrosoft.com`
 - **AADClientAppId** specifies the Application ID for the app created in Step 2.
4. Do one of the following, depending on the head node configuration:
 - In a single head node HPC Pack cluster, restart the HpcScheduler service.
 - In an HPC Pack cluster with multiple head nodes, run the following PowerShell commands on the head node to restart the HpcSchedulerStateful service:

```
Connect-ServiceFabricCluster  
Move-ServiceFabricPrimaryReplica -ServiceName "fabric:/HpcApplication/SchedulerStatefulService"
```

Step 4: Manage and submit jobs from the client

To install the HPC Pack client utilities on your computer, download the HPC Pack 2016 setup files (full installation) from the Microsoft Download Center. When you begin the installation, choose the setup option for the **HPC Pack client utilities**.

To prepare the client computer, install the certificate used during [HPC cluster setup](#) on the client computer. Use standard Windows certificate management procedures to install the public certificate to the **Certificates – Current user > Trusted Root Certification Authorities** store.

You can now run the HPC Pack commands or use the HPC Pack Job manager GUI to submit and manage cluster jobs by using the Azure AD account. For job submission options, see [Submit HPC jobs to an HPC Pack cluster in Azure](#).

NOTE

When you try to connect to the HPC Pack cluster in Azure for the first time, a popup window appears. Enter your Azure AD credentials to log in. The token is then cached. Later connections to the cluster in Azure use the cached token unless authentication changes or the cache is cleared.

For example, after completing the previous steps, you can query for jobs from an on-premises client as follows:

```
Get-HpcJob -State All -Scheduler https://<Azure load balancer DNS name> -Owner <Azure AD account>
```

Useful cmdlets for job submission with Azure AD integration

Manage the local token cache

HPC Pack 2016 provides the following HPC PowerShell cmdlets to manage the local token cache. These cmdlets are useful for submitting jobs non-interactively. See the following example:

```
Remove-HpcTokenCache  
  
$SecurePassword = "<password>" | ConvertTo-SecureString -AsPlainText -Force  
  
Set-HpcTokenCache -UserName <AADUsername> -Password $SecurePassword -scheduler https://<Azure load balancer  
DNS name>
```

Set the credentials for submitting jobs using the Azure AD account

Sometimes, you may want to run the job under the HPC cluster user (for a domain-joined HPC cluster, run as one domain user; for a non-domain-joined HPC cluster, run as one local user on the head node).

1. Use the following commands to set the credentials:

```
$localUser = "<username>"  
  
$localUserPassword="<password>"  
  
$secpasswd = ConvertTo-SecureString $localUserPassword -AsPlainText -Force  
  
$mycreds = New-Object System.Management.Automation.PSCredential ($localUser, $secpasswd)  
  
Set-HpcJobCredential -Credential $mycreds -Scheduler https://<Azure load balancer DNS name>
```

2. Then submit the job as follows. The job/task runs under \$localUser on the compute nodes.

```
$emptycreds = New-Object System.Management.Automation.PSCredential ($localUser, (new-object  
System.Security.SecureString))  
...  
$job = New-HpcJob -Scheduler https://<Azure load balancer DNS name>  
  
Add-HpcTask -Job $job -CommandLine "ping localhost" -Scheduler https://<Azure load balancer DNS name>  
  
Submit-HpcJob -Job $job -Scheduler https://<Azure load balancer DNS name> -Credential $emptycreds
```

If `-Credential` is not specified with `Submit-HpcJob`, the job or task runs under a local mapped user as the Azure AD account. (The HPC cluster creates a local user with the same name as the Azure AD account to run the task.)

3. Set extended data for the Azure AD account. This is useful when running an MPI job on Linux nodes using the Azure AD account.

- Set extended data for the Azure AD account itself

```
Set-HpcJobCredential -Scheduler https://<Azure load balancer DNS name> -ExtendedData <data> -  
AdUser
```

- Set extended data and run as HPC cluster user

```
Set-HpcJobCredential -Credential $mycreds -Scheduler https://<Azure load balancer DNS name> -  
ExtendedData <data>
```

Create the head node of an HPC Pack cluster in an Azure VM with a Marketplace image

6/27/2017 • 4 min to read • [Edit Online](#)

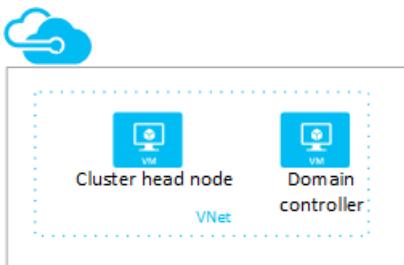
Use a [Microsoft HPC Pack 2012 R2 virtual machine image](#) from the Azure Marketplace and the Azure portal to create the head node of an HPC cluster. This HPC Pack VM image is based on Windows Server 2012 R2 Datacenter with HPC Pack 2012 R2 Update 3 pre-installed. Use this head node for a proof of concept deployment of HPC Pack in Azure. You can then add compute nodes to the cluster to run HPC workloads.

TIP

To deploy a complete HPC Pack 2012 R2 cluster in Azure that includes the head node and compute nodes, we recommend that you use an automated method. Options include the [HPC Pack IaaS deployment script](#) and Resource Manager templates such as the [HPC Pack cluster for Windows workloads](#). Resource Manager templates are also available for [Microsoft HPC Pack 2016 clusters](#).

Planning considerations

As shown in the following figure, you deploy the HPC Pack head node in an Active Directory domain in an Azure virtual network.



- **Active Directory domain:** The HPC Pack 2012 R2 head node must be joined to an Active Directory domain in Azure before you start the HPC services on the VM. As shown in this article, for a proof of concept deployment, you can promote the VM you create for the head node as a domain controller before starting the HPC services. Another option is to deploy a separate domain controller and forest in Azure to which you join the head node VM.
- **Deployment model:** For most new deployments, Microsoft recommends that you use the Resource Manager deployment model. This article assumes that you use this deployment model.
- **Azure virtual network:** When you use the Resource Manager deployment model to deploy the head node, you specify or create an Azure virtual network. You use the virtual network if you need to join the head node to an existing Active Directory domain. You also need it later to add compute node VMs to the cluster.

Steps to create the head node

Following are high-level steps to use the Azure portal to create an Azure VM for the HPC Pack head node by using the Resource Manager deployment model.

1. If you want to create a new Active Directory forest in Azure with separate domain controller VMs, one option is to use a [Resource Manager template](#). For a simple proof of concept deployment, it's fine to omit this step and configure the head node VM itself as a domain controller. This option is described later in this article.

2. On the [HPC Pack 2012 R2 on Windows Server 2012 R2 page](#) in the Azure Marketplace, click **Create Virtual Machine**.
3. In the portal, on the **HPC Pack 2012 R2 on Windows Server 2012 R2** page, select the **Resource Manager** deployment model and then click **Create**.



HPC Pack 2012 R2 on Windows Server 2012 R2

Microsoft

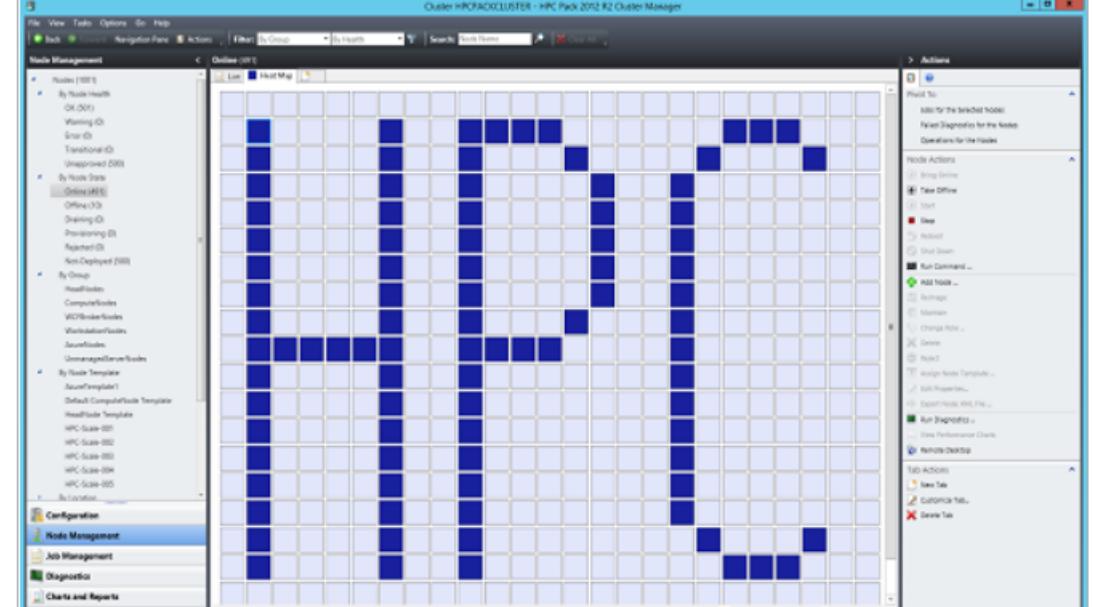
This image contains the Windows Server 2012 R2 Datacenter operating system with HPC Pack 2012 R2 Update 3 installed to create the head node of an HPC Pack cluster. Microsoft SQL Server 2014 Express is also pre-installed. Use this image to create the head node of a Windows high performance computing (HPC) cluster in Azure. We recommend using a VM size of at least A4. If you plan to add cluster compute nodes to the head node, the VM must be created in an Azure virtual network. Configure the network before creating the VM. To use the head node, you will need to join the virtual machine to an Active Directory domain and run the post-configuration script described [here](#). It is strongly recommended to use the Azure Resource Manager (ARM) templates or the HPC Pack IaaS deployment script to automatically create an HPC cluster in Azure with the HPC Pack images. For more information [see here](#).

Legal Terms

By clicking the Create button, I acknowledge that I am getting this software from Microsoft and that the [legal terms](#) of Microsoft apply to it. Microsoft does not provide rights for third-party software. Also see the [privacy statement](#) from Microsoft.

[!\[\]\(3a16f2e5aceae803cc55a453cc30ced3_img.jpg\)](#)
[!\[\]\(9a7141077f95143aaec8312eb85367e7_img.jpg\)](#)
[!\[\]\(87fadcf44603fc8126432d145ae305d5_img.jpg\)](#)
[!\[\]\(f9b483c43b5d6825d6d11e644b0a31b8_img.jpg\)](#)
[!\[\]\(c3b68ade8c4732862e89af434f35c927_img.jpg\)](#)
[!\[\]\(1b871c20ffba57864268d03d7c921dac_img.jpg\)](#)

Cluster HPCPACK2012R2 - HPC Pack 2012 R2 Cluster Manager



Select a deployment model i

Resource Manager

Create

4. Use the portal to configure the settings and create the VM. If you're new to Azure, follow the tutorial [Create a Windows virtual machine in the Azure portal](#). For a proof of concept deployment, you can usually accept the default or recommended settings.

NOTE

If you want to join the head node to an existing Active Directory domain in Azure, make sure you specify the virtual network for that domain when creating the VM.

5. After you create the VM and the VM is running, [connect to the VM](#) by Remote Desktop.
6. Join the VM to an Active Directory domain forest by choosing one of the following options:
 - If you created the VM in an Azure virtual network with an existing domain forest, join the VM to the forest by using standard Server Manager or Windows PowerShell tools. Then restart.
 - If you created the VM in a new virtual network (without an existing domain forest), then promote the VM as a domain controller. Use standard steps to install and configure the Active Directory Domain Services role on the head node. For detailed steps, see [Install a New Windows Server 2012 Active Directory Forest](#).
7. After the VM is running and is joined to an Active Directory forest, start the HPC Pack services as follows:
 - a. Connect to the head node VM using a domain account that is a member of the local Administrators group. For example, use the administrator account you set up when you created the head node VM.
 - b. For a default head node configuration, start Windows PowerShell as an administrator and type the following:

```
& $env:CCP_HOME\bin\HPCHNPrepare.ps1 -DBServerInstance ".\ComputeCluster"
```

It can take several minutes for the HPC Pack services to start.

For additional head node configuration options, type `get-help HPCHNPrepare.ps1`.

Next steps

- You can now work with the head node of your HPC Pack cluster. For example, start HPC Cluster Manager, and complete the [Deployment To-do List](#).
- If you want to increase the cluster compute capacity on-demand, add [Azure burst nodes](#) in a cloud service.
- Try running a test workload on the cluster. For an example, see the HPC Pack [getting started guide](#).

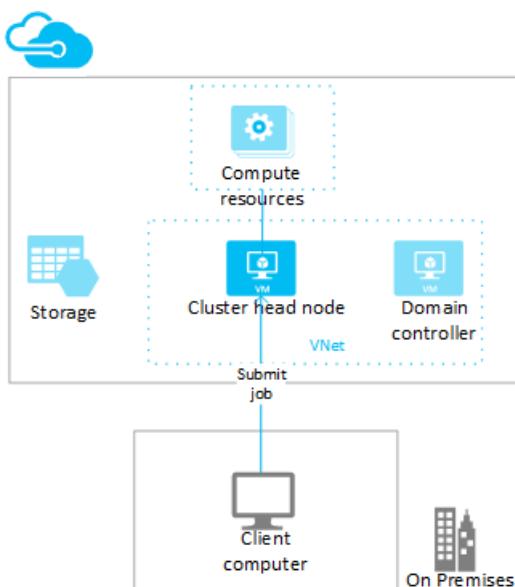
Submit HPC jobs from an on-premises computer to an HPC Pack cluster deployed in Azure

6/27/2017 • 7 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager](#) and [classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

Configure an on-premises client computer to submit jobs to a [Microsoft HPC Pack](#) cluster in Azure. This article shows you how to set up a local computer with client tools to submit job over HTTPS to the cluster in Azure. In this way, several cluster users can submit jobs to a cloud-based HPC Pack cluster, but without connecting directly to the head node VM or accessing an Azure subscription.



Prerequisites

- **HPC Pack head node deployed in an Azure VM** - We recommend that you use automated tools such as an [Azure quickstart template](#) or an [Azure PowerShell script](#) to deploy the head node and cluster. You need the DNS name of the head node and the credentials of a cluster administrator to complete the steps in this article.
- **Client computer** - You need a Windows or Windows Server client computer that can run HPC Pack client utilities (see [system requirements](#)). If you only want to use the HPC Pack web portal or REST API to submit jobs, you can use any client computer of your choice.
- **HPC Pack installation media** - To install the HPC Pack client utilities, the free installation package for the latest version of HPC Pack (HPC Pack 2012 R2) is available from the [Microsoft Download Center](#). Make sure that you download the same version of HPC Pack that is installed on the head node VM.

Step 1: Install and configure the web components on the head node

To enable a REST interface to submit jobs to the cluster over HTTPS, ensure that the HPC Pack web components are configured on the HPC Pack head node. If they aren't already installed, first install the web components by running the `HpcWebComponents.msi` installation file. Then, configure the components by running the HPC PowerShell script `Set-HPCWebComponents.ps1`.

For detailed procedures, see [Install the Microsoft HPC Pack Web Components](#).

TIP

Certain Azure quickstart templates for HPC Pack install and configure the web components automatically. If you use the [HPC Pack IaaS deployment script](#) to create the cluster, you can optionally install and configure the web components as part of the deployment.

To install the web components

1. Connect to the head node VM by using the credentials of a cluster administrator.
2. From the HPC Pack Setup folder, run HpcWebComponents.msi on the head node.
3. Follow the steps in the wizard to install the web components

To configure the web components

1. On the head node, start HPC PowerShell as an administrator.
2. To change directory to the location of the configuration script, type the following command:

```
cd $env:CCP_HOME\bin
```

3. To configure the REST interface and start the HPC Web Service, type the following command:

```
.\Set-HPCWebComponents.ps1 -Service REST -enable
```

4. When prompted to select a certificate, choose the certificate that corresponds to the public DNS name of the head node. For example, if you deploy the head node VM using the classic deployment model, the certificate name looks like CN=<HeadNodeDnsName>.cloudapp.net. If you use the Resource Manager deployment model, the certificate name looks like CN=<HeadNodeDnsName>.<region>.cloudapp.azure.com.

NOTE

You select this certificate later when you submit jobs to the head node from an on-premises computer. Don't select or configure a certificate that corresponds to the computer name of the head node in the Active Directory domain (for example, CN=MyHPCHeadNode.HpcAzure.local).

5. To configure the web portal for job submission, type the following command:

```
.\Set-HPCWebComponents.ps1 -Service Portal -enable
```

6. After the script completes, stop and restart the HPC Job Scheduler Service by typing the following commands:

```
net stop hpcscheduler  
net start hpcscheduler
```

Step 2: Install the HPC Pack client utilities on an on-premises computer

If you want to install the HPC Pack client utilities on your computer, download the HPC Pack setup files (full installation) from the [Microsoft Download Center](#). When you begin the installation, choose the setup option for the [HPC Pack client utilities](#).

To use the HPC Pack client tools to submit jobs to the head node VM, you also need to export a certificate from the head node and install it on the client computer. The certificate must be in .CER format.

To export the certificate from the head node

1. On the head node, add the Certificates snap-in to a Microsoft Management Console for the Local Computer account. For steps to add the snap-in, see [Add the Certificates Snap-in to an MMC](#).
2. In the console tree, expand **Certificates – Local Computer > Personal**, and then click **Certificates**.
3. Locate the certificate that you configured for the HPC Pack web components in [Step 1: Install and configure the web components on the head node](#) (for example, CN=<HeadNodeDnsName>.cloudapp.net).
4. Right-click the certificate, and click **All Tasks > Export**.
5. In the Certificate Export Wizard, click **Next**, and ensure that **No, do not export the private key** is selected.
6. Follow the remaining steps of the wizard to export the certificate in DER encoded binary X.509 (.CER) format.

To import the certificate on the client computer

1. Copy the certificate that you exported from the head node to a folder on the client computer.
2. On the client computer, run certmgr.msc.
3. In Certificate Manager, expand **Certificates – Current user > Trusted Root Certification Authorities**, right-click **Certificates**, and then click **All Tasks > Import**.
4. In the Certificate Import Wizard, click **Next** and follow the steps to import the certificate that you exported from the head node to the Trusted Root Certification Authorities store.

TIP

You might see a security warning, because the certification authority on the head node isn't recognized by the client computer. For testing purposes, you can ignore this warning and complete the certificate import.

Step 3: Run test jobs on the cluster

To verify your configuration, try running jobs on the cluster in Azure from the on-premises computer. For example, you can use HPC Pack GUI tools or command-line commands to submit jobs to the cluster. You can also use a web-based portal to submit jobs.

To run job submission commands on the client computer

1. On a client computer where the HPC Pack client utilities are installed, start a Command Prompt.
2. Type a sample command. For example, to list all jobs on the cluster, type a command similar to one of the following, depending on the full DNS name of the head node:

```
job list /scheduler:https://<HeadNodeDnsName>.cloudapp.net /all
```

or

```
job list /scheduler:https://<HeadNodeDnsName>.<region>.cloudapp.azure.com /all
```

TIP

Use the full DNS name of the head node, not the IP address, in the scheduler URL. If you specify the IP address, an error appears similar to "The server certificate needs to either have a valid chain of trust or to be placed in the trusted root store."

- When prompted, type the user name (in the form <DomainName>\<UserName>) and password of the HPC cluster administrator or another cluster user that you configured. You can choose to store the credentials locally for more job operations.

A list of jobs appears.

To use HPC Job Manager on the client computer

- If you didn't previously store domain credentials for a cluster user when submitting a job, you can add the credentials in Credential Manager.
 - In Control Panel on the client computer, start Credential Manager.
 - Click **Windows Credentials** > **Add a generic credential**.
 - Specify the Internet address (for example, <https://<HeadNodeDnsName>.cloudapp.net/HpcScheduler> or <https://<HeadNodeDnsName>.<region>.cloudapp.azure.com/HpcScheduler>), and the user name (<DomainName>\<UserName>) and password of the cluster administrator or another cluster user that you configured.
- On the client computer, start HPC Job Manager.
- In the **Select Head Node** dialog box, type the URL to the head node in Azure (for example, <https://<HeadNodeDnsName>.cloudapp.net> or <https://<HeadNodeDnsName>.<region>.cloudapp.azure.com>).

HPC Job Manager opens and shows a list of jobs on the head node.

To use the web portal running on the head node

- Start a web browser on the client computer, and enter one of the following addresses, depending on the full DNS name of the head node:

`https://<HeadNodeDnsName>.cloudapp.net/HpcPortal`

or

`https://<HeadNodeDnsName>.<region>.cloudapp.azure.com/HpcPortal`

- In the security dialog box that appears, type the domain credentials of the HPC cluster administrator. (You can also add other cluster users in different roles. See [Managing Cluster Users](#).)

The web portal opens to the job list view.

- To submit a sample job that returns the string "Hello World" from the cluster, click **New job** in the left-hand navigation.
- On the **New Job** page, under **From submission pages**, click **HelloWorld**. The job submission page appears.
- Click **Submit**. If prompted, provide the domain credentials of the HPC cluster administrator. The job is submitted, and the job ID appears on the **My Jobs** page.
- To view the results of the job that you submitted, click the job ID, and then click **View Tasks** to view the command output (under **Output**).

Next steps

- You can also submit jobs to the Azure cluster with the [HPC Pack REST API](#).
- If you want to submit cluster jobs from a Linux client, see the Python sample in the [HPC Pack 2012 R2 SDK and Sample Code](#).

Get started running Excel and SOA workloads on an HPC Pack cluster in Azure

6/27/2017 • 14 min to read • [Edit Online](#)

This article shows you how to deploy a Microsoft HPC Pack 2012 R2 cluster on Azure virtual machines by using an Azure quickstart template, or optionally an Azure PowerShell deployment script. The cluster uses Azure Marketplace VM images designed to run Microsoft Excel or service-oriented architecture (SOA) workloads with HPC Pack. You can use the cluster to run Excel HPC and SOA services from an on-premises client computer. The Excel HPC services include Excel workbook offloading and Excel user-defined functions, or UDFs.

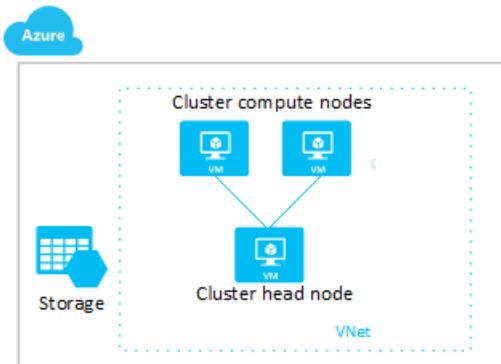
IMPORTANT

This article is based on features, templates, and scripts for HPC Pack 2012 R2. This scenario is not currently supported in HPC Pack 2016.

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

At a high level, the following diagram shows the HPC Pack cluster you create.



Prerequisites

- **Client computer** - You need a Windows-based client computer to submit sample Excel and SOA jobs to the cluster. You also need a Windows computer to run the Azure PowerShell cluster deployment script (if you choose that deployment method).
- **Azure subscription** - If you don't have an Azure subscription, you can create a [free account](#) in just a couple of minutes.
- **Cores quota** - You might need to increase the quota of cores, especially if you deploy several cluster nodes with multicore VM sizes. If you are using an Azure quickstart template, the cores quota in Resource Manager is per Azure region. In that case, you might need to increase the quota in a specific region. See [Azure subscription limits, quotas, and constraints](#). To increase a quota, [open an online customer support request](#) at no charge.
- **Microsoft Office license** - If you deploy compute nodes using a Marketplace HPC Pack 2012 R2 VM image with Microsoft Excel, a 30-day evaluation version of Microsoft Excel Professional Plus 2013 is installed. After the evaluation period, you need to provide a valid Microsoft Office license to activate Excel to continue to run workloads. See [Excel activation](#) later in this article.

Step 1. Set up an HPC Pack cluster in Azure

We show two options to set up the HPC Pack 2012 R2 cluster: first, using an Azure quickstart template and the Azure portal; and second, using an Azure PowerShell deployment script.

Option 1. Use a quickstart template

Use an Azure quickstart template to quickly deploy an HPC Pack cluster in the Azure portal. When you open the template in the portal, you get a simple UI where you enter the settings for your cluster. Here are the steps.

TIP

If you want, use an [Azure Marketplace template](#) that creates a similar cluster specifically for Excel workloads. The steps differ slightly from the following.

1. Visit the [Create HPC Cluster template page on GitHub](#). If you want, review information about the template and the source code.
2. Click **Deploy to Azure** to start a deployment with the template in the Azure portal.

Create HPC cluster with Windows compute nodes



This template allows you to create an HPC cluster with Windows compute nodes. You can choose HPC Pack 2012 R2 Compute Node image or HPC Pack 2012 R2 Compute Node with Excel image to deploy compute nodes.

3. In the portal, follow these steps to enter the parameters for the HPC cluster template.
 - a. On the **Parameters** page, enter or modify values for the template parameters. (Click the icon next to each setting for help information.) Sample values are shown in the following screen. This example creates a cluster named *hpc01* in the *hpc.local* domain consisting of a head node and 2 compute nodes. The compute nodes are created from an HPC Pack VM image that includes Microsoft Excel.

Create an HPC cluster

Azure quickstart template

TEMPLATE

 create-hpc-cluster
1 resource

Edit template Edit parameters Learn more

BASICS

* Subscription Microsoft Azure Internal Consumption

* Resource group Create new Use existing

* Location East US

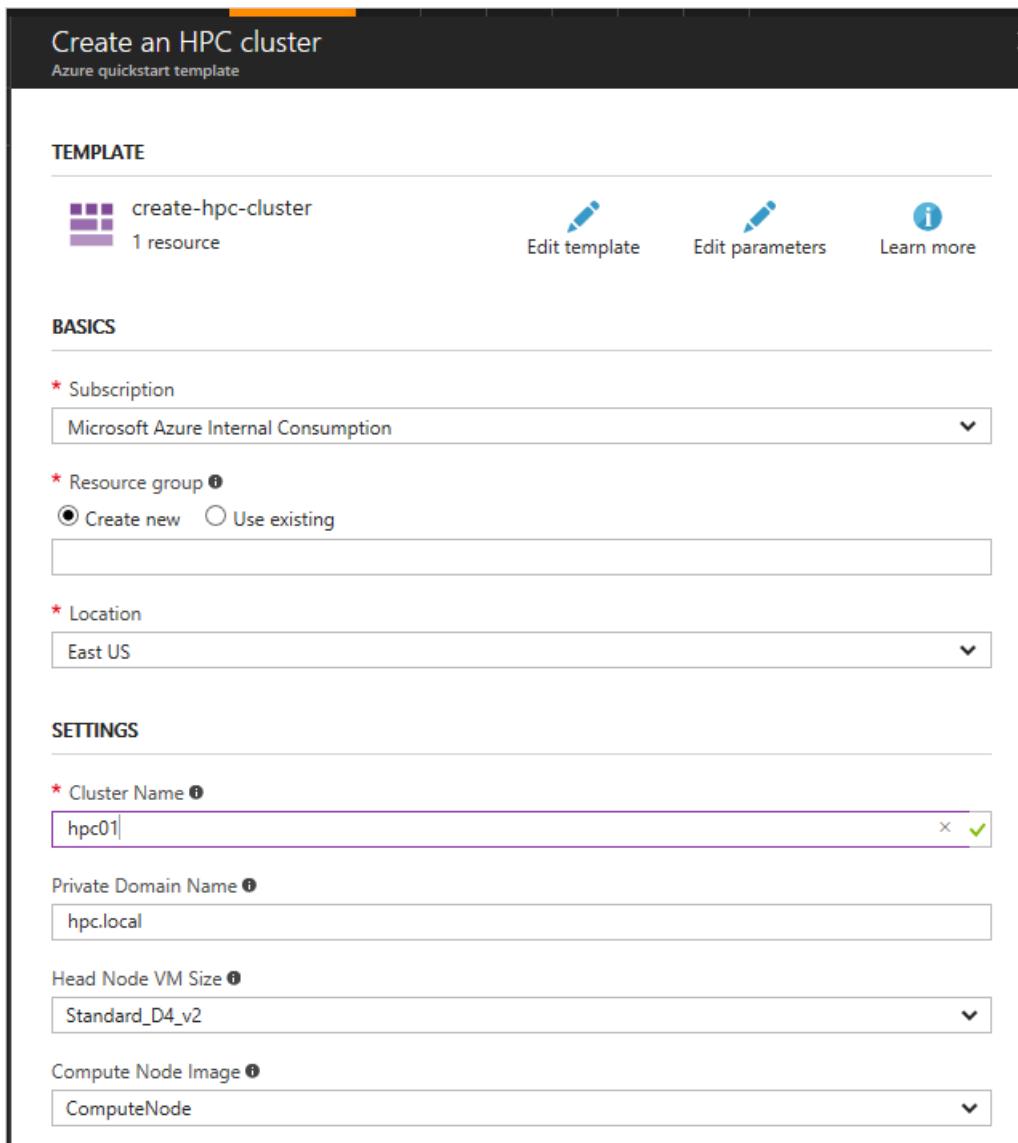
SETTINGS

* Cluster Name hpc01

Private Domain Name hpc.local

Head Node VM Size Standard_D4_v2

Compute Node Image ComputeNode



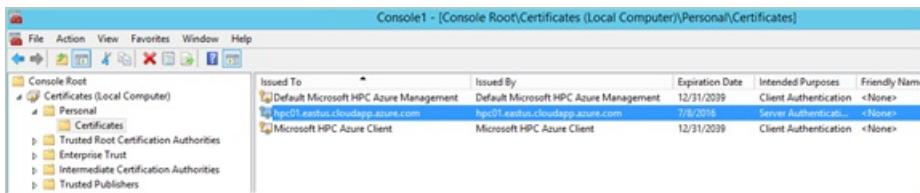
NOTE

The head node VM is created automatically from the [latest Marketplace image](#) of HPC Pack 2012 R2 on Windows Server 2012 R2. Currently the image is based on HPC Pack 2012 R2 Update 3.

Compute node VMs are created from the latest image of the selected compute node family. Select the **ComputeNodeWithExcel** option for the latest HPC Pack compute node image that includes an evaluation version of Microsoft Excel Professional Plus 2013. To deploy a cluster for general SOA sessions or for Excel UDF offloading, choose the **ComputeNode** option (without Excel installed).

- b. Choose the subscription.
 - c. Create a resource group for the cluster, such as *hpc01RG*.
 - d. Choose a location for the resource group, such as Central US.
 - e. On the **Legal terms** page, review the terms. If you agree, click **Purchase**. Then, when you are finished setting the values for the template, click **Create**.
4. When the deployment completes (it typically takes around 30 minutes), export the cluster certificate file from the cluster head node. In a later step, you import this public certificate on the client computer to provide the server-side authentication for secure HTTP binding.
- a. In the Azure portal, go to the dashboard, select the head node, and click **Connect** at the top of the page to connect using Remote Desktop.

b. Use standard procedures in Certificate Manager to export the head node certificate (located under Cert:\LocalMachine\My) without the private key. In this example, export CN = *hpc01.eastus.cloudapp.azure.com*.



Option 2. Use the HPC Pack IaaS Deployment script

The HPC Pack IaaS deployment script provides another versatile way to deploy an HPC Pack cluster. It creates a cluster in the classic deployment model, whereas the template uses the Azure Resource Manager deployment model. Also, the script is compatible with a subscription in either the Azure Global or Azure China service.

Additional prerequisites

- **Azure PowerShell** - [Install and configure Azure PowerShell](#) (version 0.8.10 or later) on your client computer.
- **HPC Pack IaaS deployment script** - Download and unpack the latest version of the script from the [Microsoft Download Center](#). Check the version of the script by running `New-HPCIaaSCluster.ps1 -Version`. This article is based on version 4.5.0 or later of the script.

Create the configuration file

The HPC Pack IaaS deployment script uses an XML configuration file as input that describes the infrastructure of the HPC cluster. To deploy a cluster consisting of a head node and 18 compute nodes created from the compute node image that includes Microsoft Excel, substitute values for your environment into the following sample configuration file. For more information about the configuration file, see the Manual.rtf file in the script folder and [Create an HPC cluster with the HPC Pack IaaS deployment script](#).

```

<?xml version="1.0" encoding="utf-8"?>
<IaaSClusterConfig>
  <Subscription>
    <SubscriptionName>MySubscription</SubscriptionName>
    <StorageAccount>hpc01</StorageAccount>
  </Subscription>
  <Location>West US</Location>
  <VNet>
    <VNetName>hpc-vnet01</VNetName>
    <SubnetName>Subnet-1</SubnetName>
  </VNet>
  <Domain>
    <DCOption>NewDC</DCOption>
    <DomainFQDN>hpc.local</DomainFQDN>
    <DomainController>
      <VMName>HPCExcelDC01</VMName>
      <ServiceName>HPCExcelDC01</ServiceName>
      <VMSize>Medium</VMSize>
    </DomainController>
  </Domain>
  <Database>
    <DBOption>LocalDB</DBOption>
  </Database>
  <HeadNode>
    <VMName>HPCExcelHN01</VMName>
    <ServiceName>HPCExcelHN01</ServiceName>
    <VMSize>Large</VMSize>
    <EnableRESTAPI/>
    <EnableWebPortal/>
    <PostConfigScript>C:\tests\PostConfig.ps1</PostConfigScript>
  </HeadNode>
  <ComputeNodes>
    <VMNamePattern>HPCExcelCN%00%</VMNamePattern>
    <ServiceName>HPCExcelCN01</ServiceName>
    <VMSize>Medium</VMSize>
    <NodeCount>18</NodeCount>
    <ImageName>HPCPack2012R2_ComputeNodeWithExcel</ImageName>
  </ComputeNodes>
</IaaSClusterConfig>

```

Notes about the configuration file

- The **VMName** of the head node **MUST** be the same as the **ServiceName**, or SOA jobs fail to run.
- Make sure you specify **EnableWebPortal** so that the head node certificate is generated and exported.
- The file specifies a post-configuration PowerShell script PostConfig.ps1 that runs on the head node. The following sample script configures the Azure storage connection string, removes the compute node role from the head node, and brings all nodes online when they are deployed.

```

# add the HPC Pack powershell cmdlets
Add-PSSnapin Microsoft.HPC

# set the Azure storage connection string for the cluster
Set-HpcClusterProperty -AzureStorageConnectionString 'DefaultEndpointsProtocol=https;AccountName=<yourstorageaccountname>;AccountKey=<yourstorageaccountkey>'

# remove the compute node role for head node to make sure the Excel workbook won't run on head node
Get-HpcNode -GroupName HeadNodes | Set-HpcNodeState -State offline | Set-HpcNode -Role BrokerNode

# total number of nodes in the deployment including the head node and compute nodes, which should match the
# number specified in the XML configuration file
$TotalNumOfNodes = 19

$ErrorActionPreference = 'SilentlyContinue'

# bring nodes online when they are deployed until all nodes are online
while ($true)
{
    Get-HpcNode -State Offline | Set-HpcNodeState -State Online -Confirm:$false
    $OnlineNodes = @(Get-HpcNode -State Online)
    if ($OnlineNodes.Count -eq $TotalNumOfNodes)
    {
        break
    }
    sleep 60
}

```

Run the script

1. Open the PowerShell console on the client computer as an administrator.
2. Change directory to the script folder (E:\IaaSClusterScript in this example).

```
cd E:\IaaSClusterScript
```

3. To deploy the HPC Pack cluster, run the following command. This example assumes that the configuration file is located in E:\HPCDemoConfig.xml.

```
.\New-HpcIaaSCluster.ps1 -ConfigFile E:\HPCDemoConfig.xml -AdminUserName MyAdminName
```

The HPC Pack deployment script runs for some time. One thing the script does is to export and download the cluster certificate and save it in the current user's Documents folder on the client computer. The script generates a message similar to the following. In a following step, you import the certificate in the appropriate certificate store.

You have enabled REST API or web portal on HPC Pack head node. Please import the following certificate in the Trusted Root Certification Authorities certificate store on the computer where you are submitting job or accessing the HPC web portal:

C:\Users\hpcuser\Documents\HPCWebComponent_HPCExcelHN004_20150707162011.cer

Step 2. Offload Excel workbooks and run UDFs from an on-premises client

Excel activation

When using the ComputeNodeWithExcel VM image for production workloads, you need to provide a valid Microsoft Office license key to activate Excel on the compute nodes. Otherwise, the evaluation version of Excel expires after 30 days, and running Excel workbooks will fail with the COMException (0x800AC472).

You can rearm Excel for another 30 days of evaluation time: Log on to the head node and clusrun

```
%ProgramFiles(x86)%\Microsoft Office\Office15\OSPPREARM.exe
```

 on all Excel compute nodes via HPC Cluster Manager.

You can rearm a maximum of two times. After that, you must provide a valid Office license key.

The Office Professional Plus 2013 installed on the VM image is a volume edition with a Generic Volume License Key (GVLK). You can activate it via Key Management Service (KMS)/Active Directory-Based Activation (AD-BA) or Multiple Activation Key (MAK).

```
* To use KMS/AD-BA, use an existing KMS server or set up a new one by using the Microsoft Office 2013 Volume License Pack. (If you want to, set up the server on the head node.) Then, activate the KMS host key via the Internet or telephone. Then clusrun `ospp.vbs` to set the KMS server and port and activate Office on all the Excel compute nodes.
```

```
* To use MAK, first clusrun `ospp.vbs` to input the key and then activate all the Excel compute nodes via the Internet or telephone.
```

NOTE

Retail product keys for Office Professional Plus 2013 cannot be used with this VM image. If you have valid keys and installation media for Office or Excel editions other than this Office Professional Plus 2013 volume edition, you can use them instead. First uninstall this volume edition and install the edition that you have. The reinstalled Excel compute node can be captured as a customized VM image to use in a deployment at scale.

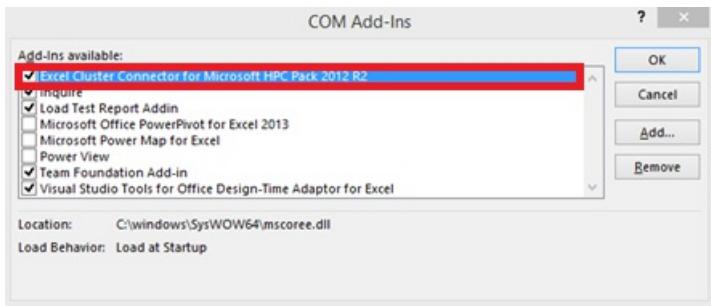
Offload Excel workbooks

Follow these steps to offload an Excel workbook so that it runs on the HPC Pack cluster in Azure. To do this, you must have Excel 2010 or 2013 already installed on the client computer.

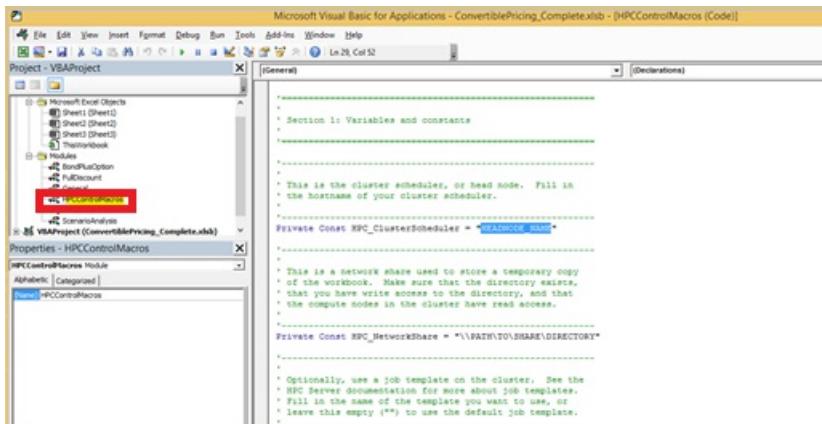
1. Use one of the options in Step 1 to deploy an HPC Pack cluster with the Excel compute node image. Obtain the cluster certificate file (.cer) and cluster username and password.
2. On the client computer, import the cluster certificate under Cert:\CurrentUser\Root.
3. Make sure Excel is installed. Create an Excel.exe.config file with the following contents in the same folder as Excel.exe on the client computer. This step ensures that the HPC Pack 2012 R2 Excel COM add-in loads successfully.

```
<?xml version="1.0"?>
<configuration>
    <startup useLegacyV2RuntimeActivationPolicy="true">
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/>
    </startup>
</configuration>
```

4. Set up the client to submit jobs to the HPC Pack cluster. One option is to download the full [HPC Pack 2012 R2 Update 3 installation](#) and install the HPC Pack client. Alternatively, download and install the [HPC Pack 2012 R2 Update 3 client utilities](#) and the appropriate Visual C++ 2010 redistributable for your computer ([x64](#), [x86](#)).
5. In this example, we use a sample Excel workbook named ConvertiblePricing_Complete.xlsx. You can download it [here](#).
6. Copy the Excel workbook to a working folder such as D:\Excel\Run.
7. Open the Excel workbook. On the **Develop** ribbon, click **COM Add-Ins** and confirm that the HPC Pack Excel COM add-in is loaded successfully.



8. Edit the VBA macro HPCControlMacros in Excel by changing the commented lines as shown in the following script. Substitute appropriate values for your environment.



```
'Private Const HPC_ClusterScheduler = "HEADNODE_NAME"
Private Const HPC_ClusterScheduler = "hpc01.eastus.cloudapp.azure.com"

'Private Const HPC_NetworkShare = "\\\PATH\TO\SHARE\DIRECTORY"
Private Const HPC_DependFiles =
"D:\Excel\Upload\ConvertiblePricing_Complete.xlsb=ConvertiblePricing_Complete.xlsb"

'HPCExcelClient.Initialize ActiveWorkbook
HPCExcelClient.Initialize ActiveWorkbook, HPC_DependFiles

'HPCWorkbookPath = HPC_NetworkShare & Application.PathSeparator & ActiveWorkbook.name
HPCWorkbookPath = "ConvertiblePricing_Complete.xlsb"

'HPCExcelClient.OpenSession headNode:=HPC_ClusterScheduler, remoteWorkbookPath:=HPCWorkbookPath
HPCExcelClient.OpenSession headNode:=HPC_ClusterScheduler, remoteWorkbookPath:=HPCWorkbookPath,
UserName:="hpc\azureuser", Password:="<YourPassword>"
```

9. Copy the Excel workbook to an upload directory such as D:\Excel\Upload. This directory is specified in the HPC_DependFiles constant in the VBA macro.
10. To run the workbook on the cluster in Azure, click the **Cluster** button on the worksheet.

Run Excel UDFs

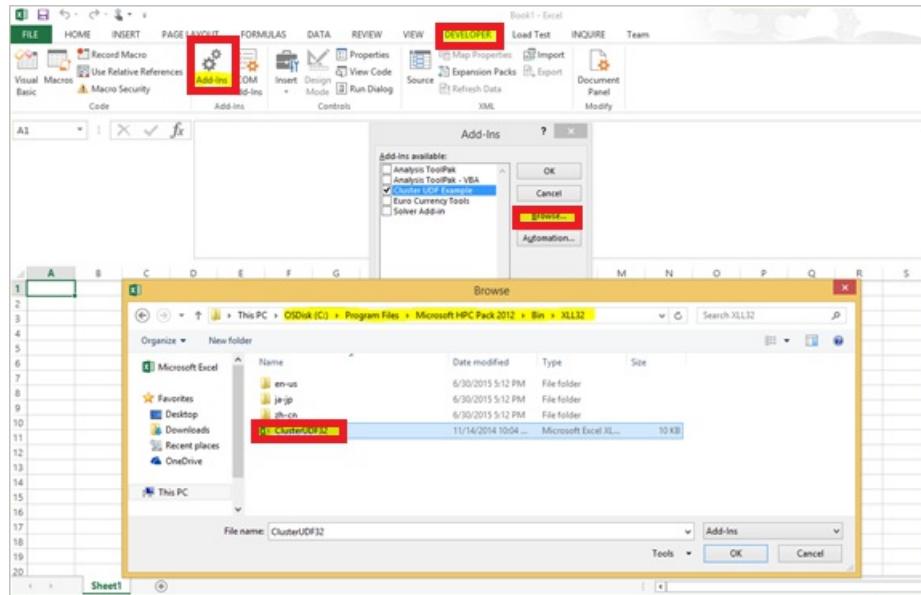
To run Excel UDFs, follow the preceding steps 1 – 3 to set up the client computer. For Excel UDFs, you don't need to have the Excel application installed on compute nodes. So, when creating your cluster compute nodes, you could choose a normal compute node image instead of the compute node image with Excel.

NOTE

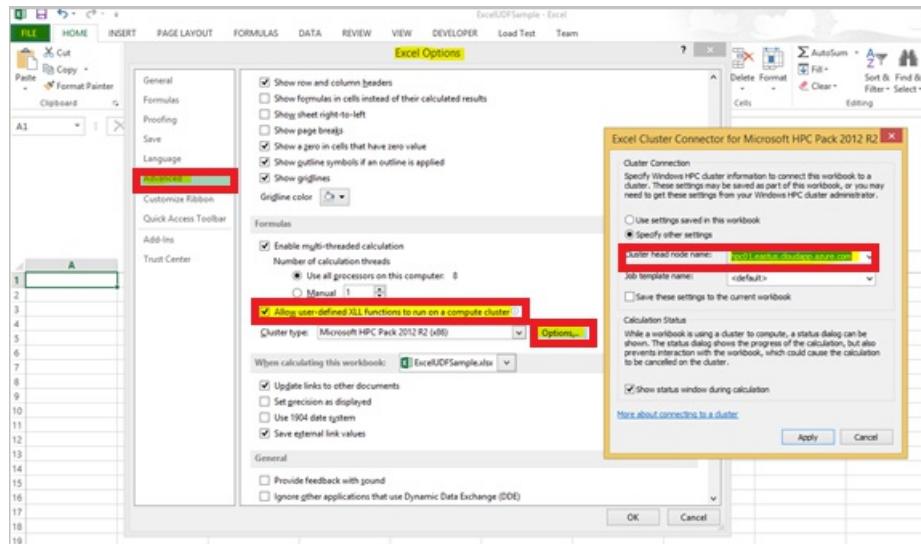
There is a 34 character limit in the Excel 2010 and 2013 cluster connector dialog box. You use this dialog box to specify the cluster that runs the UDFs. If the full cluster name is longer (for example, hpcexcelhn01.southeastasia.cloudapp.azure.com), it does not fit in the dialog box. The workaround is to set a machine-wide variable such as CCP_IAASHN with the value of the long cluster name. Then, enter %CCP_IAASHN% in the dialog box as the cluster head node name.

After the cluster is successfully deployed, continue with the following steps to run a sample built-in Excel UDF. For customized Excel UDFs, see these [resources](#) to build the XLLs and deploy them on the IaaS cluster.

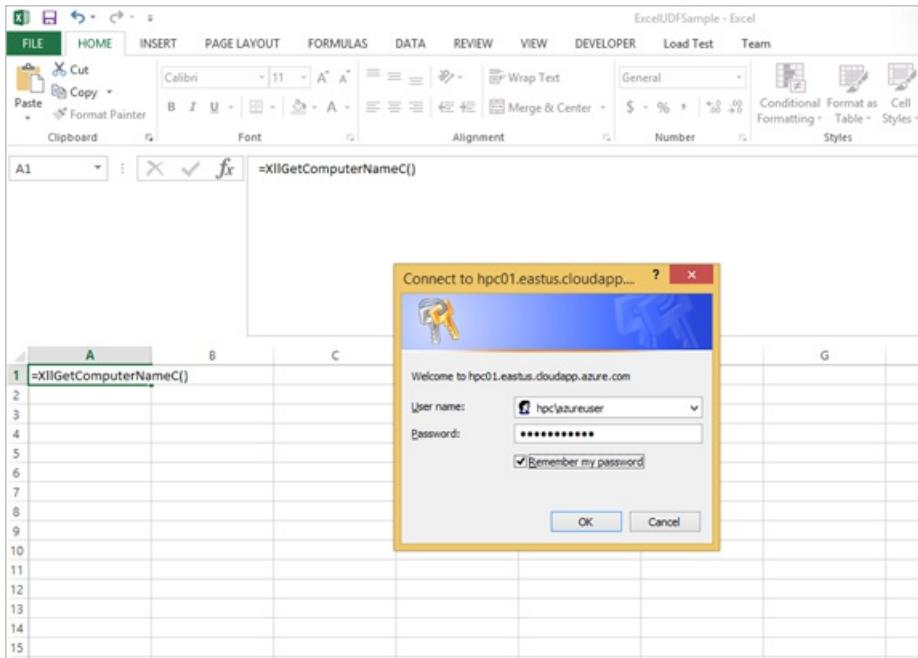
1. Open a new Excel workbook. On the **Developer** ribbon, click **Add-Ins**. Then, in the dialog box, click **Browse**, navigate to the %CCP_HOME%Bin\XLL32 folder, and select the sample ClusterUDF32.xll. If the ClusterUDF32 doesn't exist on the client machine, copy it from the %CCP_HOME%Bin\XLL32 folder on the head node.



2. Click **File > Options > Advanced**. Under **Formulas**, check **Allow user-defined XLL functions to run on a compute cluster**. Then click **Options** and enter the full cluster name in **Cluster head node name**. (As noted previously this input box is limited to 34 characters, so a long cluster name may not fit. You may use a machine-wide variable here for a long cluster name.)



3. To run the UDF calculation on the cluster, click the cell with value =XllGetComputerNameC() and press Enter. The function simply retrieves the name of the compute node on which the UDF runs. For the first run, a credentials dialog box prompts for the username and password to connect to the IaaS cluster.



When there are many cells to calculate, press Alt-Shift-Ctrl + F9 to run the calculation on all cells.

Step 3. Run a SOA workload from an on-premises client

To run general SOA applications on the HPC Pack IaaS cluster, first use one of the methods in Step 1 to deploy the cluster. Specify a generic compute node image in this case, because you do not need Excel on the compute nodes. Then follow these steps.

1. After retrieving the cluster certificate, import it on the client computer under Cert:\CurrentUser\Root.
2. Install the [HPC Pack 2012 R2 Update 3 SDK](#) and [HPC Pack 2012 R2 Update 3 client utilities](#). These tools enable you to develop and run SOA client applications.
3. Download the HelloWorldR2 [sample code](#). Open the HelloWorldR2.sln in Visual Studio 2010 or 2012. (This sample is not currently compatible with more recent versions of Visual Studio.)
4. Build the EchoService project first. Then, deploy the service to the IaaS cluster in the same way you deploy to an on-premises cluster. For detailed steps, see the Readme.doc in HelloWorldR2. Modify and build the HellWorldR2 and other projects as described in the following section to generate the SOA client applications that run on an Azure IaaS cluster.

Use Http binding with Azure storage queue

To use Http binding with an Azure storage queue, make a few changes to the sample code.

- Update the cluster name.

```
// Before
const string headnode = "[headnode]";
// After e.g.
const string headnode = "hpc01.eastus.cloudapp.azure.com";
or
const string headnode = "hpc01.cloudapp.net";
```

- Optionally, use the default TransportScheme in SessionStartInfo or explicitly set it to Http.

```
info.TransportScheme = TransportScheme.Http;
```

- Use default binding for the BrokerClient.

```
// Before
using (BrokerClient<IService1> client = new BrokerClient<IService1>(session, binding))
// After
using (BrokerClient<IService1> client = new BrokerClient<IService1>(session))
```

Or set explicitly using the basicHttpBinding.

```
BasicHttpBinding binding = new BasicHttpBinding(BasicHttpSecurityMode.TransportWithMessageCredential);
binding.Security.Message.ClientCredentialType = BasicHttpMessageCredentialType.UserName;
binding.Security.Transport.ClientCredentialType = HttpClientCredentialType.None;
```

- Optionally, set the UseAzureQueue flag to true in SessionStartInfo. If not set, it will be set to true by default when the cluster name has Azure domain suffixes and the TransportScheme is Http.

```
info.UseAzureQueue = true;
```

Use Http binding without Azure storage queue

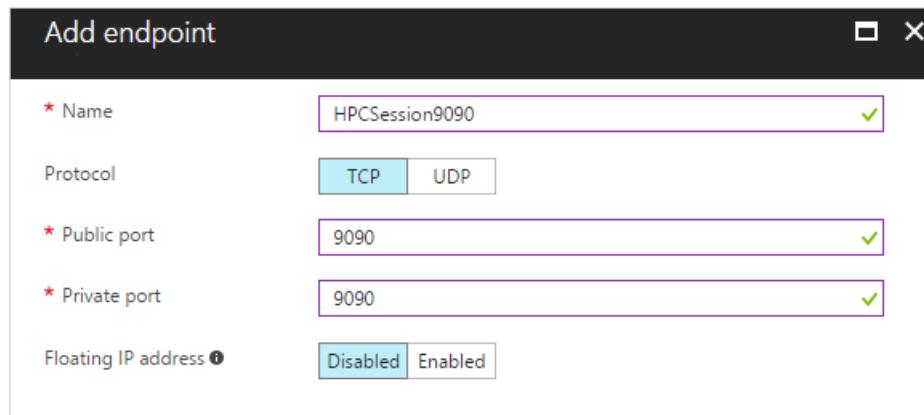
To use Http binding without an Azure storage queue, explicitly set the UseAzureQueue flag to false in the SessionStartInfo.

```
info.UseAzureQueue = false;
```

Use NetTcp binding

To use NetTcp binding, the configuration is similar to connecting to an on-premises cluster. You need to open a few endpoints on the head node VM. If you used the HPC Pack IaaS deployment script to create the cluster, for example, set the endpoints in the Azure portal as follows.

1. Stop the VM.
2. Add the TCP ports 9090, 9087, 9091, 9094 for the Session, Broker, Broker worker, and Data services, respectively



3. Start the VM.

The SOA client application requires no changes except altering the head name to the IaaS cluster full name.

Next steps

- See [these resources](#) for more information about running Excel workloads with HPC Pack.
- See [Managing SOA Services in Microsoft HPC Pack](#) for more about deploying and managing SOA services with HPC Pack.

Frequently asked questions about Azure IaaS VM disks and managed and unmanaged premium disks

8/21/2017 • 10 min to read • [Edit Online](#)

This article answers some frequently asked questions about Azure Managed Disks and Azure Premium Storage.

Managed Disks

What is Azure Managed Disks?

Managed Disks is a feature that simplifies disk management for Azure IaaS VMs by handling storage account management for you. For more information, see the [Managed Disks overview](#).

If I create a standard managed disk from an existing VHD that's 80 GB, how much will that cost me?

A standard managed disk created from an 80-GB VHD is treated as the next available standard disk size, which is an S10 disk. You're charged according to the S10 disk pricing. For more information, see the [pricing page](#).

Are there any transaction costs for standard managed disks?

Yes. You're charged for each transaction. For more information, see the [pricing page](#).

For a standard managed disk, will I be charged for the actual size of the data on the disk or for the provisioned capacity of the disk?

You're charged based on the provisioned capacity of the disk. For more information, see the [pricing page](#).

How is pricing of premium managed disks different from unmanaged disks?

The pricing of premium managed disks is the same as unmanaged premium disks.

Can I change the storage account type (Standard or Premium) of my managed disks?

Yes. You can change the storage account type of your managed disks by using the Azure portal, PowerShell, or the Azure CLI.

Is there a way that I can copy or export a managed disk to a private storage account?

Yes. You can export your managed disks by using the Azure portal, PowerShell, or the Azure CLI.

Can I use a VHD file in an Azure storage account to create a managed disk with a different subscription?

No.

Can I use a VHD file in an Azure storage account to create a managed disk in a different region?

No.

Are there any scale limitations for customers that use managed disks?

Managed Disks eliminates the limits associated with storage accounts. However, the number of managed disks per subscription is limited to 2,000 by default. You can call support to increase this number.

Can I take an incremental snapshot of a managed disk?

No. The current snapshot capability makes a full copy of a managed disk. However, we are planning to support incremental snapshots in the future.

Can VMs in an availability set consist of a combination of managed and unmanaged disks?

No. The VMs in an availability set must use either all managed disks or all unmanaged disks. When you create an availability set, you can choose which type of disks you want to use.

Is Managed Disks the default option in the Azure portal?

Yes.

Can I create an empty managed disk?

Yes. You can create an empty disk. A managed disk can be created independently of a VM, for example, without attaching it to a VM.

What is the supported fault domain count for an availability set that uses Managed Disks?

Depending on the region where the availability set that uses Managed Disks is located, the supported fault domain count is 2 or 3.

How is the standard storage account for diagnostics set up?

You set up a private storage account for VM diagnostics. In the future, we plan to switch diagnostics to Managed Disks as well.

What kind of Role-Based Access Control support is available for Managed Disks?

Managed Disks supports three key default roles:

- Owner: Can manage everything, including access
- Contributor: Can manage everything except access
- Reader: Can view everything, but can't make changes

Is there a way that I can copy or export a managed disk to a private storage account?

You can get a read-only shared access signature URI for the managed disk and use it to copy the contents to a private storage account or on-premises storage.

Can I create a copy of my managed disk?

Customers can take a snapshot of their managed disks and then use the snapshot to create another managed disk.

Are unmanaged disks still supported?

Yes. We support unmanaged and managed disks. We recommend that you use managed disks for new workloads and migrate your current workloads to managed disks.

If I create a 128-GB disk and then increase the size to 130 GB, will I be charged for the next disk size (512 GB)?

Yes.

Can I create locally redundant storage, geo-redundant storage, and zone-redundant storage managed disks?

Azure Managed Disks currently supports only locally redundant storage managed disks.

Can I shrink or downsize my managed disks?

No. This feature is not supported currently.

Can I change the computer name property when a specialized (not created by using the System Preparation tool or generalized) operating system disk is used to provision a VM?

No. You can't update the computer name property. The new VM inherits it from the parent VM, which was used to create the operating system disk.

Where can I find sample Azure Resource Manager templates to create VMs with managed disks?

- [List of templates using Managed Disks](#)
- <https://github.com/chagarw/MDPP>

Migrate to Managed Disks

What changes are required in a pre-existing Azure Backup service configuration prior/after migration to Managed Disks?

No changes are required.

Will my VM backups created via Azure Backup service before the migration continue to work?

Yes, backups work seamlessly.

What changes are required in a pre-existing Azure Disks Encryption configuration prior/after migration to Managed Disks?

No changes are required.

Is automated migration of an existing VM Scale Sets (VMSS) from unmanaged disks to Managed Disks supported?

No. You can create a new VMSS with Managed Disks using the image from your old VMSS with unmanaged disks.

Can I create a Managed Disk from a page blob snapshot taken before migrating to Managed Disks?

No. You can export a page blob snapshot as a page blob and then create a Managed Disk from the exported page blob.

Can I fail over my on-premises machines protected by Azure Site Recovery to a VM with Managed Disks?

Yes, you can choose to failover to a VM with Managed Disks.

Is there any impact of migration on Azure VMs protected by Azure Site Recovery (ASR) via Azure to Azure replication?

Yes. ASR Azure to Azure protection is not supported for VMs with Managed Disks. It is going to be supported by the end of Q1 CY2018.

Can I migrate VMs with unmanaged disks that are located on storage accounts that are or were previously encrypted to managed disks?

Yes

Managed Disks and Storage Service Encryption

Is Azure Storage Service Encryption enabled by default when I create a managed disk?

Yes.

Who manages the encryption keys?

Microsoft manages the encryption keys.

Can I disable Storage Service Encryption for my managed disks?

No.

Is Storage Service Encryption only available in specific regions?

No. It's available in all the regions where Managed Disks is available. Managed Disks is available in all public regions and Germany.

How can I find out if my managed disk is encrypted?

You can find out the time when a managed disk was created from the Azure portal, the Azure CLI, and PowerShell. If the time is after June 9, 2017, then your disk is encrypted.

How can I encrypt my existing disks that were created before June 10, 2017?

As of June 10, 2017, new data written to existing managed disks is automatically encrypted. We are also planning to encrypt existing data, and the encryption will happen asynchronously in the background. If you must encrypt existing data now, create a copy of your disk. New disks will be encrypted.

- [Copy managed disks by using the Azure CLI](#)
- [Copy managed disks by using PowerShell](#)

Are managed snapshots and images encrypted?

Yes. All managed snapshots and images created after June 9, 2017, are automatically encrypted.

Can I convert VMs with unmanaged disks that are located on storage accounts that are or were previously encrypted to managed disks?

Yes

Will an exported VHD from a managed disk or a snapshot also be encrypted?

No. But if you export a VHD to an encrypted storage account from an encrypted managed disk or snapshot, then it's encrypted.

Premium disks: Managed and unmanaged

If a VM uses a size series that supports Premium Storage, such as a DSv2, can I attach both premium and standard data disks?

Yes.

Can I attach both premium and standard data disks to a size series that doesn't support Premium Storage, such as D, Dv2, G, or F series?

No. You can attach only standard data disks to VMs that don't use a size series that supports Premium Storage.

If I create a premium data disk from an existing VHD that was 80 GB, how much will that cost?

A premium data disk created from an 80-GB VHD is treated as the next-available premium disk size, which is a P10 disk. You're charged according to the P10 disk pricing.

Are there transaction costs to use Premium Storage?

There is a fixed cost for each disk size, which comes provisioned with specific limits on IOPS and throughput. The other costs are outbound bandwidth and snapshot capacity, if applicable. For more information, see the [pricing page](#).

What are the limits for IOPS and throughput that I can get from the disk cache?

The combined limits for cache and local SSD for a DS series are 4,000 IOPS per core and 33 MB per second per core. The GS series offers 5,000 IOPS per core and 50 MB per second per core.

Is the local SSD supported for a Managed Disks VM?

The local SSD is temporary storage that is included with a Managed Disks VM. There is no extra cost for this temporary storage. We recommend that you do not use this local SSD to store your application data because it isn't persisted in Azure Blob storage.

Are there any repercussions for the use of TRIM on premium disks?

There is no downside to the use of TRIM on Azure disks on either premium or standard disks.

New disk sizes: Managed and unmanaged

What is the largest disk size supported for operating system and data disks?

The partition type that Azure supports for an operating system disk is the master boot record (MBR). The MBR format supports a disk size up to 2 TB. The largest size that Azure supports for an operating system disk is 2 TB. Azure supports up to 4 TB for data disks.

What is the largest page blob size that's supported?

The largest page blob size that Azure supports is 8 TB (8,191 GB). We don't support page blobs larger than 4 TB (4,095 GB) attached to a VM as data or operating system disks.

Do I need to use a new version of Azure tools to create, attach, resize, and upload disks larger than 1 TB?

You don't need to upgrade your existing Azure tools to create, attach, or resize disks larger than 1 TB. To upload your VHD file from on-premises directly to Azure as a page blob or unmanaged disk, you need to use the latest tool sets:

AZURE TOOLS	SUPPORTED VERSIONS
Azure PowerShell	Version number 4.1.0: June 2017 release or later
Azure CLI v1	Version number 0.10.13: May 2017 release or later
AzCopy	Version number 6.1.0: June 2017 release or later

The support for Azure CLI v2 and Azure Storage Explorer is coming soon.

Are P4 and P6 disk sizes supported for unmanaged disks or page blobs?

No. P4 (32 GB) and P6 (64 GB) disk sizes are supported only for managed disks. Support for unmanaged disks and page blobs is coming soon.

If my existing premium managed disk less than 64 GB was created before the small disk was enabled (around June 15, 2017), how is it billed?

Existing small premium disks less than 64 GB continue to be billed according to the P10 pricing tier.

How can I switch the disk tier of small premium disks less than 64 GB from P10 to P4 or P6?

You can take a snapshot of your small disks and then create a disk to automatically switch the pricing tier to P4 or P6 based on the provisioned size.

What if my question isn't answered here?

If your question isn't listed here, let us know and we'll help you find an answer. You can post a question at the end of this article in the comments. To engage with the Azure Storage team and other community members about this article, use the MSDN [Azure Storage forum](#).

To request features, submit your requests and ideas to the [Azure Storage feedback forum](#).

Attach a data disk to a Windows VM using PowerShell

11/1/2017 • 3 min to read • [Edit Online](#)

This article shows you how to attach both new and existing disks to a Windows virtual machine using PowerShell.

Before you do this, review these tips:

- The size of the virtual machine controls how many data disks you can attach. For details, see [Sizes for virtual machines](#).
- To use Premium storage, you'll need a Premium Storage enabled VM size like the DS-series or GS-series virtual machine. For details, see [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#).

Launch Azure Cloud Shell

The Azure Cloud Shell is a free interactive shell that you can use to run the steps in this article. It has common Azure tools preinstalled and configured to use with your account. Just click the **Copy** to copy the code, paste it into the Cloud Shell, and then press enter to run it. There are two ways to launch the Cloud Shell:

Click Try It in the upper right corner of a code block.	
Click the Cloud Shell button on the menu in the upper right of the Azure portal.	

If you choose to install and use the PowerShell locally, this tutorial requires the Azure PowerShell module version 3.6 or later. Run `Get-Module -ListAvailable AzureRM` to find the version. If you need to upgrade, see [Install Azure PowerShell module](#). If you are running PowerShell locally, you also need to run `Login-AzureRmAccount` to create a connection with Azure.

Add an empty data disk to a virtual machine

This example shows how to add an empty data disk to an existing virtual machine.

Using managed disks

```

$rgName = 'myResourceGroup'
$vmName = 'myVM'
$location = 'East US'
$storageType = 'PremiumLRS'
$dataDiskName = $vmName + '_datadisk1'

$diskConfig = New-AzureRmDiskConfig -AccountType $storageType -Location $location -CreateOption Empty -
DiskSizeGB 128
$dataDisk1 = New-AzureRmDisk -DiskName $dataDiskName -Disk $diskConfig -ResourceGroupName $rgName

$vm = Get-AzureRmVM -Name $vmName -ResourceGroupName $rgName
$vm = Add-AzureRmVMDataDisk -VM $vm -Name $dataDiskName -CreateOption Attach -ManagedDiskId $dataDisk1.Id -Lun
1

Update-AzureRmVM -VM $vm -ResourceGroupName $rgName

```

Using managed disks in an Availability Zone

To create a disk in an Availability Zone, use [New-AzureRmDiskConfig](#) with the `-Zone` parameter. The following example creates a disk in zone 1.

NOTE

Azure Availability Zones is in preview and is ready for your development and test scenarios. Support is available for select Azure resources, regions, and size families. For more information on how to get started, and which Azure resources, regions, and size families you can try with Availability Zones, see [Overview of Availability Zones](#). You can [provide feedback](#) on the Azure website. For support, contact [StackOverflow](#) or [open an Azure support ticket](#).

```

$rgName = 'myResourceGroup'
$vmName = 'myVM'
$location = 'East US 2'
$storageType = 'PremiumLRS'
$dataDiskName = $vmName + '_datadisk1'

$diskConfig = New-AzureRmDiskConfig -AccountType $storageType -Location $location -CreateOption Empty -
DiskSizeGB 128 -Zone 1
$dataDisk1 = New-AzureRmDisk -DiskName $dataDiskName -Disk $diskConfig -ResourceGroupName $rgName

$vm = Get-AzureRmVM -Name $vmName -ResourceGroupName $rgName
$vm = Add-AzureRmVMDataDisk -VM $vm -Name $dataDiskName -CreateOption Attach -ManagedDiskId $dataDisk1.Id -Lun
1

Update-AzureRmVM -VM $vm -ResourceGroupName $rgName

```

Initialize the disk

After you add an empty disk, you need to initialize it. To initialize the disk, you can log in to a VM and use disk management. If you enabled WinRM and a certificate on the VM when you created it, you can use remote PowerShell to initialize the disk. You can also use a custom script extension:

```

.setLocation = "location-name"
$scriptName = "script-name"
$fileName = "script-file-name"
Set-AzureRmVMCustomScriptExtension -ResourceGroupName $rgName -Location $locName -VMName $vmName -Name
$scriptName -TypeHandlerVersion "1.4" -StorageAccountName "mystore1" -StorageAccountKey "primary-key" -
FileName $fileName -ContainerName "scripts"

```

The script file can contain something like this code to initialize the disks:

```
$disks = Get-Disk | Where partitionstyle -eq 'raw' | sort number

$letters = 70..89 | ForEach-Object { [char]$_.ToString() }
$count = 0
$labels = "data1","data2"

foreach ($disk in $disks) {
    $driveLetter = $letters[$count].ToString()
    $disk |
        Initialize-Disk -PartitionStyle MBR -PassThru |
        New-Partition -UseMaximumSize -DriveLetter $driveLetter |
        Format-Volume -FileSystem NTFS -NewFileSystemLabel $labels[$count] -Confirm:$false -Force
    $count++
}
```

Attach an existing data disk to a VM

You can attach an existing managed disk to a VM as a data disk.

```
$rgName = "myResourceGroup"
$vmName = "myVM"
$location = "East US"
$dataDiskName = "myDisk"
$disk = Get-AzureRmDisk -ResourceGroupName $rgName -DiskName $dataDiskName

$vm = Get-AzureRmVM -Name $vmName -ResourceGroupName $rgName

$vm = Add-AzureRmVMDataDisk -CreateOption Attach -Lun 0 -VM $vm -ManagedDiskId $disk.Id

Update-AzureRmVM -VM $vm -ResourceGroupName $rgName
```

Next steps

Create a [snapshot](#).

How to attach a managed data disk to a Windows VM in the Azure portal

12/14/2017 • 2 min to read • [Edit Online](#)

This article shows you how to attach a new managed data disk to Windows virtual machines through the Azure portal. Before you do this, review these tips:

- The size of the virtual machine controls how many data disks you can attach. For details, see [Sizes for virtual machines](#).
- For a new disk, you don't need to create it first because Azure creates it when you attach it.

You can also [attach a data disk using Powershell](#).

Add a data disk

1. In the menu on the left, click **Virtual Machines**.
2. Select the virtual machine from the list.
3. On the virtual machine page, click **Disk**s.
4. On the **Disk**s page, click **+ Add data disk**.
5. In the drop-down for the new disk, select **Create disk**.
6. In the **Create managed disk** page, type in a name for the disk and adjust the other settings as necessary.
When you are done, click **Create**.
7. In the **Disk**s page, click **Save** to save the new disk configuration for the VM.
8. After Azure creates the disk and attaches it to the virtual machine, the new disk is listed in the virtual machine's disk settings under **Data disks**.

Initialize a new data disk

1. Connect to the VM.
2. Click the start menu inside the VM and type **diskmgmt.msc** and hit **Enter**. Disk Management snap-in opens.
3. Disk Management recognizes that you have a new, un-initialized disk and the **Initialize Disk** window pops up.
4. Make sure the new disk is selected and click **OK** to initialize it.
5. The new disk appears as **unallocated**. Right-click anywhere on the disk and select **New simple volume**. The **New Simple Volume Wizard** opens.
6. Go through the wizard, keeping all of the defaults, when you are done select **Finish**.
7. Close Disk Management.
8. You get a pop-up that you need to format the new disk before you can use it. Click **Format disk**.
9. In the **Format new disk** dialog, check the settings and then click **Start**.
10. You get a warning that formatting the disks erases all of the data, click **OK**.
11. When the format is complete, click **OK**.

Use TRIM with standard storage

If you use standard storage (HDD), you should enable TRIM. TRIM discards unused blocks on the disk so you are only billed for storage that you are actually using. This can save on costs if you create large files and then delete them.

You can run this command to check the TRIM setting. Open a command prompt on your Windows VM and type:

```
fsutil behavior query DisableDeleteNotify
```

If the command returns 0, TRIM is enabled correctly. If it returns 1, run the following command to enable TRIM:

```
fsutil behavior set DisableDeleteNotify 0
```

After deleting data from your disk, you can ensure the TRIM operations flush properly by running defrag with TRIM:

```
defrag.exe <volume:> -l
```

You can also ensure the entire volume is trimmed by formatting the volume.

Next steps

If your application needs to use the D: drive to store data, you can [change the drive letter of the Windows temporary disk](#).

How to detach a data disk from a Windows virtual machine

11/20/2017 • 1 min to read • [Edit Online](#)

When you no longer need a data disk that's attached to a virtual machine, you can easily detach it. This removes the disk from the virtual machine, but doesn't remove it from storage.

WARNING

If you detach a disk it is not automatically deleted. If you have subscribed to Premium storage, you will continue to incur storage charges for the disk. For more information refer to [Pricing and Billing when using Premium Storage](#).

If you want to use the existing data on the disk again, you can reattach it to the same virtual machine, or another one.

Detach a data disk using the portal

1. In the left menu, select **Virtual Machines**.
2. Select the virtual machine that has the data disk you want to detach and click **Stop** to deallocate the VM.
3. In the virtual machine pane, select **Disks**.
4. At the top of the **Disks** pane, select **Edit**.
5. In the **Disk** pane, to the far right of the data disk that you would like to detach, click the detach button.
6. After the disk has been removed, click **Save** on the top of the pane.
7. In the virtual machine pane, click **Overview** and then click the **Start** button at the top of the pane to restart the VM.

The disk remains in storage but is no longer attached to a virtual machine.

Detach a data disk using PowerShell

In this example, the first command gets the virtual machine named **MyVM07** in the **RG11** resource group using the [Get-AzureRmVM](#) cmdlet and stores it in the **\$VirtualMachine** variable.

The second line removes the data disk named **DataDisk3** from the virtual machine using the [Remove-AzureRmVMDataDisk](#) cmdlet.

The third line updates the state of the virtual machine, using the [Update-AzureRmVM](#) cmdlet, to complete the process of removing the data disk.

```
$VirtualMachine = Get-AzureRmVM -ResourceGroupName "RG11" -Name "MyVM07"
Remove-AzureRmVMDataDisk -VM $VirtualMachine -Name "DataDisk3"
Update-AzureRmVM -ResourceGroupName "RG11" -VM $VirtualMachine
```

For more information, see [Remove-AzureRmVMDataDisk](#).

Next steps

If you want to reuse the data disk, you can just [attach it to another VM](#)

How to expand the OS drive of a Virtual Machine in an Azure Resource Group

6/27/2017 • 2 min to read • [Edit Online](#)

Overview

When you create a new virtual machine (VM) in a Resource Group by deploying an image from [Azure Marketplace](#), the default OS drive is often 127 GB (some images have smaller OS disk sizes by default). Even though it's possible to add data disks to the VM (how many depending upon the SKU you've chosen) and moreover it's recommended to install applications and CPU intensive workloads on these addendum disks, oftentimes customers need to expand the OS drive to support certain scenarios such as following:

1. Support legacy applications that install components on OS drive.
2. Migrate a physical PC or virtual machine from on-premises with a larger OS drive.

IMPORTANT

Azure has two different deployment models for creating and working with resources: Resource Manager and Classic. This article covers using the Resource Manager model. Microsoft recommends that most new deployments use the Resource Manager model.

Resize the OS drive

In this article we'll accomplish the task of resizing the OS drive using resource manager modules of [Azure Powershell](#). Open your Powershell ISE or Powershell window in administrative mode and follow the steps below:

1. Sign-in to your Microsoft Azure account in resource management mode and select your subscription as follows:

```
Login-AzureRmAccount  
Select-AzureRmSubscription -SubscriptionName 'my-subscription-name'
```

2. Set your resource group name and VM name as follows:

```
$rgName = 'my-resource-group-name'  
$vmName = 'my-vm-name'
```

3. Obtain a reference to your VM as follows:

```
$vm = Get-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

4. Stop the VM before resizing the disk as follows:

```
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

5. And here comes the moment we've been waiting for! Set the size of the OS disk to the desired value and update the VM as follows:

```
$vm.StorageProfile.OSDisk.DiskSizeGB = 1023  
Update-AzureRmVM -ResourceGroupName $rgName -VM $vm
```

WARNING

The new size should be greater than the existing disk size. The maximum allowed is 2048 GB. (It is possible to expand the VHD blob beyond that size, but the OS will only be able to work with the first 2048 GB of space.)

6. Updating the VM may take a few seconds. Once the command finishes executing, restart the VM as follows:

```
Start-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

And that's it! Now RDP into the VM, open Computer Management (or Disk Management) and expand the drive using the newly allocated space.

Summary

In this article, we used Azure Resource Manager modules of Powershell to expand the OS drive of an IaaS virtual machine. Reproduced below is the complete script for your reference:

```
Login-AzureRmAccount  
Select-AzureRmSubscription -SubscriptionName 'my-subscription-name'  
$rgName = 'my-resource-group-name'  
$vmName = 'my-vm-name'  
$vm = Get-AzureRmVM -ResourceGroupName $rgName -Name $vmName  
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName  
$vm.StorageProfile.OSDisk.DiskSizeGB = 1023  
Update-AzureRmVM -ResourceGroupName $rgName -VM $vm  
Start-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

Next Steps

Though in this article, we focused primarily on expanding the OS disk of the VM, the developed script may also be used for expanding the data disks attached to the VM by changing a single line of code. For example, to expand the first data disk attached to the VM, replace the `osDisk` object of `StorageProfile` with `DataDisks` array and use a numeric index to obtain a reference to first attached data disk, as shown below:

```
$vm.StorageProfile.DataDisks[0].DiskSizeGB = 1023
```

Similarly you may reference other data disks attached to the VM, either by using an index as shown above or the `Name` property of the disk as illustrated below:

```
($vm.StorageProfile.DataDisks | Where {$_.Name -eq 'my-second-data-disk'})[0].DiskSizeGB = 1023
```

If you want to find out how to attach disks to an Azure Resource Manager VM, check this [article](#).

Create a snapshot

10/13/2017 • 1 min to read • [Edit Online](#)

Take a snapshot of an OS or data disk VHD for backup or to troubleshoot VM issues. A snapshot is a full, read-only copy of a VHD.

Use Azure portal to take a snapshot

1. Sign in to the [Azure portal](#).
2. Starting in the upper left, click **New** and search for **snapshot**.
3. In the Snapshot blade, click **Create**.
4. Enter a **Name** for the snapshot.
5. Select an existing [Resource group](#) or type the name for a new one.
6. Select an Azure datacenter Location.
7. For **Source disk**, select the Managed Disk to snapshot.
8. Select the **Account type** to use to store the snapshot. We recommend **Standard_LRS** unless you need it stored on a high performing disk.
9. Click **Create**.

Use PowerShell to take a snapshot

The following steps show you how to get the VHD disk to be copied, create the snapshot configurations, and take a snapshot of the disk by using the [New-AzureRmSnapshot cmdlet](#).

Make sure that you have the latest version of the AzureRM.Compute PowerShell module installed. Run the following command to install it.

```
Install-Module AzureRM.Compute -RequiredVersion 2.6.0
```

For more information, see [Azure PowerShell Versioning](#).

1. Set some parameters.

```
$resourceGroupName = 'myResourceGroup'  
$location = 'eastus'  
$dataDiskName = 'myDisk'  
$snapshotName = 'mySnapshot'
```

2. Get the VHD disk to be copied.

```
$disk = Get-AzureRmDisk -ResourceGroupName $resourceGroupName -DiskName $dataDiskName
```

3. Create the snapshot configurations.

```
$snapshot = New-AzureRmSnapshotConfig -SourceUri $disk.Id -CreateOption Copy -Location $location
```

4. Take the snapshot.

```
New-AzureRmSnapshot -Snapshot $snapshot -SnapshotName $snapshotName -ResourceGroupName  
$resourceGroupName
```

If you plan to use the snapshot to create a Managed Disk and attach it to a VM that needs to be high performing, use the parameter `-AccountType Premium_LRS` with the `New-AzureRmSnapshot` command. The parameter creates the snapshot so that it's stored as a Premium Managed Disk. Premium Managed Disks are more expensive than Standard. So be sure you really need Premium before using that parameter.

Next steps

Create a virtual machine from a snapshot by creating a managed disk from a snapshot and then attaching the new managed disk as the OS disk. For more information, see the [Create a VM from a snapshot](#) sample.

Back up Azure unmanaged VM disks with incremental snapshots

8/21/2017 • 7 min to read • [Edit Online](#)

Overview

Azure Storage provides the capability to take snapshots of blobs. Snapshots capture the blob state at that point in time. In this article, we describe a scenario in which you can maintain backups of virtual machine disks using snapshots. You can use this methodology when you choose not to use Azure Backup and Recovery Service, and wish to create a custom backup strategy for your virtual machine disks.

Azure virtual machine disks are stored as page blobs in Azure Storage. Since we are describing a backup strategy for virtual machine disks in this article, we refer to snapshots in the context of page blobs. To learn more about snapshots, refer to [Creating a Snapshot of a Blob](#).

What is a snapshot?

A blob snapshot is a read-only version of a blob that is captured at a point in time. Once a snapshot has been created, it can be read, copied, or deleted, but not modified. Snapshots provide a way to back up a blob as it appears at a moment in time. Until REST version 2015-04-05, you had the ability to copy full snapshots. With the REST version 2015-07-08 and above, you can also copy incremental snapshots.

Full snapshot copy

Snapshots can be copied to another storage account as a blob to keep backups of the base blob. You can also copy a snapshot over its base blob, which is like restoring the blob to an earlier version. When a snapshot is copied from one storage account to another, it occupies the same space as the base page blob. Therefore, copying whole snapshots from one storage account to another is slow and consumes much space in the target storage account.

NOTE

If you copy the base blob to another destination, the snapshots of the blob are not copied along with it. Similarly, if you overwrite a base blob with a copy, snapshots associated with the base blob are not affected and stay intact under the base blob name.

Back up disks using snapshots

As a backup strategy for your virtual machine disks, you can take periodic snapshots of the disk or page blob, and copy them to another storage account using tools like [Copy Blob](#) operation or [AzCopy](#). You can copy a snapshot to a destination page blob with a different name. The resulting destination page blob is a writeable page blob and not a snapshot. Later in this article, we describe steps to take backups of virtual machine disks using snapshots.

Restore disks using snapshots

When it is time to restore your disk to a stable version that was previously captured in one of the backup snapshots, you can copy a snapshot over the base page blob. After the snapshot is promoted to the base page blob, the snapshot remains, but its source is overwritten with a copy that can be both read and written. Later in this article we describe steps to restore a previous version of your disk from its snapshot.

Implementing full snapshot copy

You can implement a full snapshot copy by doing the following,

- First, take a snapshot of the base blob using the [Snapshot Blob](#) operation.
- Then, copy the snapshot to a target storage account using [Copy Blob](#).
- Repeat this process to maintain backup copies of your base blob.

Incremental snapshot copy

The new feature in the [GetPageRanges](#) API provides a much better way to back up the snapshots of your page blobs or disks. The API returns the list of changes between the base blob and the snapshots, which reduces the amount of storage space used on the backup account. The API supports page blobs on Premium Storage as well as Standard Storage. Using this API, you can build faster and more efficient backup solutions for Azure VMs. This API will be available with the REST version 2015-07-08 and higher.

Incremental Snapshot Copy allows you to copy from one storage account to another the difference between,

- Base blob and its Snapshot OR
- Any two snapshots of the base blob

Provided the following conditions are met,

- The blob was created on Jan-1-2016 or later.
- The blob was not overwritten with [PutPage](#) or [Copy Blob](#) between two snapshots.

Note: This feature is available for Premium and Standard Azure Page Blobs.

When you have a custom backup strategy using snapshots, copying the snapshots from one storage account to another can be slow and can consume much storage space. Instead of copying the entire snapshot to a backup storage account, you can write the difference between consecutive snapshots to a backup page blob. This way, the time to copy and the space to store backups is substantially reduced.

Implementing Incremental Snapshot Copy

You can implement incremental snapshot copy by doing the following,

- Take a snapshot of the base blob using [Snapshot Blob](#).
- Copy the snapshot to the target backup storage account using [Copy Blob](#). This is the backup page blob. Take a snapshot of the backup page blob and store it in the backup account.
- Take another snapshot of the base blob using [Snapshot Blob](#).
- Get the difference between the first and second snapshots of the base blob using [GetPageRanges](#). Use the new parameter **prevsnapshot**, to specify the DateTime value of the snapshot you want to get the difference with. When this parameter is present, the REST response includes only the pages that were changed between target snapshot and previous snapshot including clear pages.
- Use [PutPage](#) to apply these changes to the backup page blob.
- Finally, take a snapshot of the backup page blob and store it in the backup storage account.

In the next section, we will describe in more detail how you can maintain backups of disks using Incremental Snapshot Copy

Scenario

In this section, we describe a scenario that involves a custom backup strategy for virtual machine disks using snapshots.

Consider a DS-series Azure VM with a premium storage P30 disk attached. The P30 disk called *mypremiumdisk* is stored in a premium storage account called *mypremiumaccount*. A standard storage account called *mybackupsstdaccount* is used for storing the backup of *mypremiumdisk*. We would like to keep a snapshot of *mypremiumdisk* every 12 hours.

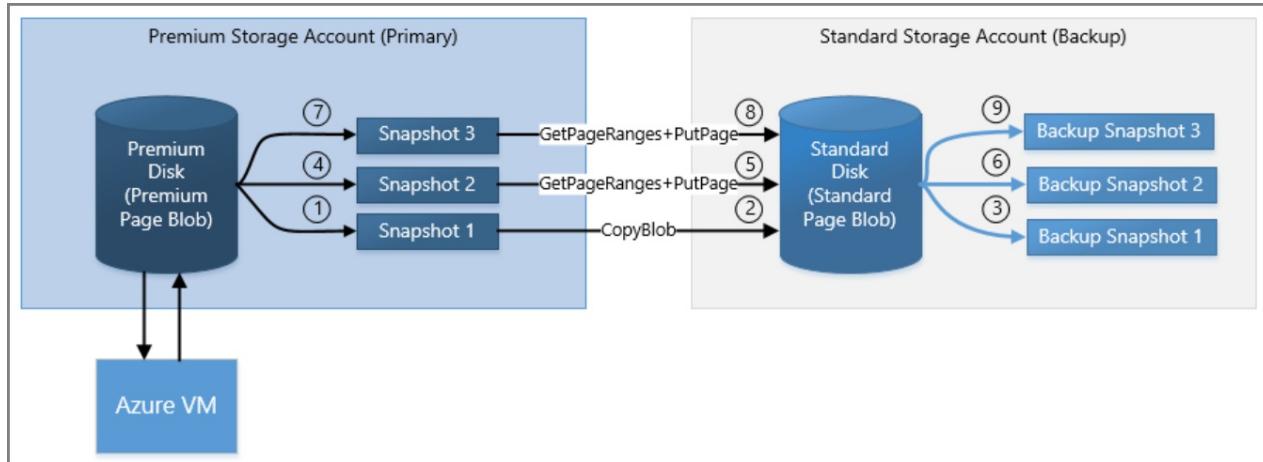
To learn about creating storage account and disks, refer to [About Azure storage accounts](#).

To learn about backing up Azure VMs, refer to [Plan Azure VM backups](#).

Steps to maintain backups of a disk using incremental snapshots

The following steps describe how to take snapshots of *mypremiumdisk* and maintain the backups in *mybackupsstdaccount*. The backup is a standard page blob called *mybackupsstdpageblob*. The backup page blob always reflects the same state as the last snapshot of *mypremiumdisk*.

1. Create the backup page blob for your premium storage disk, by taking a snapshot of *mypremiumdisk* called *mypremiumdisk_ss1*.
2. Copy this snapshot to *mybackupsstdaccount* as a page blob called *mybackupsstdpageblob*.
3. Take a snapshot of *mybackupsstdpageblob* called *mybackupsstdpageblob_ss1*, using [Snapshot Blob](#) and store it in *mybackupsstdaccount*.
4. During the backup window, create another snapshot of *mypremiumdisk*, say *mypremiumdisk_ss2*, and store it in *mypremiumaccount*.
5. Get the incremental changes between the two snapshots, *mypremiumdisk_ss2* and *mypremiumdisk_ss1*, using [GetPageRanges](#) on *mypremiumdisk_ss2* with the **prevsnapshot** parameter set to the timestamp of *mypremiumdisk_ss1*. Write these incremental changes to the backup page blob *mybackupsstdpageblob* in *mybackupsstdaccount*. If there are deleted ranges in the incremental changes, they must be cleared from the backup page blob. Use [PutPage](#) to write incremental changes to the backup page blob.
6. Take a snapshot of the backup page blob *mybackupsstdpageblob*, called *mybackupsstdpageblob_ss2*. Delete the previous snapshot *mypremiumdisk_ss1* from premium storage account.
7. Repeat steps 4-6 every backup window. In this way, you can maintain backups of *mypremiumdisk* in a standard storage account.



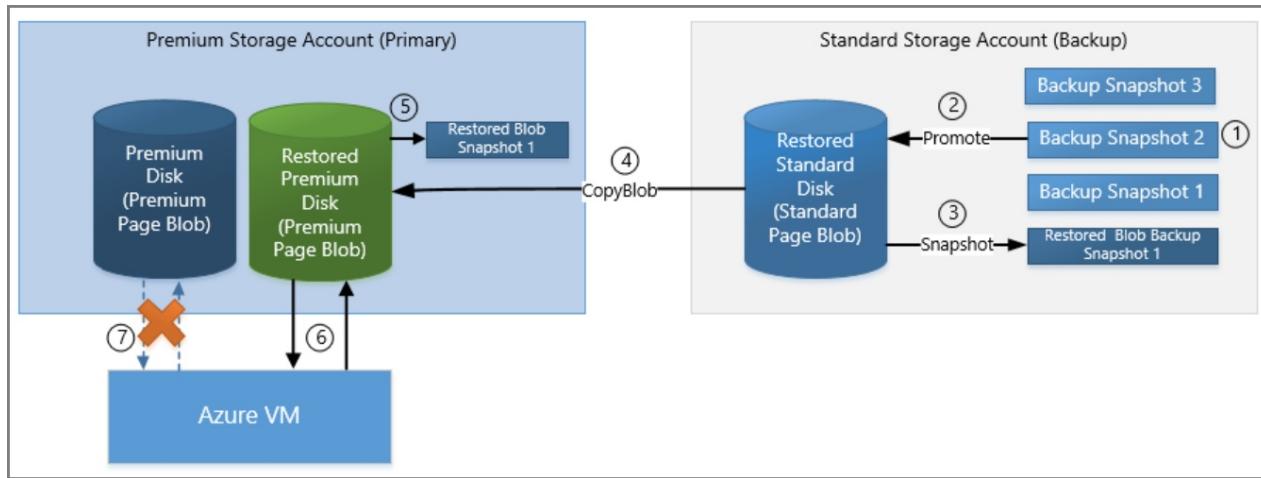
Steps to restore a disk from snapshots

The following steps, describe how to restore the premium disk, *mypremiumdisk* to an earlier snapshot from the backup storage account *mybackupsstdaccount*.

1. Identify the point in time that you wish to restore the premium disk to. Let's say that it is snapshot *mybackupsstdpageblob_ss2*, which is stored in the backup storage account *mybackupsstdaccount*.
2. In *mybackupsstdaccount*, promote the snapshot *mybackupsstdpageblob_ss2* as the new backup base page blob *mybackupsstdpageblobrestored*.
3. Take a snapshot of this restored backup page blob, called *mybackupsstdpageblobrestored_ss1*.
4. Copy the restored page blob *mybackupsstdpageblobrestored* from *mybackupsstdaccount* to *mypremiumaccount* as the new premium disk *mypremiumdiskrestored*.
5. Take a snapshot of *mypremiumdiskrestored*, called *mypremiumdiskrestored_ss1* for making future incremental

backups.

6. Point the DS series VM to the restored disk *mypremiumdiskrestored* and detach the old *mypremiumdisk* from the VM.
7. Begin the Backup process described in previous section for the restored disk *mypremiumdiskrestored*, using the *mybackupstdpageblobrestored* as the backup page blob.



Next Steps

Use the following links to learn more about creating snapshots of a blob and planning your VM backup infrastructure.

- [Creating a Snapshot of a Blob](#)
- [Plan your VM Backup Infrastructure](#)

Convert Azure managed disks storage from standard to premium, and vice versa

11/9/2017 • 3 min to read • [Edit Online](#)

Managed disks offers two storage options: [Premium](#) (SSD-based) and [Standard](#) (HDD-based). It allows you to easily switch between the two options with minimal downtime based on your performance needs. This capability is not available for unmanaged disks. But you can easily [convert to managed disks](#) to easily switch between the two options.

This article shows you how to convert managed disks from standard to premium, and vice versa by using Azure PowerShell. If you need to install or upgrade it, see [Install and configure Azure PowerShell](#).

Before you begin

- The conversion requires a restart of the VM, so schedule the migration of your disks storage during a pre-existing maintenance window.
- If you are using unmanaged disks, first [convert to managed disks](#) to use this article to switch between the two storage options.

Convert all the managed disks of a VM from standard to premium, and vice versa

In the following example, we show how to switch all the disks of a VM from standard to premium storage. To use premium managed disks, your VM must use a [VM size](#) that supports premium storage. This example also switches to a size that supports premium storage.

```

# Name of the resource group that contains the VM
$rgName = 'yourResourceGroup'

# Name of the your virtual machine
$vmName = 'yourVM'

# Choose between StandardLRS and PremiumLRS based on your scenario
$storageType = 'PremiumLRS'

# Premium capable size
# Required only if converting storage from standard to premium
$size = 'Standard_DS2_v2'
$vm = Get-AzureRmVM -Name $vmName -resourceGroupName $rgName

# Stop and deallocate the VM before changing the size
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName -Force

# Change the VM size to a size that supports premium storage
# Skip this step if converting storage from premium to standard
$vm.HardwareProfile.VmSize = $size
Update-AzureRmVM -VM $vm -ResourceGroupName $rgName

# Get all disks in the resource group of the VM
$vmDisks = Get-AzureRmDisk -ResourceGroupName $rgName

# For disks that belong to the selected VM, convert to premium storage
foreach ($disk in $vmDisks)
{
    if ($disk.OwnerId -eq $vm.Id)
    {
        $diskUpdateConfig = New-AzureRmDiskUpdateConfig -AccountType $storageType ` 
        Update-AzureRmDisk -DiskUpdate $diskUpdateConfig -ResourceGroupName $rgName ` 
        -DiskName $disk.Name
    }
}

Start-AzureRmVM -ResourceGroupName $rgName -Name $vmName

```

Convert a managed disk from standard to premium, and vice versa

For your dev/test workload, you may want to have mixture of standard and premium disks to reduce your cost. You can accomplish it by upgrading to premium storage, only the disks that require better performance. In the following example, we show how to switch a single disk of a VM from standard to premium storage, and vice versa. To use premium managed disks, your VM must use a [VM size](#) that supports premium storage. This example also switches to a size that supports premium storage.

```

$diskName = 'yourDiskName'
# resource group that contains the managed disk
$rgName = 'yourResourceGroupName'
# Choose between StandardLRS and PremiumLRS based on your scenario
$storageType = 'PremiumLRS'
# Premium capable size
$size = 'Standard_DS2_v2'

$disk = Get-AzureRmDisk -DiskName $diskName -ResourceGroupName $rgName

# Get the ARM resource to get name and resource group of the VM
$vmResource = Get-AzureRmResource -ResourceId $disk.diskId
$vm = Get-AzureRmVM $vmResource.ResourceGroupName -Name $vmResource.ResourceName

# Stop and deallocate the VM before changing the storage type
Stop-AzureRmVM -ResourceGroupName $vm.ResourceGroupName -Name $vm.Name -Force

# Change the VM size to a size that supports premium storage
# Skip this step if converting storage from premium to standard
$vm.HardwareProfile.VmSize = $size
Update-AzureRmVM -VM $vm -ResourceGroupName $rgName

# Update the storage type
$diskUpdateConfig = New-AzureRmDiskUpdateConfig -AccountType $storageType -DiskSizeGB $disk.DiskSizeGB
Update-AzureRmDisk -DiskUpdate $diskUpdateConfig -ResourceGroupName $rgName `

-DiskName $disk.Name

Start-AzureRmVM -ResourceGroupName $vm.ResourceGroupName -Name $vm.Name

```

Next steps

Take a read-only copy of a VM by using [snapshots](#).

Migrate to Premium Storage by using Azure Site Recovery

12/11/2017 • 11 min to read • [Edit Online](#)

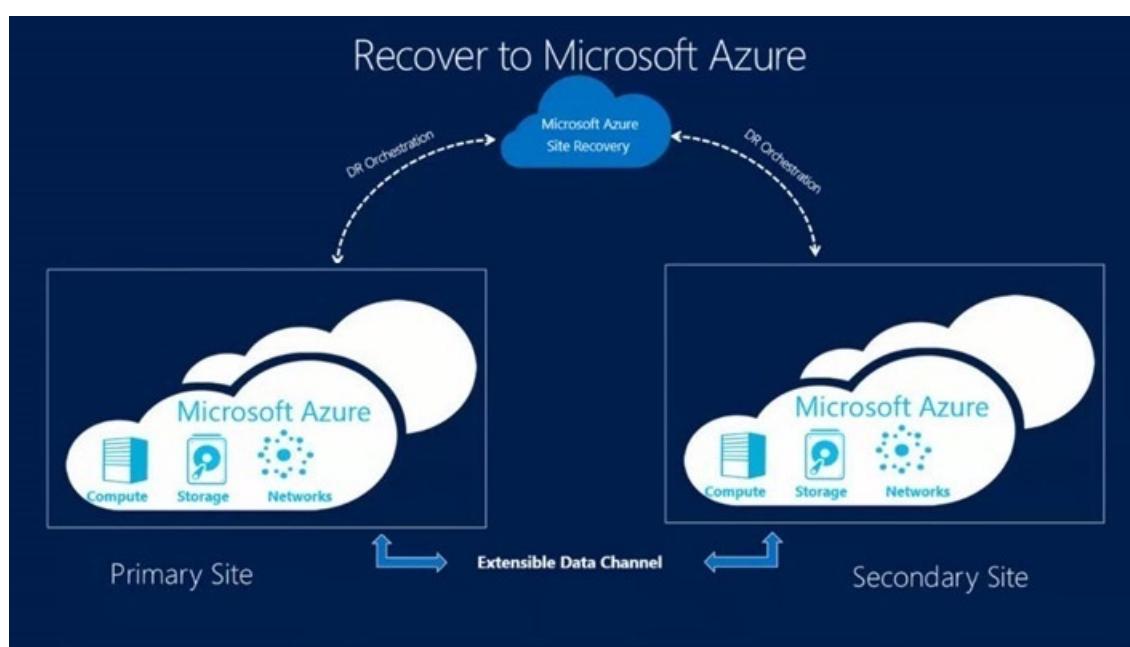
Azure Premium Storage delivers high-performance, low-latency disk support for virtual machines (VMs) that are running I/O-intensive workloads. This guide helps you migrate your VM disks from a standard storage account to a premium storage account by using [Azure Site Recovery](#).

Site Recovery is an Azure service that contributes to your strategy for business continuity and disaster recovery by orchestrating the replication of on-premises physical servers and VMs to the cloud (Azure) or to a secondary datacenter. When outages occur in your primary location, you fail over to the secondary location to keep applications and workloads available. You fail back to your primary location when it returns to normal operation.

Site Recovery provides test failovers to support disaster recovery drills without affecting production environments. You can run failovers with minimal data loss (depending on replication frequency) for unexpected disasters. In the scenario of migrating to Premium Storage, you can use the [failover in Site Recovery](#) to migrate target disks to a premium storage account.

We recommend migrating to Premium Storage by using Site Recovery because this option provides minimal downtime. This option also avoids the manual execution of copying disks and creating new VMs. Site Recovery will systematically copy your disks and create new VMs during failover.

Site Recovery supports a number of types of failover with minimal or no downtime. To plan your downtime and estimate data loss, see the [types of failover in Site Recovery](#). If you [prepare to connect to Azure VMs after failover](#), you should be able to connect to the Azure VM by using RDP after failover.



Azure Site Recovery components

These Site Recovery components are relevant to this migration scenario:

- **Configuration server** is an Azure VM that coordinates communication and manages data replication and recovery processes. On this VM, you run a single setup file to install the configuration server and an additional component, called a process server, as a replication gateway. Read about [configuration server](#)

[prerequisites](#). You set up the configuration server only once, and you can use it for all migrations to the same region.

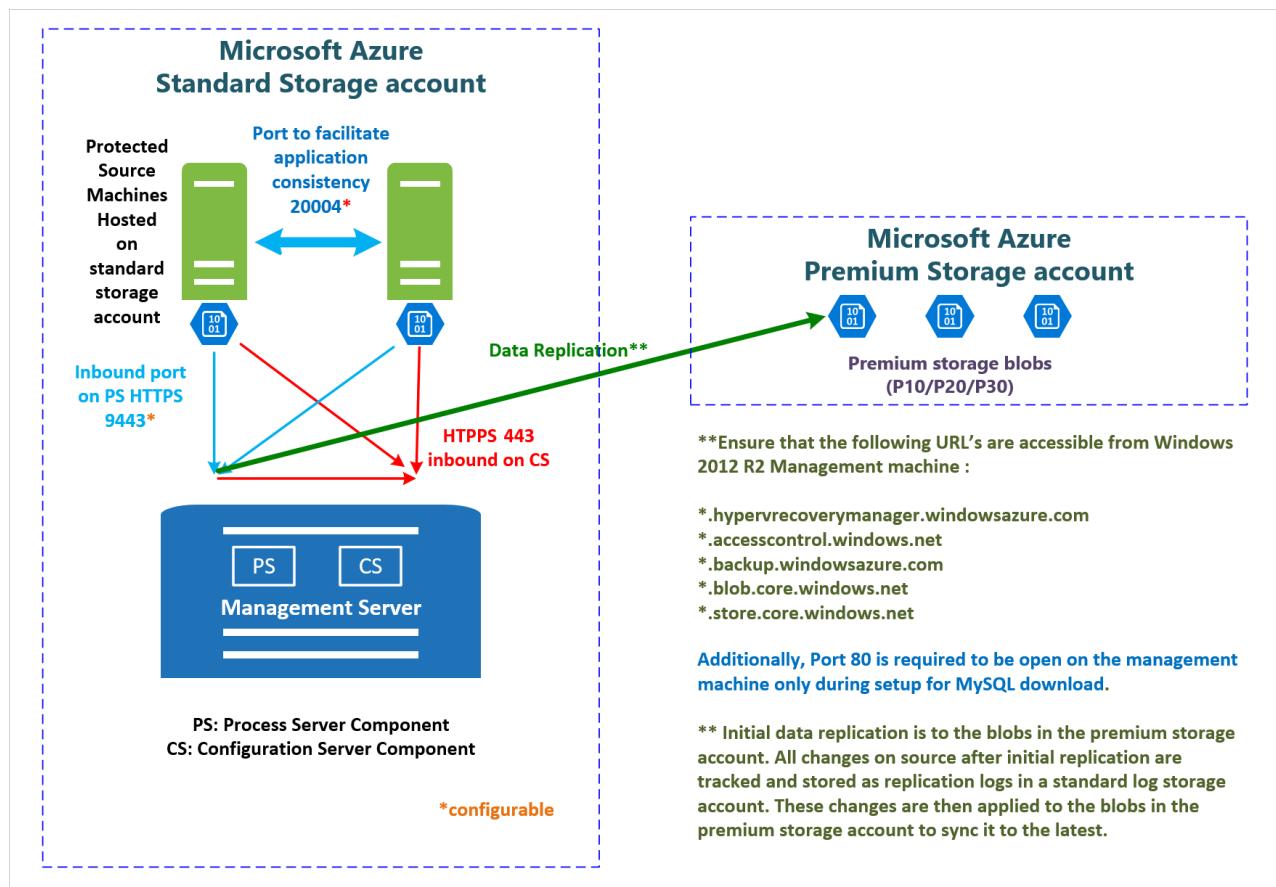
- **Process server** is a replication gateway that:

1. Receives replication data from source VMs.
2. Optimizes the data with caching, compression, and encryption.
3. Sends the data to a storage account.

It also handles push installation of the mobility service to source VMs and performs automatic discovery of source VMs. The default process server is installed on the configuration server. You can deploy additional standalone process servers to scale your deployment. Read about [best practices for process server deployment](#) and [deploying additional process servers](#). You set up the process server only once, and you can use it for all migrations to the same region.

- **Mobility service** is a component that is deployed on every standard VM that you want to replicate. It captures data writes on the standard VM and forwards them to the process server. Read about [replicated machine prerequisites](#).

This graphic shows how these components interact:



NOTE

Site Recovery does not support the migration of Storage Spaces disks.

For additional components for other scenarios, see [Scenario architecture](#).

Azure essentials

These are the Azure requirements for this migration scenario:

- An Azure subscription.

- An Azure premium storage account to store replicated data.
- An Azure virtual network to which VMs will connect when they're created at failover. The Azure virtual network must be in the same region as the one in which Site Recovery runs.
- An Azure standard storage account to store replication logs. This can be the same storage account for the VM disks that are being migrated.

Prerequisites

- Understand the relevant migration scenario components in the preceding section.
- Plan your downtime by learning about [failover in Site Recovery](#).

Setup and migration steps

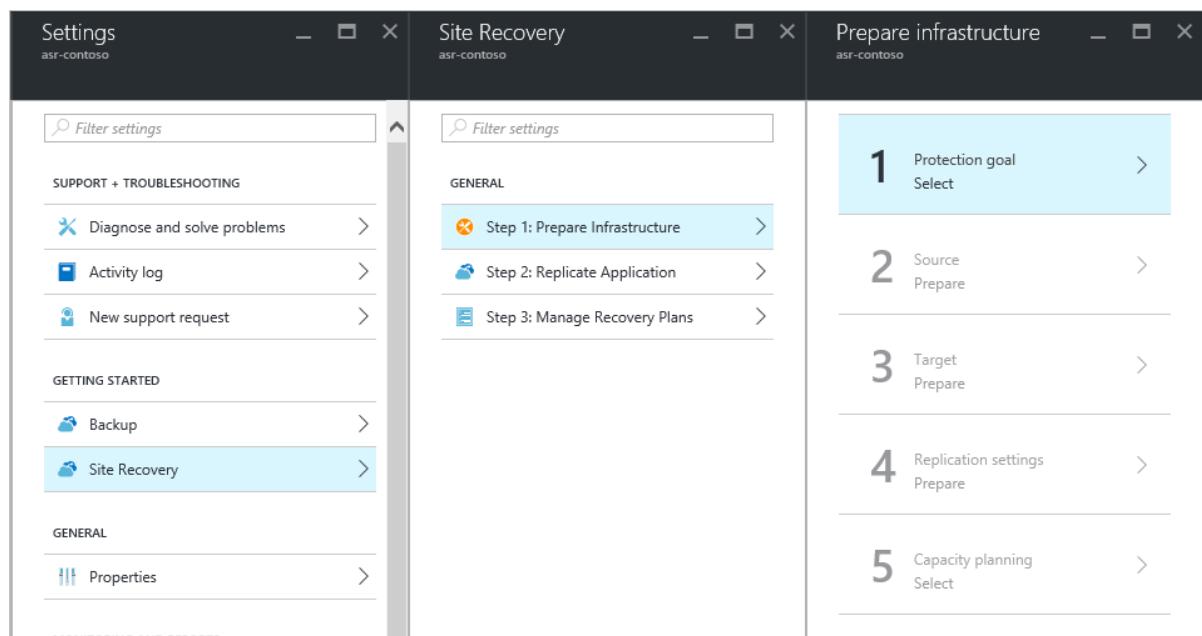
You can use Site Recovery to migrate Azure IaaS VMs between regions or within same region. The following instructions are tailored for this migration scenario from the article [Replicate VMware VMs or physical servers to Azure](#). Please follow the links for detailed steps in addition to the instructions in this article.

Step 1: Create a Recovery Services vault

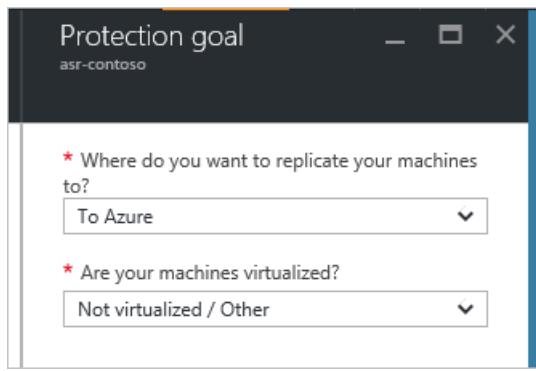
1. Open the [Azure portal](#).
2. Select **New > Management > Backup and Site Recovery (OMS)**. Alternatively, you can select **Browse > Recovery Services Vault > Add**.
3. Specify a region that VMs will be replicated to. For the purpose of migration in the same region, select the region where your source VMs and source storage accounts are.

Step 2: Choose your protection goals

1. On the VM where you want to install the configuration server, open the [Azure portal](#).
2. Go to **Recovery Services vaults > Settings > Site Recovery > Step 1: Prepare Infrastructure > Protection goal**.



3. Under **Protection goal**, in the first drop-down list, select **To Azure**. In the second drop-down list, select **Not virtualized / Other**, and then select **OK**.

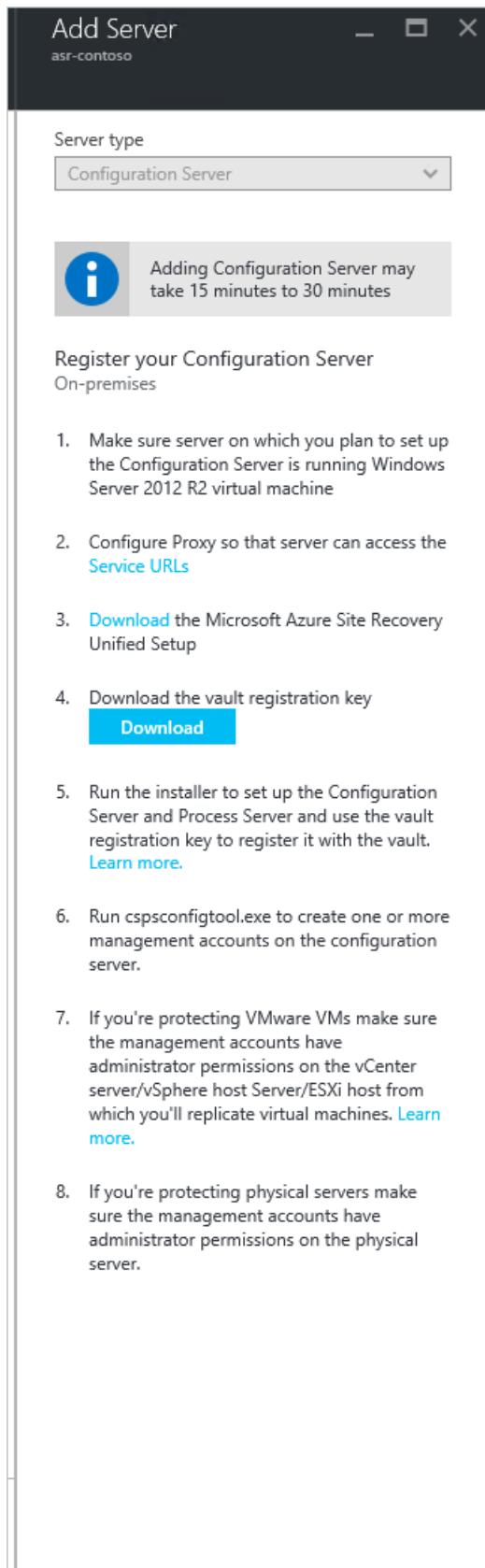


Step 3: Set up the source environment (configuration server)

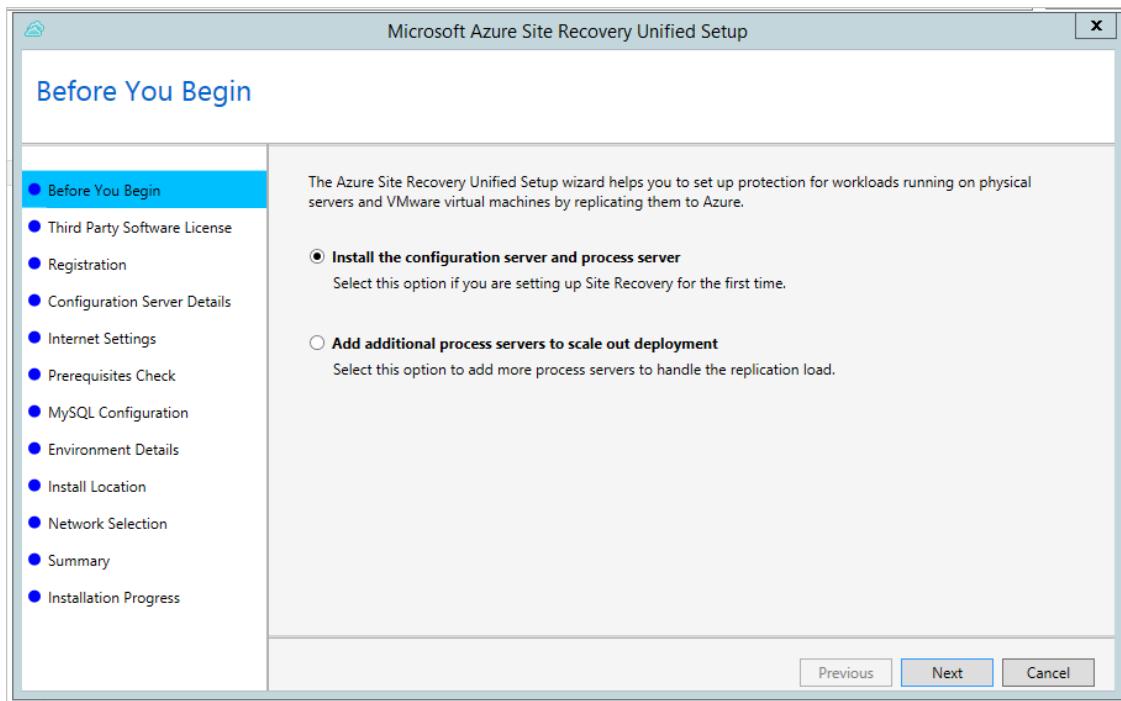
1. Download **Azure Site Recovery Unified Setup** and the vault registration key by going to the **Prepare infrastructure > Prepare source > Add Server** panes.

You will need the vault registration key to run the unified setup. The key is valid for five days after you generate it.

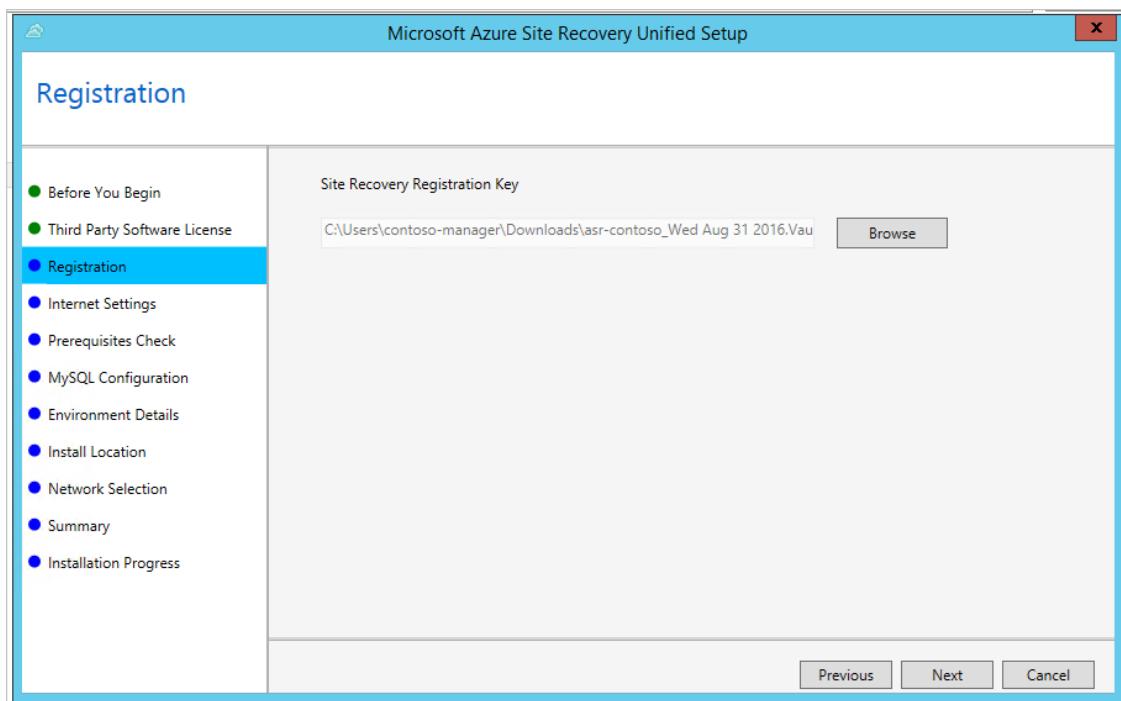
2. In the **Add Server** pane, add a configuration server.



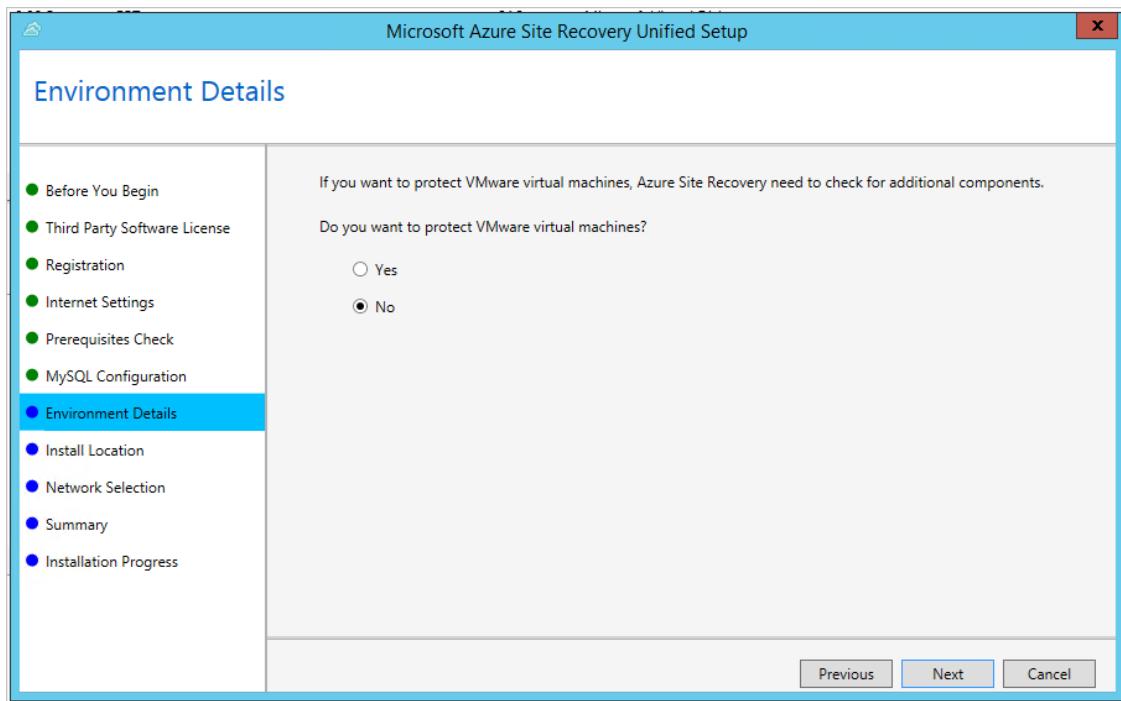
3. On the VM that you're using as the configuration server, run Unified Setup to install the configuration server and the process server. You can [walk through the screenshots](#) to complete the installation. You can refer to the following screenshots for steps specified for this migration scenario.
 - a. In **Before You Begin**, select **Install the configuration server and process server**.



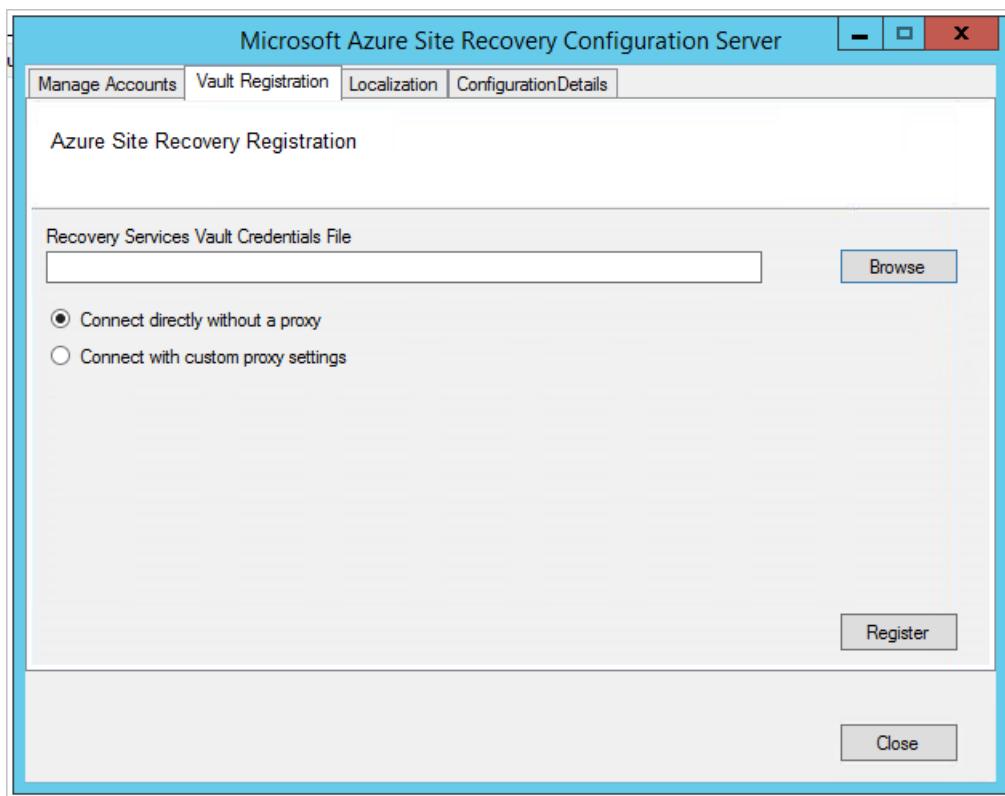
- b. In **Registration**, browse and select the registration key that you downloaded from the vault.



- c. In **Environment Details**, select whether you're going to replicate VMware VMs. For this migration scenario, choose **No**.



4. After the installation is complete, do the following in the **Microsoft Azure Site Recovery Configuration Server** window:
 - a. Use the **Manage Accounts** tab to create the account that Site Recovery can use for automatic discovery. (In the scenario about protecting physical machines, setting up the account isn't relevant, but you need at least one account to enable one of the following steps. In this case, you can name the account and password as any.)
 - b. Use the **Vault Registration** tab to upload the vault credential file.



Step 4: Set up the target environment

Select **Prepare infrastructure > Target**, and specify the deployment model that you want to use for VMs after failover. You can choose **Classic** or **Resource Manager**, depending on your scenario.

The screenshot shows the Azure Site Recovery wizard interface. On the left, a vertical list of steps is shown:

- 1 Protection goal (VMware VMs/physical servers t...)
- 2 Source (CONTOSO-CONFIG)
- 3 Target (Prepare) - This step is highlighted with a blue background.
- 4 Replication settings (Prepare)
- 5 Capacity planning (Select)

On the right, the "Target" pane displays the configuration for the selected target environment:

- Step 1 : Select Azure subscription**
 - * Subscription: Visual Studio Enterprise
 - * Select the deployment model used after failover:
 - Classic (selected)
 - Resource Manager
- Step 2 : Ensure that at least one compatible Azure storage account exist**
 - Storage account(s):
 - Found 5 compatible Azure storage accounts out of 6 available in the subscription
- Step 3 : Ensure that at least one compatible Azure virtual network exist**
 - Network(s):
 - Found 2 compatible Azure virtual networks out of 2 available in the subscription

Site Recovery checks that you have one or more compatible Azure storage accounts and networks.

NOTE

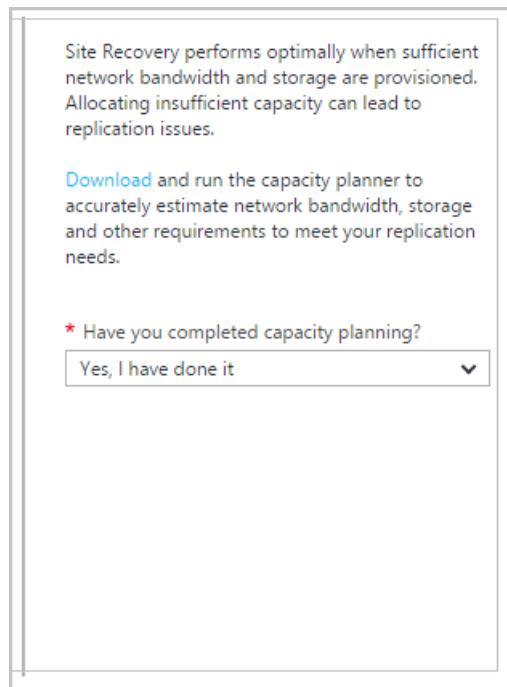
If you're using a premium storage account for replicated data, you need to set up an additional standard storage account to store replication logs.

Step 5: Set up replication settings

To verify that your configuration server is successfully associated with the replication policy that you create, follow [Set up replication settings](#).

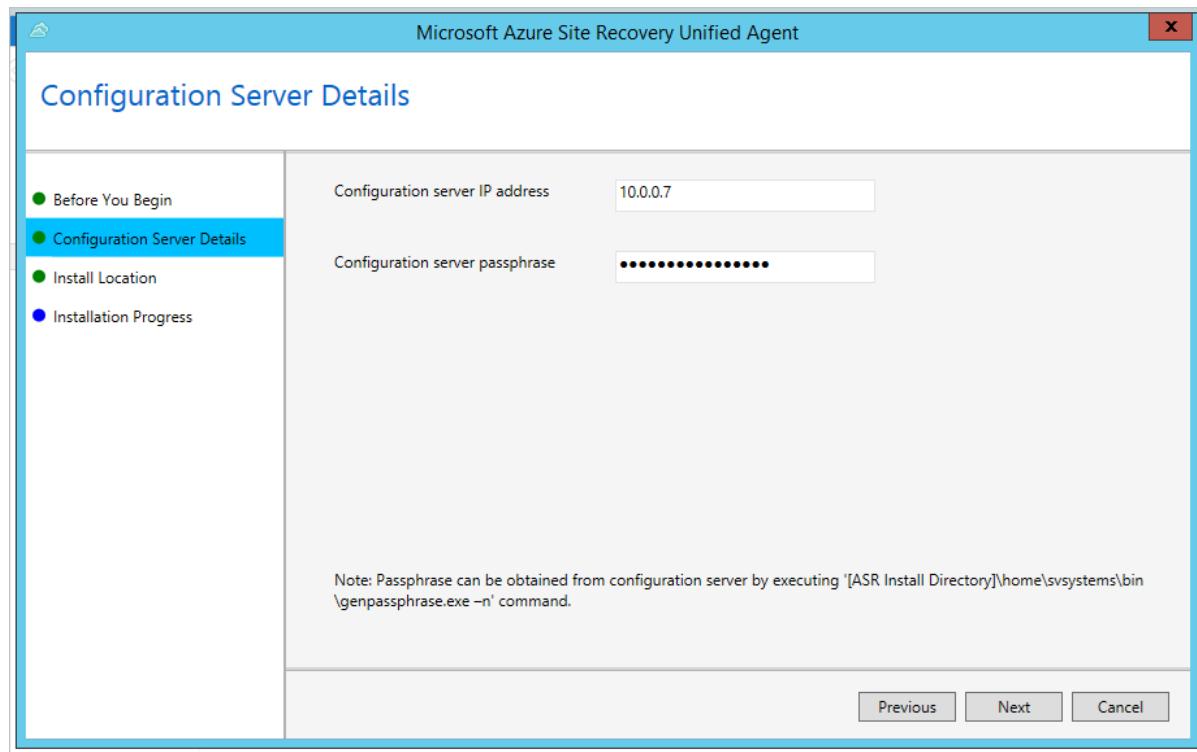
Step 6: Plan capacity

1. Use the [capacity planner](#) to accurately estimate network bandwidth, storage, and other requirements to meet your replication needs.
2. When you're done, select **Yes, I have done it** in **Have you completed capacity planning?**.



Step 7: Install the mobility service and enable replication

1. You can choose to [push installation](#) to your source VMs or to [manually install the mobility service](#) on your source VMs. You can find the requirement of pushing installation and the path of the manual installer in the provided link. If you're doing a manual installation, you might need to use an internal IP address to find the configuration server.



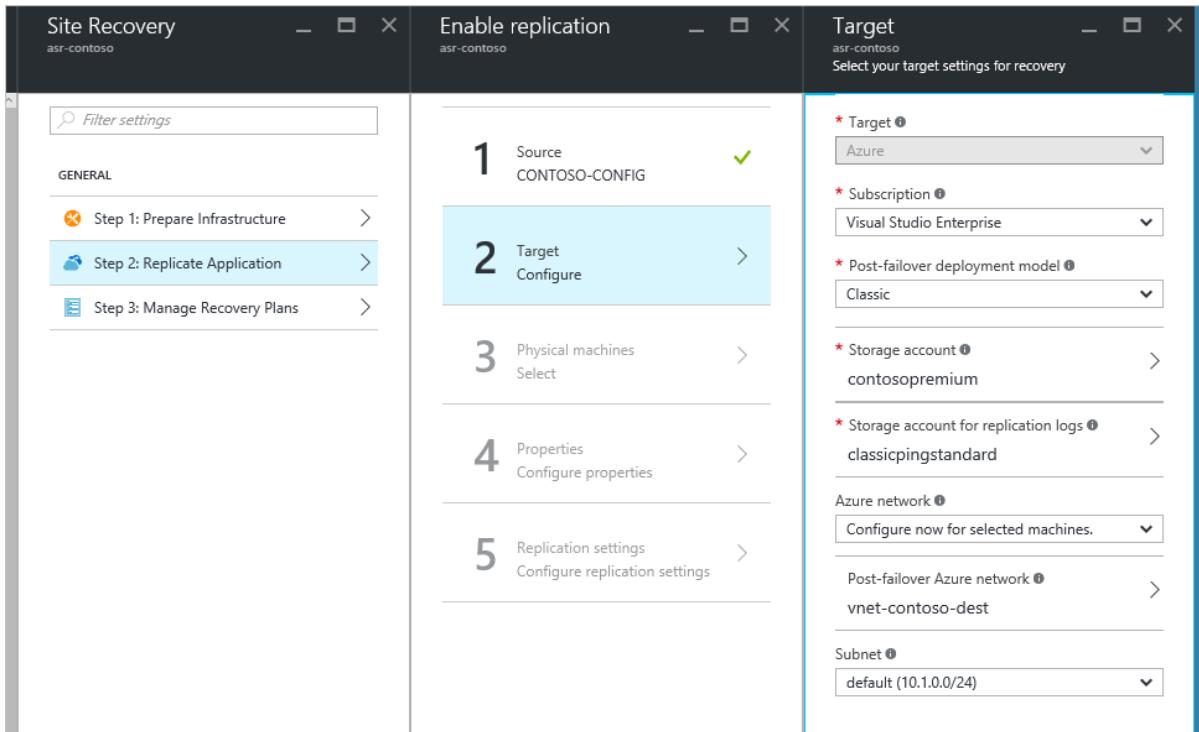
The failed-over VM will have two temporary disks: one from the primary VM and the other created during the provisioning of the VM in the recovery region. To exclude the temporary disk before replication, install the mobility service before you enable replication. To learn more about how to exclude the temporary disk, see [Exclude disks from replication](#).

2. Enable replication as follows:
 - a. Select **Replicate Application > Source**. After you've enabled replication for the first time, select **+Replicate** in the vault to enable replication for additional machines.

- b. In step 1, set up **Source** as your process server.
- c. In step 2, specify the post-failover deployment model, a premium storage account to migrate to, a standard storage account to save logs, and a virtual network to fail to.
- d. In step 3, add protected VMs by IP address. (You might need an internal IP address to find them.)
- e. In step 4, configure the properties by selecting the accounts that you set up previously on the process server.
- f. In step 5, choose the replication policy that you created previously in "Step 5: Set up replication settings."
- g. Select **OK**.

NOTE

When an Azure VM is deallocated and started again, there is no guarantee that it will get the same IP address. If the IP address of the configuration server/process server or the protected Azure VMs changes, the replication in this scenario might not work correctly.



When you design your Azure Storage environment, we recommend that you use separate storage accounts for each VM in an availability set. We recommend that you follow the best practice in the storage layer to [use multiple storage accounts for each availability set](#). Distributing VM disks to multiple storage accounts helps to improve storage availability and distributes the I/O across the Azure storage infrastructure.

If your VMs are in an availability set, instead of replicating disks of all VMs into one storage account, we highly recommend migrating multiple VMs multiple times. That way, the VMs in the same availability set do not share a single storage account. Use the **Enable Replication** pane to set up a destination storage account for each VM, one at a time.

You can choose a post-failover deployment model according to your need. If you choose Azure Resource Manager as your post-failover deployment model, you can fail over a VM (Resource Manager) to a VM (Resource Manager), or you can fail over a VM (classic) to a VM (Resource Manager).

Step 8: Run a test failover

To check whether your replication is complete, select your Site Recovery instance and then select **Settings > Replicated Items**. You will see the status and percentage of your replication process.

After initial replication is complete, run a test failover to validate your replication strategy. For detailed steps of a

test failover, see [Run a test failover in Site Recovery](#).

NOTE

Before you run any failover, make sure that your VMs and replication strategy meet the requirements. For more information about running a test failover, see [Test failover to Azure in Site Recovery](#).

You can see the status of your test failover in **Settings > Jobs > YOUR_FAILOVER_PLAN_NAME**. In the pane, you can see a breakdown of the steps and success/failure results. If the test failover fails at any step, select the step to check the error message.

Step 9: Run a failover

After the test failover is completed, run a failover to migrate your disks to Premium Storage and replicate the VM instances. Follow the detailed steps in [Run a failover](#).

Be sure to select **Shut down VMs and synchronize the latest data**. This option specifies that Site Recovery should try to shut down the protected VMs and synchronize the data so that the latest version of the data will be failed over. If you don't select this option or the attempt doesn't succeed, the failover will be from the latest available recovery point for the VM.

Site Recovery will create a VM instance whose type is the same as or similar to a Premium Storage-capable VM. You can check the performance and price of various VM instances by going to [Windows Virtual Machines Pricing](#) or [Linux Virtual Machines Pricing](#).

Post-migration steps

1. **Configure replicated VMs to the availability set if applicable.** Site Recovery does not support migrating VMs along with the availability set. Depending on the deployment of your replicated VM, do one of the following:
 - For a VM created through the classic deployment model: Add the VM to the availability set in the Azure portal. For detailed steps, go to [Add an existing virtual machine to an availability set](#).
 - For a VM created through the Resource Manager deployment model: Save your configuration of the VM and then delete and re-create the VMs in the availability set. To do so, use the script at [Set Azure Resource Manager VM Availability Set](#). Before you run this script, check its limitations and plan your downtime.
2. **Delete old VMs and disks.** Make sure that the Premium disks are consistent with source disks and that the new VMs perform the same function as the source VMs. Delete the VM and delete the disks from your source storage accounts in the Azure portal. If there's a problem in which the disk is not deleted even though you deleted the VM, see [Troubleshoot errors when you delete VHDS](#).
3. **Clean the Azure Site Recovery infrastructure.** If Site Recovery is no longer needed, you can clean its infrastructure. Delete replicated items, the configuration server, and the recovery policy, and then delete the Azure Site Recovery vault.

Troubleshooting

- [Monitor and troubleshoot protection for virtual machines and physical servers](#)
- [Microsoft Azure Site Recovery forum](#)

Next steps

For specific scenarios for migrating virtual machines, see the following resources:

- [Migrate Azure Virtual Machines between Storage Accounts](#)
- [Create and upload a Windows Server VHD to Azure](#)
- [Creating and uploading a virtual hard disk that contains the Linux operating system](#)
- [Migrating Virtual Machines from Amazon AWS to Microsoft Azure](#)

Also, see the following resources to learn more about Azure Storage and Azure Virtual Machines:

- [Azure Storage](#)
- [Azure Virtual Machines](#)
- [Premium Storage: High-performance storage for Azure virtual machine workloads](#)

Use Azure file shares with Windows VMs

8/21/2017 • 2 min to read • [Edit Online](#)

You can use Azure file shares as a way to store and access files from your VM. For example, you can store a script or an application configuration file that you want all your VMs to share. In this topic, we show you how to create and mount an Azure file share, and how to upload and download files.

Connect to a file share from a VM

This section assumes you already have a file share that you want to connect to. If you need to create one, see [Create a file share](#) later in this topic.

1. Sign in to the [Azure portal](#).
2. On the left menu, click **Storage accounts**.
3. Choose your storage account.
4. In the **Overview** page, under **Services**, select **Files**.
5. Select a file share.
6. Click **Connect** to open a page that shows the command-line syntax for mounting the file share from Windows or Linux.
7. Highlight the syntax of the command and paste it into Notepad or someplace else where you can easily access it.
8. Edit the syntax to remove the leading > **** and replace [drive letter] with the drive letter (for example, **Y:)** where you would like to mount the file share.
9. Connect to your VM and open a command prompt.
10. Paste in the edited connection syntax and hit **Enter**.
11. When the connection has been created, you get the message **The command completed successfully**.
12. Check the connection by typing in the drive letter to switch to that drive and then type **dir** to see the contents of the file share.

Create a file share

1. Sign in to the [Azure portal](#).
2. On the left menu, click **Storage accounts**.
3. Choose your storage account.
4. In the **Overview** page, under **Services**, select **Files**.
5. In the File Service page, click **+ File share** to create your first file share.\
6. Fill in the file share name. File share names can use lowercase letters, numbers and single hyphens. The name cannot start with a hyphen and you can't use multiple, consecutive hyphens.
7. Fill in a limit on how large the file can be, up to 5120 GB.
8. Click **OK** to deploy the file share.

Upload files

1. Sign in to the [Azure portal](#).
2. On the left menu, click **Storage accounts**.
3. Choose your storage account.
4. In the **Overview** page, under **Services**, select **Files**.
5. Select a file share.

6. Click **Upload** to open the **Upload files** page.
7. Click on the folder icon to browse your local file system for a file to upload.
8. Click **Upload** to upload the file to the file share.

Download files

1. Sign in to the [Azure portal](#).
2. On the left menu, click **Storage accounts**.
3. Choose your storage account.
4. In the **Overview** page, under **Services**, select **Files**.
5. Select a file share.
6. Right-click on the file and choose **Download** to download it to your local machine.

Next steps

You can also create and manage file shares using PowerShell. For more information, see [Get started with Azure File storage on Windows](#).

Using Managed Disks in Azure Resource Manager Templates

8/21/2017 • 4 min to read • [Edit Online](#)

This document walks through the differences between managed and unmanaged disks when using Azure Resource Manager templates to provision virtual machines. This will help you to update existing templates that are using unmanaged Disks to managed disks. For reference, we are using the [101-vm-simple-windows](#) template as a guide. You can see the template using both [managed Disks](#) and a prior version using [unmanaged disks](#) if you'd like to directly compare them.

Unmanaged Disks template formatting

To begin, we take a look at how unmanaged disks are deployed. When creating unmanaged disks, you need a storage account to hold the VHD files. You can create a new storage account or use one that already exists. This article will show you how to create a new storage account. To accomplish this, you need a storage account resource in the resources block as shown below.

```
{  
  "type": "Microsoft.Storage/storageAccounts",  
  "name": "[variables('storageAccountName')]",  
  "apiVersion": "2016-01-01",  
  "location": "[resourceGroup().location]",  
  "sku": {  
    "name": "Standard_LRS"  
  },  
  "kind": "Storage",  
  "properties": {}  
}
```

Within the virtual machine object, we need a dependency on the storage account to ensure that it's created before the virtual machine. Within the `storageProfile` section, we then specify the full URI of the VHD location, which references the storage account and is needed for the OS disk and any data disks.

```
{
    "apiVersion": "2015-06-15",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[variables('vmName')]",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Storage/storageAccounts/', variables('storageAccountName'))]",
        "[resourceId('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties": {
        "hardwareProfile": {...},
        "osProfile": {...},
        "storageProfile": {
            "imageReference": {
                "publisher": "MicrosoftWindowsServer",
                "offer": "WindowsServer",
                "sku": "[parameters('windowsOSVersion')]",
                "version": "latest"
            },
            "osDisk": {
                "name": "osdisk",
                "vhd": {
                    "uri": "[concat(reference(resourceId('Microsoft.Storage/storageAccounts/',
variables('storageAccountName'))).primaryEndpoints.blob, 'vhds/osdisk.vhd')]"
                },
                "caching": "ReadWrite",
                "createOption": "FromImage"
            },
            "dataDisks": [
                {
                    "name": "datadisk1",
                    "diskSizeGB": 1023,
                    "lun": 0,
                    "vhd": {
                        "uri": "[concat(reference(resourceId('Microsoft.Storage/storageAccounts/',
variables('storageAccountName'))).primaryEndpoints.blob, 'vhds/datadisk1.vhd')]"
                    },
                    "createOption": "Empty"
                }
            ]
        },
        "networkProfile": {...},
        "diagnosticsProfile": {...}
    }
}
```

Managed disks template formatting

With Azure Managed Disks, the disk becomes a top-level resource and no longer requires a storage account to be created by the user. Managed disks were first exposed in the [2016-04-30-preview](#) API version, they are available in all subsequent API versions and are now the default disk type. The following sections walk through the default settings and detail how to further customize your disks.

NOTE

It is recommended to use an API version later than [2016-04-30-preview](#) as there were breaking changes between [2016-04-30-preview](#) and [2017-03-30](#).

Default managed disk settings

To create a VM with managed disks, you no longer need to create the storage account resource and can update your virtual machine resource as follows. Specifically note that the `apiVersion` reflects [2017-03-30](#) and the `osDisk`

and `dataDisks` no longer refer to a specific URI for the VHD. When deploying without specifying additional properties, the disk will use [Standard LRS storage](#). If no name is specified, it takes the format of `<VMName>_OsDisk_1_<randomstring>` for the OS disk and `<VMName>_disk<#>_<randomstring>` for each data disk. By default, Azure disk encryption is disabled; caching is Read/Write for the OS disk and None for data disks. You may notice in the example below there is still a storage account dependency, though this is only for storage of diagnostics and is not needed for disk storage.

```
{
    "apiVersion": "2017-03-30",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[variables('vmName')]",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Storage/storageAccounts/', variables('storageAccountName'))]",
        "[resourceId('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties": {
        "hardwareProfile": {...},
        "osProfile": {...},
        "storageProfile": {
            "imageReference": {
                "publisher": "MicrosoftWindowsServer",
                "offer": "WindowsServer",
                "sku": "[parameters('windowsOSVersion')]",
                "version": "latest"
            },
            "osDisk": {
                "createOption": "FromImage"
            },
            "dataDisks": [
                {
                    "diskSizeGB": 1023,
                    "lun": 0,
                    "createOption": "Empty"
                }
            ]
        },
        "networkProfile": {...},
        "diagnosticsProfile": {...}
    }
}
```

Using a top-level managed disk resource

As an alternative to specifying the disk configuration in the virtual machine object, you can create a top-level disk resource and attach it as part of the virtual machine creation. For example, we can create a disk resource as follows to use as a data disk.

```
{
    "type": "Microsoft.Compute/disks",
    "name": "[concat(variables('vmName'), '-datadisk1')]",
    "apiVersion": "2017-03-30",
    "location": "[resourceGroup().location]",
    "sku": {
        "name": "Standard_LRS"
    },
    "properties": {
        "creationData": {
            "createOption": "Empty"
        },
        "diskSizeGB": 1023
    }
}
```

Within the VM object, we can then reference this disk object to be attached. Specifying the resource ID of the managed disk we created in the `managedDisk` property allows the attachment of the disk as the VM is created. Note that the `apiVersion` for the VM resource is set to `2017-03-30`. Also note that we've created a dependency on the disk resource to ensure it's successfully created before VM creation.

```
{  
    "apiVersion": "2017-03-30",  
    "type": "Microsoft.Compute/virtualMachines",  
    "name": "[variables('vmName')]",  
    "location": "[resourceGroup().location]",  
    "dependsOn": [  
        "[resourceId('Microsoft.Storage/storageAccounts/', variables('storageAccountName'))]",  
        "[resourceId('Microsoft.Network/networkInterfaces/', variables('nicName'))]",  
        "[resourceId('Microsoft.Compute/disks/', concat(variables('vmName'), '-datadisk1'))]"  
    ],  
    "properties": {  
        "hardwareProfile": {...},  
        "osProfile": {...},  
        "storageProfile": {  
            "imageReference": {  
                "publisher": "MicrosoftWindowsServer",  
                "offer": "WindowsServer",  
                "sku": "[parameters('windowsOSVersion')]",  
                "version": "latest"  
            },  
            "osDisk": {  
                "createOption": "FromImage"  
            },  
            "dataDisks": [  
                {  
                    "lun": 0,  
                    "name": "[concat(variables('vmName'), '-datadisk1')]",  
                    "createOption": "attach",  
                    "managedDisk": {  
                        "id": "[resourceId('Microsoft.Compute/disks/', concat(variables('vmName'), '-datadisk1'))]"  
                    }  
                }  
            ]  
        },  
        "networkProfile": {...},  
        "diagnosticsProfile": {...}  
    }  
}
```

Create managed availability sets with VMs using managed disks

To create managed availability sets with VMs using managed disks, add the `sku` object to the availability set resource and set the `name` property to `Aligned`. This ensures that the disks for each VM are sufficiently isolated from each other to avoid single points of failure. Also note that the `apiVersion` for the availability set resource is set to `2017-03-30`.

```
{  
    "apiVersion": "2017-03-30",  
    "type": "Microsoft.Compute/availabilitySets",  
    "location": "[resourceGroup().location]",  
    "name": "[variables('avSetName')]",  
    "properties": {  
        "PlatformUpdateDomainCount": 3,  
        "PlatformFaultDomainCount": 2  
    },  
    "sku": {  
        "name": "Aligned"  
    }  
}
```

Additional scenarios and customizations

To find full information on the REST API specifications, please review the [create a managed disk REST API documentation](#). You will find additional scenarios, as well as default and acceptable values that can be submitted to the API through template deployments.

Next steps

- For full templates that use managed disks visit the following Azure Quickstart Repo links.
 - [Windows VM with managed disk](#)
 - [Linux VM with managed disk](#)
 - [Full list of managed disk templates](#)
- Visit the [Azure Managed Disks Overview](#) document to learn more about managed disks.
- Review the template reference documentation for virtual machine resources by visiting the [Microsoft.Compute/virtualMachines template reference](#) document.
- Review the template reference documentation for disk resources by visiting the [Microsoft.Compute/disks template reference](#) document.

Migrate Azure VMs to Managed Disks in Azure

11/1/2017 • 3 min to read • [Edit Online](#)

Azure Managed Disks simplifies your storage management by removing the need to separately manage storage accounts. You can also migrate your existing Azure VMs to Managed Disks to benefit from better reliability of VMs in an Availability Set. It ensures that the disks of different VMs in an Availability Set will be sufficiently isolated from each other to avoid single point of failures. It automatically places disks of different VMs in an Availability Set in different Storage scale units (stamps) which limits the impact of single Storage scale unit failures caused due to hardware and software failures. Based on your needs, you can choose from two types of storage options:

- [Premium Managed Disks](#) are Solid State Drive (SSD) based storage media which delivers high performance, low-latency disk support for virtual machines running I/O-intensive workloads. You can take advantage of the speed and performance of these disks by migrating to Premium Managed Disks.
- [Standard Managed Disks](#) use Hard Disk Drive (HDD) based storage media and are best suited for Dev/Test and other infrequent access workloads that are less sensitive to performance variability.

You can migrate to Managed Disks in following scenarios:

MIGRATE...	DOCUMENTATION LINK
Convert stand alone VMs and VMs in an availability set to managed disks	Convert VMs to use managed disks
A single VM from classic to Resource Manager on managed disks	Migrate a single VM
All the VMs in a vNet from classic to Resource Manager on managed disks	Migrate IaaS resources from classic to Resource Manager and then Convert a VM from unmanaged disks to managed disks

Plan for the conversion to Managed Disks

This section helps you to make the best decision on VM and disk types.

Location

Pick a location where Azure Managed Disks are available. If you are moving to Premium Managed Disks, also ensure that Premium storage is available in the region where you are planning to move to. See [Azure Services by Region](#) for up-to-date information on available locations.

VM sizes

If you are migrating to Premium Managed Disks, you have to update the size of the VM to Premium Storage capable size available in the region where VM is located. Review the VM sizes that are Premium Storage capable. The Azure VM size specifications are listed in [Sizes for virtual machines](#). Review the performance characteristics of virtual machines that work with Premium Storage and choose the most appropriate VM size that best suits your workload. Make sure that there is sufficient bandwidth available on your VM to drive the disk traffic.

Disk sizes

Premium Managed Disks

There are seven types of premium managed disks that can be used with your VM and each has specific IOPs and throughput limits. Take into consideration these limits when choosing the Premium disk type for your VM based on the needs of your application in terms of capacity, performance, scalability, and peak loads.

Premium Disks Type	P4	P6	P10	P20	P30	P40	P50
Disk size	128 GB	512 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)
IOPS per disk	120	240	500	2300	5000	7500	7500
Throughput per disk	25 MB per second	50 MB per second	100 MB per second	150 MB per second	200 MB per second	250 MB per second	250 MB per second

Standard Managed Disks

There are seven types of standard managed disks that can be used with your VM. Each of them have different capacity but have same IOPS and throughput limits. Choose the type of Standard Managed disks based on the capacity needs of your application.

Standard Disk Type	S4	S6	S10	S20	S30	S40	S50
Disk size	30 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2TB)	4095 GB (4 TB)
IOPS per disk	500	500	500	500	500	500	500
Throughput per disk	60 MB per second						

Disk caching policy

Premium Managed Disks

By default, disk caching policy is *Read-Only* for all the Premium data disks, and *Read-Write* for the Premium operating system disk attached to the VM. This configuration setting is recommended to achieve the optimal performance for your application's IOs. For write-heavy or write-only data disks (such as SQL Server log files), disable disk caching so that you can achieve better application performance.

Pricing

Review the [pricing for Managed Disks](#). Pricing of Premium Managed Disks is same as the Premium Unmanaged Disks. But pricing for Standard Managed Disks is different than Standard Unmanaged Disks.

Next steps

- Learn more about [Managed Disks](#)

Convert a Windows virtual machine from unmanaged disks to managed disks

12/15/2017 • 3 min to read • [Edit Online](#)

If you have existing Windows virtual machines (VMs) that use unmanaged disks, you can convert the VMs to use managed disks through the [Azure Managed Disks](#) service. This process converts both the OS disk and any attached data disks.

This article shows you how to convert VMs by using Azure PowerShell. If you need to install or upgrade it, see [Install and configure Azure PowerShell](#).

Before you begin

- Review [Plan for the migration to Managed Disks](#).
- Review [the FAQ about migration to Managed Disks](#).
- The conversion requires a restart of the VM, so schedule the migration of your VMs during a pre-existing maintenance window.
- The conversion is not reversible.
- Be sure to test the conversion. Migrate a test virtual machine before you perform the migration in production.
- During the conversion, you deallocate the VM. The VM receives a new IP address when it is started after the conversion. If needed, you can [assign a static IP address](#) to the VM.
- The original VHDs and the storage account used by the VM before conversion are not deleted. They continue to incur charges. To avoid being billed for these artifacts, delete the original VHD blobs after you verify that the conversion is complete.
- Review the minimum version of the Azure VM agent required to support the conversion process. For information on how to check and update your agent version, see [Minimum version support for VM agents in Azure](#)

Convert single-instance VMs

This section covers how to convert single-instance Azure VMs from unmanaged disks to managed disks. (If your VMs are in an availability set, see the next section.)

1. Deallocate the VM by using the [Stop-AzureRmVM](#) cmdlet. The following example deallocates the VM named `myVM` in the resource group named `myResourceGroup` :

```
$rgName = "myResourceGroup"
$vmName = "myVM"
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName -Force
```

2. Convert the VM to managed disks by using the [ConvertTo-AzureRmVMManagedDisk](#) cmdlet. The following process converts the previous VM, including the OS disk and any data disks:

```
ConvertTo-AzureRmVMManagedDisk -ResourceGroupName $rgName -VMName $vmName
```

3. Start the VM after the conversion to managed disks by using [Start-AzureRmVM](#). The following example restarts the previous VM:

```
Start-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

Convert VMs in an availability set

If the VMs that you want to convert to managed disks are in an availability set, you first need to convert the availability set to a managed availability set.

1. Convert the availability set by using the [Update-AzureRmAvailabilitySet](#) cmdlet. The following example updates the availability set named `myAvailabilitySet` in the resource group named `myResourceGroup`:

```
$rgName = 'myResourceGroup'  
$avSetName = 'myAvailabilitySet'  
  
$avSet = Get-AzureRmAvailabilitySet -ResourceGroupName $rgName -Name $avSetName  
Update-AzureRmAvailabilitySet -AvailabilitySet $avSet -Sku Aligned
```

If the region where your availability set is located has only 2 managed fault domains but the number of unmanaged fault domains is 3, this command shows an error similar to "The specified fault domain count 3 must fall in the range 1 to 2." To resolve the error, update the fault domain to 2 and update `Sku` to `Aligned` as follows:

```
$avSet.PlatformFaultDomainCount = 2  
Update-AzureRmAvailabilitySet -AvailabilitySet $avSet -Sku Aligned
```

2. Deallocate and convert the VMs in the availability set. The following script deallocates each VM by using the [Stop-AzureRmVM](#) cmdlet, converts it by using [ConvertTo-AzureRmVMManagedDisk](#), and restarts it by using [Start-AzureRmVM](#):

```
$avSet = Get-AzureRmAvailabilitySet -ResourceGroupName $rgName -Name $avSetName  
  
foreach($vmInfo in $avSet.VirtualMachinesReferences)  
{  
    $vm = Get-AzureRmVM -ResourceGroupName $rgName | Where-Object {$_.Id -eq $vmInfo.id}  
    Stop-AzureRmVM -ResourceGroupName $rgName -Name $vm.Name -Force  
    ConvertTo-AzureRmVMManagedDisk -ResourceGroupName $rgName -VMName $vm.Name  
    Start-AzureRmVM -ResourceGroupName $rgName -Name $vm.Name  
}
```

Troubleshooting

If there is an error during conversion, or if a VM is in a failed state because of issues in a previous conversion, run the `ConvertTo-AzureRmVMManagedDisk` cmdlet again. A simple retry usually unblocks the situation. Before converting, make sure all the VM extensions are in the 'Provisioning succeeded' state or the conversion will fail with the error code 409.

Next steps

Convert standard managed disks to premium

Take a read-only copy of a VM by using [snapshots](#).

Manually migrate a Classic VM to a new ARM Managed Disk VM from the VHD

8/21/2017 • 5 min to read • [Edit Online](#)

This section helps you to migrate your existing Azure VMs from the classic deployment model to [Managed Disks](#) in the Resource Manager deployment model.

Plan for the migration to Managed Disks

This section helps you to make the best decision on VM and disk types.

Location

Pick a location where Azure Managed Disks are available. If you are migrating to Premium Managed Disks, also ensure that Premium storage is available in the region where you are planning to migrate to. See [Azure Services byRegion](#) for up-to-date information on available locations.

VM sizes

If you are migrating to Premium Managed Disks, you have to update the size of the VM to Premium Storage capable size available in the region where VM is located. Review the VM sizes that are Premium Storage capable. The Azure VM size specifications are listed in [Sizes for virtual machines](#). Review the performance characteristics of virtual machines that work with Premium Storage and choose the most appropriate VM size that best suits your workload. Make sure that there is sufficient bandwidth available on your VM to drive the disk traffic.

Disk sizes

Premium Managed Disks

There are seven types of premium Managed disks that can be used with your VM and each has specific IOPs and throughput limits. Consider these limits when choosing the Premium disk type for your VM based on the needs of your application in terms of capacity, performance, scalability, and peak loads.

PREMIUM DISKS TYPE	P4	P6	P10	P20	P30	P40	P50
Disk size	128 GB	512 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2 TB)	4095 GB (4 TB)
IOPS per disk	120	240	500	2300	5000	7500	7500
Throughput per disk	25 MB per second	50 MB per second	100 MB per second	150 MB per second	200 MB per second	250 MB per second	250 MB per second

Standard Managed Disks

There are seven types of Standard Managed disks that can be used with your VM. Each of them have different capacity but have same IOPS and throughput limits. Choose the type of Standard Managed disks based on the capacity needs of your application.

STANDARD DISK TYPE	S4	S6	S10	S20	S30	S40	S50
Disk size	30 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)	2048 GB (2TB)	4095 GB (4 TB)
IOPS per disk	500	500	500	500	500	500	500
Throughput per disk	60 MB per second						

Disk caching policy

Premium Managed Disks

By default, disk caching policy is *Read-Only* for all the Premium data disks, and *Read-Write* for the Premium operating system disk attached to the VM. This configuration setting is recommended to achieve the optimal performance for your application's IOs. For write-heavy or write-only data disks (such as SQL Server log files), disable disk caching so that you can achieve better application performance.

Pricing

Review the [pricing for Managed Disks](#). Pricing of Premium Managed Disks is same as the Premium Unmanaged Disks. But pricing for Standard Managed Disks is different than Standard Unmanaged Disks.

Checklist

1. If you are migrating to Premium Managed Disks, make sure it is available in the region you are migrating to.
2. Decide the new VM series you will be using. It should be a Premium Storage capable if you are migrating to Premium Managed Disks.
3. Decide the exact VM size you will use which are available in the region you are migrating to. VM size needs to be large enough to support the number of data disks you have. For example, if you have four data disks, the VM must have two or more cores. Also, consider processing power, memory and network bandwidth needs.
4. Have the current VM details handy, including the list of disks and corresponding VHD blobs.

Prepare your application for downtime. To do a clean migration, you have to stop all the processing in the current system. Only then you can get it to consistent state which you can migrate to the new platform. Downtime duration depends on the amount of data in the disks to migrate.

Migrate the VM

Prepare your application for downtime. To do a clean migration, you have to stop all the processing in the current system. Only then you can get it to consistent state which you can migrate to the new platform. Downtime duration depends on the amount of data in the disks to migrate.

1. First, set the common parameters:

```

$resourceGroupName = 'yourResourceGroupName'

$location = 'your location'

$virtualNetworkName = 'yourExistingVirtualNetworkName'

$virtualMachineName = 'yourVMName'

$virtualMachineSize = 'Standard_DS3'

$adminUserName = "youradminusername"

$adminPassword = "yourpassword" | ConvertTo-SecureString -AsPlainText -Force

$imageName = 'yourImageName'

$osVhdUri = 'https://storageaccount.blob.core.windows.net/vhdcontainer/osdisk.vhd'

$dataVhdUri = 'https://storageaccount.blob.core.windows.net/vhdcontainer/datadisk1.vhd'

$dataDiskName = 'dataDisk1'

```

2. Create a managed OS disk using the VHD from the classic VM.

Ensure that you have provided the complete URI of the OS VHD to the \$osVhdUri parameter. Also, enter **-AccountType** as **PremiumLRS** or **StandardLRS** based on type of disks (Premium or Standard) you are migrating to.

```

$osDisk = New-AzureRmDisk -DiskName $osDiskName -Disk (New-AzureRmDiskConfig ' -AccountType PremiumLRS -Location $location -CreateOption Import -SourceUri $osVhdUri ) -ResourceGroupName $resourceGroupName

```

3. Attach the OS disk to the new VM.

```

$VirtualMachine = New-AzureRmVMConfig -VMName $virtualMachineName -VMSize $virtualMachineSize
$VirtualMachine = Set-AzureRmVMOSDisk -VM $VirtualMachine -ManagedDiskId $osDisk.Id ' -StorageAccountType PremiumLRS -DiskSizeInGB 128 -CreateOption Attach -Windows

```

4. Create a managed data disk from the data VHD file and add it to the new VM.

```

$dataDisk1 = New-AzureRmDisk -DiskName $dataDiskName -Disk (New-AzureRmDiskConfig ' -AccountType PremiumLRS -Location $location -CreationDataCreateOption Import ' -SourceUri $dataVhdUri ) -ResourceGroupName $resourceGroupName

$VirtualMachine = Add-AzureRmVMDataDisk -VM $VirtualMachine -Name $dataDiskName ' -CreateOption Attach -ManagedDiskId $dataDisk1.Id -Lun 1

```

5. Create the new VM by setting public IP, Virtual Network and NIC.

```
$publicIp = New-AzureRmPublicIpAddress -Name ($VirtualMachineName.ToLower()+'_ip') '  
-ResourceGroupName $resourceGroupName -Location $location -AllocationMethod Dynamic  
  
$vnet = Get-AzureRmVirtualNetwork -Name $virtualNetworkName -ResourceGroupName $resourceGroupName  
  
$nic = New-AzureRmNetworkInterface -Name ($VirtualMachineName.ToLower()+'_nic') '  
-ResourceGroupName $resourceGroupName -Location $location -SubnetId $vnet.Subnets[0].Id '  
-PublicIpAddressId $publicIp.Id  
  
$VirtualMachine = Add-AzureRmVMNetworkInterface -VM $VirtualMachine -Id $nic.Id  
  
New-AzureRmVM -VM $VirtualMachine -ResourceGroupName $resourceGroupName -Location $location
```

NOTE

There may be additional steps necessary to support your application that is not be covered by this guide.

Next steps

- Connect to the virtual machine. For instructions, see [How to connect and log on to an Azure virtual machine running Windows](#).

Common PowerShell commands for Azure Virtual Networks

10/12/2017 • 4 min to read • [Edit Online](#)

If you want to create a virtual machine, you need to create a [virtual network](#) or know about an existing virtual network in which the VM can be added. Typically, when you create a VM, you also need to consider creating the resources described in this article.

See [How to install and configure Azure PowerShell](#) for information about installing the latest version of Azure PowerShell, selecting your subscription, and signing in to your account.

Some variables might be useful for you if running more than one of the commands in this article:

- \$location - The location of the network resources. You can use [Get-AzureRmLocation](#) to find a [geographical region](#) that works for you.
- \$myResourceGroup - The name of the resource group where the network resources are located.

Create network resources

TASK	COMMAND
Create subnet configurations	\$subnet1 = New-AzureRmVirtualNetworkSubnetConfig -Name "mySubnet1" -AddressPrefix XX.X.X.X/XX \$subnet2 = New-AzureRmVirtualNetworkSubnetConfig -Name "mySubnet2" -AddressPrefix XX.X.X.X/XX A typical network might have a subnet for an internet facing load balancer and a separate subnet for an internal load balancer .
Create a virtual network	\$vnet = New-AzureRmVirtualNetwork -Name "myVNet" -ResourceGroupName \$myResourceGroup -Location \$location -AddressPrefix XX.X.X.X/XX -Subnet \$subnet1, \$subnet2
Test for a unique domain name	Test-AzureRmDnsAvailability -DomainNameLabel "myDNS" -Location \$location You can specify a DNS domain name for a public IP resource , which creates a mapping for domainname.location.cloudapp.azure.com to the public IP address in the Azure-managed DNS servers. The name can contain only letters, numbers, and hyphens. The first and last character must be a letter or number and the domain name must be unique within its Azure location. If True is returned, your proposed name is globally unique.
Create a public IP address	\$pip = New-AzureRmPublicIpAddress -Name "myPublicIp" -ResourceGroupName \$myResourceGroup -DomainNameLabel "myDNS" -Location \$location -AllocationMethod Dynamic The public IP address uses the domain name that you previously tested and is used by the frontend configuration of the load balancer.

TASK	COMMAND
Create a frontend IP configuration	<pre>\$frontendIP = New-AzureRmLoadBalancerFrontendIpConfig -Name "myFrontendIP" -PublicIpAddress \$pip</pre> <p>The frontend configuration includes the public IP address that you previously created for incoming network traffic.</p>
Create a backend address pool	<pre>\$beAddressPool = New-AzureRmLoadBalancerBackendAddressPoolConfig -Name "myBackendAddressPool"</pre> <p>Provides internal addresses for the backend of the load balancer that are accessed through a network interface.</p>
Create a probe	<pre>\$healthProbe = New-AzureRmLoadBalancerProbeConfig -Name "myProbe" -RequestPath 'HealthProbe.aspx' -Protocol http -Port 80 -IntervalInSeconds 15 -ProbeCount 2</pre> <p>Contains health probes used to check availability of virtual machines instances in the backend address pool.</p>
Create a load balancing rule	<pre>\$lbRule = New-AzureRmLoadBalancerRuleConfig -Name HTTP -FrontendIpConfiguration \$frontendIP -BackendAddressPool \$beAddressPool -Probe \$healthProbe -Protocol Tcp -FrontendPort 80 -BackendPort 80</pre> <p>Contains rules that assign a public port on the load balancer to a port in the backend address pool.</p>
Create an inbound NAT rule	<pre>\$inboundNATRule = New-AzureRmLoadBalancerInboundNatRuleConfig -Name "myInboundRule1" -FrontendIpConfiguration \$frontendIP -Protocol TCP -FrontendPort 3441 -BackendPort 3389</pre> <p>Contains rules mapping a public port on the load balancer to a port for a specific virtual machine in the backend address pool.</p>
Create a load balancer	<pre>\$loadBalancer = New-AzureRmLoadBalancer -ResourceGroupName \$myResourceGroup -Name "myLoadBalancer" -Location \$location -FrontendIpConfiguration \$frontendIP -InboundNatRule \$inboundNATRule -LoadBalancingRule \$lbRule -BackendAddressPool \$beAddressPool -Probe \$healthProbe</pre>
Create a network interface	<pre>\$nic1 = New-AzureRmNetworkInterface -ResourceGroupName \$myResourceGroup -Name "myNIC" -Location \$location -PrivateIpAddress XX.X.X.X -Subnet \$subnet2 -LoadBalancerBackendAddressPool \$loadBalancer.BackendAddressPools[0] -LoadBalancerInboundNatRule \$loadBalancer.InboundNatRules[0]</pre> <p>Create a network interface using the public IP address and virtual network subnet that you previously created.</p>

Get information about network resources

TASK	COMMAND
List virtual networks	<code>Get-AzureRmVirtualNetwork -ResourceGroupName \$myResourceGroup</code> Lists all the virtual networks in the resource group.
Get information about a virtual network	<code>Get-AzureRmVirtualNetwork -Name "myVNet" -ResourceGroupName \$myResourceGroup</code>
List subnets in a virtual network	<code>Get-AzureRmVirtualNetwork -Name "myVNet" -ResourceGroupName \$myResourceGroup Select Subnets</code>
Get information about a subnet	<code>Get-AzureRmVirtualNetworkSubnetConfig -Name "mySubnet1" -VirtualNetwork \$vnet</code> Gets information about the subnet in the specified virtual network. The \$vnet value represents the object returned by Get-AzureRmVirtualNetwork.
List IP addresses	<code>Get-AzureRmPublicIpAddress -ResourceGroupName \$myResourceGroup</code> Lists the public IP addresses in the resource group.
List load balancers	<code>Get-AzureRmLoadBalancer -ResourceGroupName \$myResourceGroup</code> Lists all the load balancers in the resource group.
List network interfaces	<code>Get-AzureRmNetworkInterface -ResourceGroupName \$myResourceGroup</code> Lists all the network interfaces in the resource group.
Get information about a network interface	<code>Get-AzureRmNetworkInterface -Name "myNIC" -ResourceGroupName \$myResourceGroup</code> Gets information about a specific network interface.
Get the IP configuration of a network interface	<code>Get-AzureRmNetworkInterfaceIPConfig -Name "myNICIP" -NetworkInterface \$nic</code> Gets information about the IP configuration of the specified network interface. The \$nic value represents the object returned by Get-AzureRmNetworkInterface.

Manage network resources

TASK	COMMAND
Add a subnet to a virtual network	<code>Add-AzureRmVirtualNetworkSubnetConfig -AddressPrefix XX.X.X.XXX -Name "mySubnet1" -VirtualNetwork \$vnet</code> Adds a subnet to an existing virtual network. The \$vnet value represents the object returned by Get-AzureRmVirtualNetwork.

TASK	COMMAND
Delete a virtual network	<pre>Remove-AzureRmVirtualNetwork -Name "myVNet" - ResourceGroupName \$myResourceGroup</pre> <p>Removes the specified virtual network from the resource group.</p>
Delete a network interface	<pre>Remove-AzureRmNetworkInterface -Name "myNIC" - ResourceGroupName \$myResourceGroup</pre> <p>Removes the specified network interface from the resource group.</p>
Delete a load balancer	<pre>Remove-AzureRmLoadBalancer -Name "myLoadBalancer" - ResourceGroupName \$myResourceGroup</pre> <p>Removes the specified load balancer from the resource group.</p>
Delete a public IP address	<pre>Remove-AzureRmPublicIpAddress -Name "myIPAddress" - ResourceGroupName \$myResourceGroup</pre> <p>Removes the specified public IP address from the resource group.</p>

Next Steps

- Use the network interface that you just created when you [create a VM](#).
- Learn about how you can [create a VM with multiple network interfaces](#).

Create a virtual network with multiple subnets

9/6/2017 • 10 min to read • [Edit Online](#)

In this tutorial, learn how to create a basic Azure virtual network that has separate public and private subnets. Resources in virtual networks can communicate with each other, and with resources in other networks connected to a virtual network. You can create Azure resources, like Virtual machines, App Service environments, Virtual machine scale sets, Azure HDInsight, and Cloud services in the same, or different subnets within a virtual network. Creating resources in different subnets enables you to filter network traffic in and out of subnets independently with [network security groups](#), and to [route traffic between subnets](#) through network virtual appliances, such as a firewall, if you choose to.

The following sections include steps that you can take to create a virtual network by using the [Azure portal](#), the Azure command-line interface ([Azure CLI](#)), [Azure PowerShell](#), and an [Azure Resource Manager template](#). The result is the same, regardless of which tool you use to create the virtual network. Click a tool link to go to that section of the tutorial. Learn more about all [virtual network](#) and [subnet](#) settings.

This article provides steps to create a virtual network through the Resource Manager deployment model, which is the deployment model we recommend using when creating new virtual networks. If you need to create a virtual network (classic), see [Create a virtual network \(classic\)](#). If you're not familiar with Azure's deployment models, see [Understand Azure deployment models](#).

Azure portal

1. In an Internet browser, go to the [Azure portal](#). Log in using your [Azure account](#). If you don't have an Azure account, you can sign up for a [free trial](#).
2. In the portal, click **+New > Networking > Virtual network**.
3. On the **Create virtual network** blade, enter the following values, and then click **Create**:

SETTING	VALUE
Name	myVnet
Address space	10.0.0.0/16
Subnet name	Public
Subnet address range	10.0.0.0/24
Resource group	Leave Create new selected, and then enter myResourceGroup .
Subscription and location	Select your subscription and location.

If you're new to Azure, learn more about [resource groups](#), [subscriptions](#), and [locations](#) (also referred to as [regions](#)).

4. In the portal, you can create only one subnet when you create a virtual network. In this tutorial, you create a second subnet after you create the virtual network. You might later create Internet-accessible resources in the **Public** subnet. You also might create resources that aren't accessible from the Internet in the **Private** subnet. To create the second subnet, in the **Search resources** box at the top of the page, enter **myVnet**. In the search

results, click **myVnet**. If you have multiple virtual networks with the same name in your subscription, check the resource groups that are listed under each virtual network. Ensure that you click the **myVnet** search result that has the resource group **myResourceGroup**.

5. On the **myVnet** blade, under **SETTINGS**, click **Subnets**.
6. On the **myVnet - Subnets** blade, click **+Subnet**.
7. On the **Add subnet** blade, for **Name**, enter **Private**. For **Address range**, enter **10.0.1.0/24**. Click **OK**.
8. On the **myVnet - Subnets** blade, review the subnets. You can see the **Public** and **Private** subnets that you created.
9. **Optional:** Complete additional tutorials listed under [Next steps](#) to filter network traffic in and out of each subnet with network security groups, to route traffic between subnets through a network virtual appliance, or to connect the virtual network to other virtual networks or on-premises networks.
10. **Optional:** Delete the resources that you create in this tutorial by completing the steps in [Delete resources](#).

Azure CLI

Azure CLI commands are the same, whether you execute the commands from Windows, Linux, or macOS. However, there are scripting differences between operating system shells. The script in the following steps executes in a Bash shell.

1. [Install and configure the Azure CLI](#). Ensure you have the most recent version of the Azure CLI installed. To get help for CLI commands, type `az <command> --help`. Rather than installing the CLI and its pre-requisites, you can use the Azure Cloud Shell. The Azure Cloud Shell is a free Bash shell that you can run directly within the Azure portal. The Cloud Shell has the Azure CLI preinstalled and configured to use with your account. To use the Cloud Shell, click the Cloud Shell (`>_`) button at the top of the [portal](#) or just click the *Try it* button in the steps that follow.
2. If running the CLI locally, log in to Azure with the `az login` command. If using the Cloud Shell, you're already logged in.
3. Review the following script and its comments. In your browser, copy the script and paste it into your CLI session:

```
#!/bin/bash

# Create a resource group.
az group create \
    --name myResourceGroup \
    --location eastus

# Create a virtual network with one subnet named Public.
az network vnet create \
    --name myVnet \
    --resource-group myResourceGroup \
    --subnet-name Public

# Create an additional subnet named Private in the virtual network.
az network vnet subnet create \
    --name Private \
    --address-prefix 10.0.1.0/24 \
    --vnet-name myVnet \
    --resource-group myResourceGroup
```

4. When the script is finished running, review the subnets for the virtual network. Copy the following command, and then paste it into your CLI session:

```
az network vnet subnet list --resource-group myResourceGroup --vnet-name myVnet --output table
```

5. **Optional:** Complete additional tutorials listed under [Next steps](#) to filter network traffic in and out of each subnet with network security groups, to route traffic between subnets through a network virtual appliance, or to connect the virtual network to other virtual networks or on-premises networks.
6. **Optional:** Delete the resources that you create in this tutorial by completing the steps in [Delete resources](#).

PowerShell

1. Install the latest version of the PowerShell [AzureRm](#) module. If you're new to Azure PowerShell, see [Azure PowerShell overview](#).
2. In a PowerShell session, log in to Azure with your [Azure account](#) using the `login-azurermaccount` command.
3. Review the following script and its comments. In your browser, copy the script and paste it into your PowerShell session:

```
# Create a resource group.  
New-AzureRmResourceGroup `  
    -Name myResourceGroup `  
    -Location eastus  
  
# Create the public and private subnets.  
$Subnet1 = New-AzureRmVirtualNetworkSubnetConfig `  
    -Name Public `  
    -AddressPrefix 10.0.0.0/24  
$Subnet2 = New-AzureRmVirtualNetworkSubnetConfig `  
    -Name Private `  
    -AddressPrefix 10.0.1.0/24  
  
# Create a virtual network.  
$Vnet=New-AzureRmVirtualNetwork `  
    -ResourceGroupName myResourceGroup `  
    -Location eastus `  
    -Name myVnet `  
    -AddressPrefix 10.0.0.0/16 `  
    -Subnet $Subnet1,$Subnet2
```

4. To review the subnets for the virtual network, copy the following command, and then paste it into your PowerShell session:

```
$Vnet.subnets | Format-Table Name, AddressPrefix
```

5. **Optional:** Complete additional tutorials listed under [Next steps](#) to filter network traffic in and out of each subnet with network security groups, to route traffic between subnets through a network virtual appliance, or to connect the virtual network to other virtual networks or on-premises networks.
6. **Optional:** Delete the resources that you create in this tutorial by completing the steps in [Delete resources](#).

Resource Manager template

You can deploy a virtual network by using an Azure Resource Manager template. To learn more about templates, see [What is Resource Manager](#). To access the template and to learn about its parameters, see the [Create a virtual network with two subnets](#) template. You can deploy the template by using the [portal](#), [Azure CLI](#), or [PowerShell](#).

Optional steps after you deploy the template:

1. Complete additional tutorials listed under [Next steps](#) to filter network traffic in and out of each subnet with network security groups, to route traffic between subnets through a network virtual appliance, or to connect the virtual network to other virtual networks or on-premises networks.

- Delete the resources that you create in this tutorial by completing the steps in any subsections of [Delete resources](#).

Azure portal

- In your browser, open the [template page](#).
- Click the **Deploy to Azure** button. If you're not already logged in to Azure, log in on the Azure portal login screen that appears.
- Sign in to the portal by using your [Azure account](#). If you don't have an Azure account, you can sign up for a [free trial](#).
- Enter the following values for the parameters:

PARAMETER	VALUE
Subscription	Select your subscription
Resource group	myResourceGroup
Location	Select a location
Vnet Name	myVnet
Vnet Address Prefix	10.0.0.0/16
Subnet1Prefix	10.0.0.0/24
Subnet1Name	Public
Subnet2Prefix	10.0.1.0/24
Subnet2Name	Private

- Agree to the terms and conditions, and then click **Purchase** to deploy the virtual network.

Azure CLI

- [Install and configure the Azure CLI](#). Ensure you have the most recent version of the Azure CLI installed. To get help for CLI commands, type `az <command> --help`. Rather than installing the CLI and its pre-requisites, you can use the Azure Cloud Shell. The Azure Cloud Shell is a free Bash shell that you can run directly within the Azure portal. The Cloud Shell has the Azure CLI preinstalled and configured to use with your account. To use the Cloud Shell, click the Cloud Shell >_ button at the top of the [portal](#), or just click the **Try it** button in the steps that follow.
- If running the CLI locally, log in to Azure with the `az login` command. If using the Cloud Shell, you're already logged in.
- To create a resource group for the virtual network, copy the following command and paste it into your CLI session:

```
az group create --name myResourceGroup --location eastus
```

- You can deploy the template by using one of the following parameters options:

- **Default parameter values**. Enter the following command:

```
az group deployment create --resource-group myResourceGroup --name VnetTutorial --template-uri https://raw.githubusercontent.com/azure/azure-quickstart-templates/master/101-vnet-two-subnets/azuredeploy.json`
```

- **Custom parameter values.** Download and modify the template before you deploy the template. You also can deploy the template by using parameters at the command line, or deploy the template with a separate parameters file. To download the template and parameters files, click the **Browse on GitHub** button on the [Create a virtual network with two subnets](#) template page. In GitHub, click the **azuredeploy.parameters.json** or **azuredeploy.json** file. Then, click the **Raw** button to display the file. In your browser, copy the contents of the file. Save the contents to a file on your computer. You can modify the parameter values in the template, or deploy the template with a separate parameters file.

To learn more about how to deploy templates by using these methods, type

```
az group deployment create --help .
```

PowerShell

1. Install the latest version of the PowerShell [AzureRm](#) module. If you're new to Azure PowerShell, see [Azure PowerShell overview](#).
2. In a PowerShell session, to sign in with your [Azure account](#), enter `login-azurermaccount`.
3. To create a resource group for the virtual network, enter the following command:

```
New-AzureRmResourceGroup -Name myResourceGroup -Location eastus
```

4. You can deploy the template by using one of the following parameters options:

- **Default parameter values.** Enter the following command:

```
New-AzureRmResourceGroupDeployment -Name VnetTutorial -ResourceGroupName myResourceGroup -TemplateUri https://raw.githubusercontent.com/azure/azure-quickstart-templates/master/101-vnet-two-subnets/azuredeploy.json
```

- **Custom parameter values.** Download and modify the template before you deploy it. You also can deploy the template by using parameters at the command line, or deploy the template with a separate parameters file. To download the template and parameters files, click the **Browse on GitHub** button on the [Create a virtual network with two subnets](#) template page. In GitHub, click the **azuredeploy.parameters.json** or **azuredeploy.json** file. Then, click the **Raw** button to display the file. In your browser, copy the contents of the file. Save the contents to a file on your computer. You can modify the parameter values in the template, or deploy the template with a separate parameters file.

To learn more about how to deploy templates by using these methods, type

```
Get-Help New-AzureRmResourceGroupDeployment .
```

Delete resources

When you finish this tutorial, you might want to delete the resources that you created, so that you don't incur usage charges. Deleting a resource group also deletes all resources that are in the resource group.

Azure portal

1. In the portal search box, enter **myResourceGroup**. In the search results, click **myResourceGroup**.
2. On the **myResourceGroup** blade, click the **Delete** icon.
3. To confirm the deletion, in the **TYPE THE RESOURCE GROUP NAME** box, enter **myResourceGroup**, and then

click **Delete**.

Azure CLI

In a CLI session, enter the following command:

```
az group delete --name myResourceGroup --yes
```

PowerShell

In a PowerShell session, enter the following command:

```
Remove-AzureRmResourceGroup -Name myResourceGroup -Force
```

Next steps

- To learn about all virtual network and subnet settings, see [Manage virtual networks](#) and [Manage virtual network subnets](#). You have various options for using virtual networks and subnets in a production environment to meet different requirements.
- Filter inbound and outbound subnet traffic by creating and applying [network security groups](#) to subnets.
- Route traffic between subnets through a network virtual appliance, by creating [user-defined routes](#) and apply the routes to each subnet.
- Create a [Windows](#) or a [Linux](#) virtual machine in an existing virtual network.
- Connect two virtual networks by creating a [virtual network peering](#) between the virtual networks.
- Connect the virtual network to an on-premises network by using a [VPN Gateway](#) or [Azure ExpressRoute](#) circuit.

How to open ports to a virtual machine with the Azure portal

12/13/2017 • 2 min to read • [Edit Online](#)

You open a port, or create an endpoint, to a virtual machine (VM) in Azure by creating a network filter on a subnet or VM network interface. You place these filters, which control both inbound and outbound traffic, on a Network Security Group attached to the resource that receives the traffic.

Let's use a common example of web traffic on port 80. Once you have a VM that is configured to serve web requests on the standard TCP port 80 (remember to start the appropriate services and open any OS firewall rules on the VM as well), you:

1. Create a Network Security Group.
2. Create an inbound rule allowing traffic with:
 - the destination port range of "80"
 - the source port range of "*" (allowing any source port)
 - a priority value of less 65,500 (to be higher in priority than the default catch-all deny inbound rule)
3. Associate the Network Security Group with the VM network interface or subnet.

You can create complex network configurations to secure your environment using Network Security Groups and rules. Our example uses only one or two rules that allow HTTP traffic or remote management. For more information, see the following '[More Information](#)' section or [What is a Network Security Group?](#)

Quick commands

You can also [perform these steps using Azure PowerShell](#).

First, create your Network Security Group. Select a resource group in the portal, choose **Add**, then search for and select **Network security group**:

The screenshot shows the Azure portal interface for creating a new resource. The top navigation bar has a 'Filter' dropdown set to 'Everything'. The main area is titled 'Essentials'. On the left, there are buttons for '+ Add', 'Columns', 'Delete', 'Refresh', and 'Move'. A search bar contains the text 'Network security group'. Below the search bar is a 'Results' section with a table. The table has columns 'NAME', 'TYPE', and 'LOCATION'. One row in the table is highlighted with a red box and labeled 'Network security group'. Another row below it is labeled 'Azure Networking Analytics (Preview)'.

Enter a name for your Network Security Group, select or create a resource group, and select a location. Select **Create** when finished:

Create network securit...

— □ ×

* Name

* Subscription

* Resource group 

Create new Use existing

* Location

Pin to dashboard

Create

Automation options

Select your new Network Security Group. Select 'Inbound security rules', then select the **Add** button to create a rule:

The screenshot shows the AWS Management Console interface for managing Network Security Groups. On the left, there's a sidebar with various navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, and a SETTINGS section. Within the SETTINGS section, 'Inbound security rules' is highlighted with a red box. At the top right, there are 'Add' and 'Default rules' buttons. The main content area has a search bar labeled 'Search inbound security rules'. Below the search bar is a table with two columns: 'PRIORITY' and 'NAME'. The message 'No results.' is displayed.

To create a rule that allows traffic:

- Select the **Basic** button. By default, the **Advanced** window provides some additional configuration options, such as to define a specific source IP block or port range.
- Choose a common **Service** from the drop-down menu, such as *HTTP*. You can also select *Custom* to provide a specific port to use.
- If desired, change the priority or name. The priority affects the order in which rules are applied - the lower the numerical value, the earlier the rule is applied.
- When you are ready, select **OK** to create the rule:

The screenshot shows the 'Add inbound security rule' dialog box. At the top, it says 'Add inbound security rule' and 'myNetworkSecurityGroup'. There's an 'Advanced' button with a crossed-out icon. The main form has four fields:

- Service**: A dropdown menu showing 'HTTP'.
- * Port range**: A text input field containing '80'.
- * Priority**: A dropdown menu showing '100'.
- * Name**: A text input field containing 'HTTP'.

At the bottom left is a blue 'OK' button.

Your final step is to associate your Network Security Group with a subnet or a specific network interface. Let's associate the Network Security Group with a subnet. Select **Subnets**, then choose **Associate**:

The screenshot shows the Azure portal interface for managing a Network Security Group named 'myNetworkSecurityGroup'. On the left, there's a sidebar with various navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Inbound security rules, Outbound security rules, Network interfaces, Subnets (which is highlighted with a red box), and Properties. The main content area is titled 'Subnets' and contains a search bar labeled 'Search subnets' and a table with columns 'NAME' and 'ADDRESS RANGE'. The table displays the message 'No results.'

Select your virtual network, and then select the appropriate subnet:

The screenshot shows two overlapping dialog boxes. The left dialog is titled 'Associate subnet' and shows a list with one item: '1 Virtual network myVnet' with a green checkmark next to it. The right dialog is titled 'Choose subnet' and shows a list with one item: '2 Subnet Choose a subnet' with a blue arrow pointing to it. Both dialogs have standard window controls (minimize, maximize, close).

You have now created a Network Security Group, created an inbound rule that allows traffic on port 80, and associated it with a subnet. Any VMs you connect to that subnet are reachable on port 80.

More information on Network Security Groups

The quick commands here allow you to get up and running with traffic flowing to your VM. Network Security Groups provide many great features and granularity for controlling access to your resources. You can read more about [creating a Network Security Group and ACL rules here](#).

For highly available web applications, you should place your VMs behind an Azure Load Balancer. The load balancer distributes traffic to VMs, with a Network Security Group that provides traffic filtering. For more information, see [How to load balance Linux virtual machines in Azure to create a highly available application](#).

Next steps

In this example, you created a simple rule to allow HTTP traffic. You can find information on creating more detailed

environments in the following articles:

- [Azure Resource Manager overview](#)
- [What is a Network Security Group \(NSG\)?](#)

How to open ports and endpoints to a VM in Azure using PowerShell

12/13/2017 • 2 min to read • [Edit Online](#)

You open a port, or create an endpoint, to a virtual machine (VM) in Azure by creating a network filter on a subnet or VM network interface. You place these filters, which control both inbound and outbound traffic, on a Network Security Group attached to the resource that receives the traffic.

Let's use a common example of web traffic on port 80. Once you have a VM that is configured to serve web requests on the standard TCP port 80 (remember to start the appropriate services and open any OS firewall rules on the VM as well), you:

1. Create a Network Security Group.
2. Create an inbound rule allowing traffic with:
 - the destination port range of "80"
 - the source port range of "*" (allowing any source port)
 - a priority value of less 65,500 (to be higher in priority than the default catch-all deny inbound rule)
3. Associate the Network Security Group with the VM network interface or subnet.

You can create complex network configurations to secure your environment using Network Security Groups and rules. Our example uses only one or two rules that allow HTTP traffic or remote management. For more information, see the following '[More Information](#)' section or [What is a Network Security Group?](#)

Quick commands

To create a Network Security Group and ACL rules you need [the latest version of Azure PowerShell installed](#). You can also [perform these steps using the Azure portal](#).

Log in to your Azure account:

```
Login-AzureRmAccount
```

In the following examples, replace parameter names with your own values. Example parameter names included *myResourceGroup*, *myNetworkSecurityGroup*, and *myVnet*.

Create a rule with [New-AzureRmNetworkSecurityRuleConfig](#). The following example creates a rule named *myNetworkSecurityGroupRule* to allow *tcp* traffic on port 80:

```
$httprule = New-AzureRmNetworkSecurityRuleConfig `  
    -Name "myNetworkSecurityGroupRule" `  
    -Description "Allow HTTP" `  
    -Access "Allow" `  
    -Protocol "Tcp" `  
    -Direction "Inbound" `  
    -Priority "100" `  
    -SourceAddressPrefix "Internet" `  
    -SourcePortRange * `  
    -DestinationAddressPrefix * `  
    -DestinationPortRange 80
```

Next, create your Network Security group with [New-AzureRmNetworkSecurityGroup](#) and assign the HTTP rule

you just created as follows. The following example creates a Network Security Group named *myNetworkSecurityGroup*:

```
$nsg = New-AzureRmNetworkSecurityGroup `  
    -ResourceGroupName "myResourceGroup" `  
    -Location "EastUS" `  
    -Name "myNetworkSecurityGroup" `  
    -SecurityRules $httprule
```

Now let's assign your Network Security Group to a subnet. The following example assigns an existing virtual network named *myVnet* to the variable `$vnet` with [Get-AzureRmVirtualNetwork](#):

```
$vnet = Get-AzureRmVirtualNetwork `  
    -ResourceGroupName "myResourceGroup" `  
    -Name "myVnet"
```

Associate your Network Security Group with your subnet with [Set-AzureRmVirtualNetworkSubnetConfig](#). The following example associates the subnet named *mySubnet* with your Network Security Group:

```
$subnetPrefix = $vnet.Subnets | ?{$_ . Name -eq 'mySubnet'}`  
  
Set-AzureRmVirtualNetworkSubnetConfig `  
    -VirtualNetwork $vnet `  
    -Name "mySubnet" `  
    -AddressPrefix $subnetPrefix.AddressPrefix `  
    -NetworkSecurityGroup $nsg
```

Finally, update your virtual network with [Set-AzureRmVirtualNetwork](#) in order for your changes to take effect:

```
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet
```

More information on Network Security Groups

The quick commands here allow you to get up and running with traffic flowing to your VM. Network Security Groups provide many great features and granularity for controlling access to your resources. You can read more about [creating a Network Security Group and ACL rules here](#).

For highly available web applications, you should place your VMs behind an Azure Load Balancer. The load balancer distributes traffic to VMs, with a Network Security Group that provides traffic filtering. For more information, see [How to load balance Linux virtual machines in Azure to create a highly available application](#).

Next steps

In this example, you created a simple rule to allow HTTP traffic. You can find information on creating more detailed environments in the following articles:

- [Azure Resource Manager overview](#)
- [What is a Network Security Group \(NSG\)?](#)
- [Azure Resource Manager Overview for Load Balancers](#)

Create a VM with a static public IP address using the Azure portal

6/27/2017 • 2 min to read • [Edit Online](#)

You can create virtual machines (VMs) in Azure and expose them to the public Internet by using a public IP address. By default, Public IPs are dynamic and the address associated to them may change when the VM is deleted or stopped/deallocated. To guarantee that the VM always uses the same public IP address, you need to create a static Public IP.

Before you can implement static Public IPs in VMs, it is necessary to understand when you can use static Public IPs, and how they are used. Read the [IP addressing overview](#) to learn more about IP addressing in Azure.

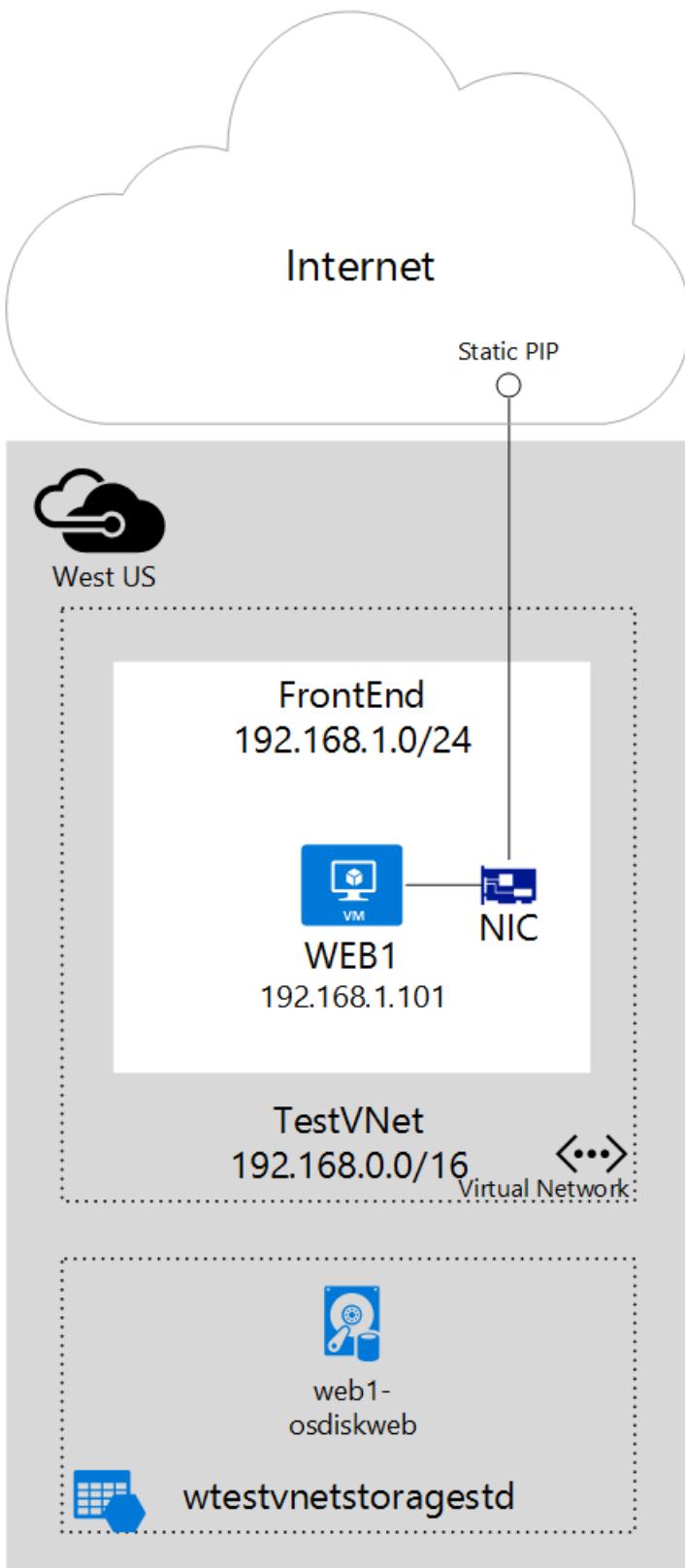
NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager](#) and [classic](#). This article covers using the Resource Manager deployment model, which Microsoft recommends for most new deployments instead of the classic deployment model.

Scenario

This document will walk through a deployment that uses a static public IP address allocated to a virtual machine (VM). In this scenario, you have a single VM with its own static public IP address. The VM is part of a subnet named **FrontEnd** and also has a static private IP address (**192.168.1.101**) in that subnet.

You may need a static IP address for web servers that require SSL connections in which the SSL certificate is linked to an IP address.



You can follow the steps below to deploy the environment shown in the figure above.

Create a VM with a static public IP

To create a VM with a static public IP address in the Azure portal, complete the following steps:

1. From a browser, navigate to the [Azure portal](#) and, if necessary, sign in with your Azure account.
2. On the top left hand corner of the portal, click **New >> Compute > Windows Server 2012 R2 Datacenter**.
3. In the **Select a deployment model** list, select **Resource Manager** and click **Create**.
4. In the **Basics** blade, enter the VM information as shown below, and then click **OK**.

* Name
WEB1 ✓

* User name
adminUser ✓

* Password
***** ✓

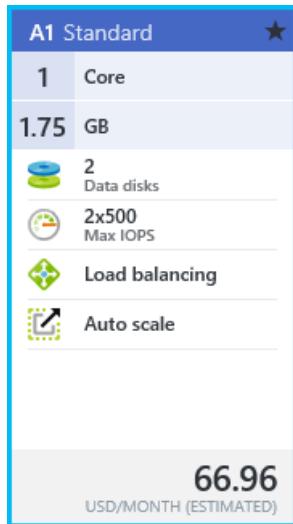
* Subscription
Microsoft Azure Internal Consumption

* Resource group
PublicIPTest ✕ ✓

Select existing

* Location
East US 2

5. In the **Choose a size** blade, click **A1 Standard** as shown below, and then click **Select**.



6. In the **Settings** blade, click **Public IP address**, then in the **Create public IP address** blade, under **Assignment**, click **Static** as shown below. And then click **OK**.

* Name
WEB1 ✓

Assignment
Dynamic Static

7. In the **Settings** blade, click **OK**.
8. Review the **Summary** blade, as shown below, and then click **OK**.

Basics	
Subscription	Microsoft Azure Internal Consumption
Resource group	(new) PublicIPTest
Location	East US 2
Settings	
Computer name	WEB1
User name	adminUser
Size	Standard A1
Disk type	Standard
Storage account	(new) publiciptest3109
Virtual network	(new) PublicIPTest
Subnet	(new) default (10.6.0.0/24)
Public IP address	(new) WEB1
Network security group	(new) WEB1
Availability set	None
Diagnostics	Enabled
Diagnostics storage account	(new) publiciptest3109

9. Notice the new tile in your dashboard.



10. Once the VM is created, the **Settings** blade will be displayed as shown below

 WEB1
Virtual machine

Settings Connect Start Restart Stop Delete

Essentials

Resource group PublicIPTest	Computer name WEB1
Status Running	Size Standard A1 (1 core, 1.75 GB memory)
Location East US 2	Operating system Windows
Subscription name Microsoft Azure Internal Consumption	Public IP address/DNS name label 104.208.232.131/<none>
Subscription ID 628dad04-b5d1-4f10-b3a4-dc61d88cf97c	Virtual network/subnet PublicIPTest/default

All settings →

Monitoring

CPU percentage

100%
80%
60%
40%
20%
0%

11 AM 11:15 AM 11:30 AM 11:45 AM

CPU UTILIZATION
0.08 %

Add tiles +

Add a group +

Settings

WEB1

INVESTIGATE

- Audit logs >
- Boot diagnostics >
- Reset password >
- Troubleshoot >
- New support request >

MANAGE

- Properties >
- Disks >
- Network interfaces >
- Availability set >
- Extensions >
- Size >

MONITOR

- Alert rules >
- Diagnostics >

RESOURCE MANAGEMENT

- Users >
- Tags >

Create and manage a Windows virtual machine that has multiple NICs

12/15/2017 • 7 min to read • [Edit Online](#)

Virtual machines (VMs) in Azure can have multiple virtual network interface cards (NICs) attached to them. A common scenario is to have different subnets for front-end and back-end connectivity, or a network dedicated to a monitoring or backup solution. This article details how to create a VM that has multiple NICs attached to it. You also learn how to add or remove NICs from an existing VM. Different [VM sizes](#) support a varying number of NICs, so size your VM accordingly.

Prerequisites

Make sure that you have the [latest Azure PowerShell version installed and configured](#).

In the following examples, replace example parameter names with your own values. Example parameter names include *myResourceGroup*, *myVnet*, and *myVM*.

Create a VM with multiple NICs

First, create a resource group. The following example creates a resource group named *myResourceGroup* in the *EastUs* location:

```
New-AzureRmResourceGroup -Name "myResourceGroup" -Location "EastUS"
```

Create virtual network and subnets

A common scenario is for a virtual network to have two or more subnets. One subnet may be for front-end traffic, the other for back-end traffic. To connect to both subnets, you then use multiple NICs on your VM.

1. Define two virtual network subnets with [New-AzureRmVirtualNetworkSubnetConfig](#). The following example defines the subnets for *mySubnetFrontEnd* and *mySubnetBackEnd*:

```
$mySubnetFrontEnd = New-AzureRmVirtualNetworkSubnetConfig -Name "mySubnetFrontEnd" `  
    -AddressPrefix "192.168.1.0/24"  
$mySubnetBackEnd = New-AzureRmVirtualNetworkSubnetConfig -Name "mySubnetBackEnd" `  
    -AddressPrefix "192.168.2.0/24"
```

2. Create your virtual network and subnets with [New-AzureRmVirtualNetwork](#). The following example creates a virtual network named *myVnet*:

```
$myVnet = New-AzureRmVirtualNetwork -ResourceGroupName "myResourceGroup" `  
    -Location "EastUs" `  
    -Name "myVnet" `  
    -AddressPrefix "192.168.0.0/16" `  
    -Subnet $mySubnetFrontEnd,$mySubnetBackEnd
```

Create multiple NICs

Create two NICs with [New-AzureRmNetworkInterface](#). Attach one NIC to the front-end subnet and one NIC to the back-end subnet. The following example creates NICs named *myNic1* and *myNic2*:

```

$frontEnd = $myVnet.Subnets | ?{$_ .Name -eq 'mySubnetFrontEnd'}
$myNic1 = New-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" ` 
    -Name "myNic1" ` 
    -Location "EastUs" ` 
    -SubnetId $frontEnd.Id

$backEnd = $myVnet.Subnets | ?{$_ .Name -eq 'mySubnetBackEnd'}
$myNic2 = New-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" ` 
    -Name "myNic2" ` 
    -Location "EastUs" ` 
    -SubnetId $backEnd.Id

```

Typically you also create a [network security group](#) to filter network traffic to the VM and a [load balancer](#) to distribute traffic across multiple VMs.

Create the virtual machine

Now start to build your VM configuration. Each VM size has a limit for the total number of NICs that you can add to a VM. For more information, see [Windows VM sizes](#).

1. Set your VM credentials to the `$cred` variable as follows:

```
$cred = Get-Credential
```

2. Define your VM with [New-AzureRmVMConfig](#). The following example defines a VM named *myVM* and uses a VM size that supports more than two NICs (*Standard_DS3_v2*):

```
$vmConfig = New-AzureRmVMConfig -VMName "myVM" -VMSize "Standard_DS3_v2"
```

3. Create the rest of your VM configuration with [Set-AzureRmVMOperatingSystem](#) and [Set-AzureRmVMSourceImage](#). The following example creates a Windows Server 2016 VM:

```

$vmConfig = Set-AzureRmVMOperatingSystem -VM $vmConfig ` 
    -Windows ` 
    -ComputerName "myVM" ` 
    -Credential $cred ` 
    -ProvisionVMAgent ` 
    -EnableAutoUpdate
$vmConfig = Set-AzureRmVMSourceImage -VM $vmConfig ` 
    -PublisherName "MicrosoftWindowsServer" ` 
    -Offer "WindowsServer" ` 
    -Skus "2016-Datacenter" ` 
    -Version "latest"

```

4. Attach the two NICs that you previously created with [Add-AzureRmVMNetworkInterface](#):

```

$vmConfig = Add-AzureRmVMNetworkInterface -VM $vmConfig -Id $myNic1.Id -Primary
$vmConfig = Add-AzureRmVMNetworkInterface -VM $vmConfig -Id $myNic2.Id

```

5. Finally, create your VM with [New-AzureRmVM](#):

```
New-AzureRmVM -VM $vmConfig -ResourceGroupName "myResourceGroup" -Location "EastUs"
```

Add a NIC to an existing VM

To add a virtual NIC to an existing VM, you deallocate the VM, add the virtual NIC, then start the VM. Different [VM sizes](#) support a varying number of NICs, so size your VM accordingly. If needed, you can [resize a VM](#).

1. Deallocate the VM with [Stop-AzureRmVM](#). The following example deallocates the VM named *myVM* in *myResourceGroup*:

```
Stop-AzureRmVM -Name "myVM" -ResourceGroupName "myResourceGroup"
```

2. Get the existing configuration of the VM with [Get-AzureRmVm](#). The following example gets information for the VM named *myVM* in *myResourceGroup*:

```
$vm = Get-AzureRmVm -Name "myVM" -ResourceGroupName "myResourceGroup"
```

3. The following example creates a virtual NIC with [New-AzureRmNetworkInterface](#) named *myNic3* that is attached to *mySubnetBackEnd*. The virtual NIC is then attached to the VM named *myVM* in *myResourceGroup* with [Add-AzureRmVMNetworkInterface](#):

```
# Get info for the back end subnet
$myVnet = Get-AzureRmVirtualNetwork -Name "myVnet" -ResourceGroupName "myResourceGroup"
$backEnd = $myVnet.Subnets | ?{$_ . Name -eq 'mySubnetBackEnd'}`

# Create a virtual NIC
$myNic3 = New-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" ` 
    -Name "myNic3" ` 
    -Location "EastUs" ` 
    -SubnetId $backEnd.Id

# Get the ID of the new virtual NIC and add to VM
$nicId = (Get-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" -Name "MyNic3").Id
Add-AzureRmVMNetworkInterface -VM $vm -Id $nicId | Update-AzureRmVm -ResourceGroupName "myResourceGroup"
```

Primary virtual NICs

One of the NICs on a multi-NIC VM needs to be primary. If one of the existing virtual NICs on the VM is already set as primary, you can skip this step. The following example assumes that two virtual NICs are now present on a VM and you wish to add the first NIC ([0]) as the primary:

```
# List existing NICs on the VM and find which one is primary
$vm.NetworkProfile.NetworkInterfaces

# Set NIC 0 to be primary
$vm.NetworkProfile.NetworkInterfaces[0].Primary = $true
$vm.NetworkProfile.NetworkInterfaces[1].Primary = $false

# Update the VM state in Azure
Update-AzureRmVM -VM $vm -ResourceGroupName "myResourceGroup"
```

4. Start the VM with [Start-AzureRmVm](#):

```
Start-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVM"
```

Remove a NIC from an existing VM

To remove a virtual NIC from an existing VM, you deallocate the VM, remove the virtual NIC, then start the VM.

1. Deallocate the VM with [Stop-AzureRmVM](#). The following example deallocates the VM named *myVM* in

myResourceGroup:

```
Stop-AzureRmVM -Name "myVM" -ResourceGroupName "myResourceGroup"
```

2. Get the existing configuration of the VM with [Get-AzureRmVm](#). The following example gets information for the VM named *myVM* in *myResourceGroup*:

```
$vm = Get-AzureRmVm -Name "myVM" -ResourceGroupName "myResourceGroup"
```

3. Get information about the NIC remove with [Get-AzureRmNetworkInterface](#). The following example gets information about *myNic3*:

```
# List existing NICs on the VM if you need to determine NIC name  
$vm.NetworkProfile.NetworkInterfaces  
  
$nicId = (Get-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" -Name "myNic3").Id
```

4. Remove the NIC with [Remove-AzureRmVMNetworkInterface](#) and then update the VM with [Update-AzureRmVm](#). The following example removes *myNic3* as obtained by `$nicId` in the preceding step:

```
Remove-AzureRmVMNetworkInterface -VM $vm -NetworkInterfaceIDs $nicId | `  
Update-AzureRmVm -ResourceGroupName "myResourceGroup"
```

5. Start the VM with [Start-AzureRmVm](#):

```
Start-AzureRmVM -Name "myVM" -ResourceGroupName "myResourceGroup"
```

Create multiple NICs with templates

Azure Resource Manager templates provide a way to create multiple instances of a resource during deployment, such as creating multiple NICs. Resource Manager templates use declarative JSON files to define your environment. For more information, see [overview of Azure Resource Manager](#). You can use *copy* to specify the number of instances to create:

```
"copy": {  
    "name": "multiplenics",  
    "count": "[parameters('count')]"  
}
```

For more information, see [creating multiple instances by using copy](#).

You can also use `copyIndex()` to append a number to a resource name. You can then create *myNic1*, *MyNic2* and so on. The following code shows an example of appending the index value:

```
"name": "[concat('myNic', copyIndex())]",
```

You can read a complete example of [creating multiple NICs by using Resource Manager templates](#).

Configure guest OS for multiple NICs

Azure assigns a default gateway to the first (primary) network interface attached to the virtual machine. Azure does

not assign a default gateway to additional (secondary) network interfaces attached to a virtual machine. Therefore, you are unable to communicate with resources outside the subnet that a secondary network interface is in, by default. Secondary network interfaces can, however, communicate with resources outside their subnet, though the steps to enable communication are different for different operating systems.

1. From a Windows command prompt, run the `route print` command, which returns output similar to the following output for a virtual machine with two attached network interfaces:

```
=====
Interface List
3...00 0d 3a 10 92 ce ....Microsoft Hyper-V Network Adapter #3
7...00 0d 3a 10 9b 2a ....Microsoft Hyper-V Network Adapter #4
=====
```

In this example, **Microsoft Hyper-V Network Adapter #4** (interface 7) is the secondary network interface that doesn't have a default gateway assigned to it.

2. From a command prompt, run the `ipconfig` command to see which IP address is assigned to the secondary network interface. In this example, 192.168.2.4 is assigned to interface 7. No default gateway address is returned for the secondary network interface.
3. To route all traffic destined for addresses outside the subnet of the secondary network interface to the gateway for the subnet, run the following command:

```
route add -p 0.0.0.0 MASK 0.0.0.0 192.168.2.1 METRIC 5015 IF 7
```

The gateway address for the subnet is the first IP address (ending in .1) in the address range defined for the subnet. If you don't want to route all traffic outside the subnet, you could add individual routes to specific destinations, instead. For example, if you only wanted to route traffic from the secondary network interface to the 192.168.3.0 network, you enter the command:

```
route add -p 192.168.3.0 MASK 255.255.255.0 192.168.2.1 METRIC 5015 IF 7
```

4. To confirm successful communication with a resource on the 192.168.3.0 network, for example, enter the following command to ping 192.168.3.4 using interface 7 (192.168.2.4):

```
ping 192.168.3.4 -S 192.168.2.4
```

You may need to open ICMP through the Windows firewall of the device you're pinging with the following command:

```
netsh advfirewall firewall add rule name=Allow-ping protocol=icmpv4 dir=in action=allow
```

5. To confirm the added route is in the route table, enter the `route print` command, which returns output similar to the following text:

```
=====
Active Routes:
Network Destination      Netmask          Gateway        Interface Metric
          0.0.0.0          0.0.0.0        192.168.1.1    192.168.1.4      15
          0.0.0.0          0.0.0.0        192.168.2.1    192.168.2.4    5015
=====
```

The route listed with 192.168.1.1 under **Gateway**, is the route that is there by default for the primary

network interface. The route with `192.168.2.1` under **Gateway**, is the route you added.

Next steps

Review [Windows VM sizes](#) when you're trying to create a VM that has multiple NICs. Pay attention to the maximum number of NICs that each VM size supports.

Create a fully qualified domain name in the Azure portal for a Windows VM

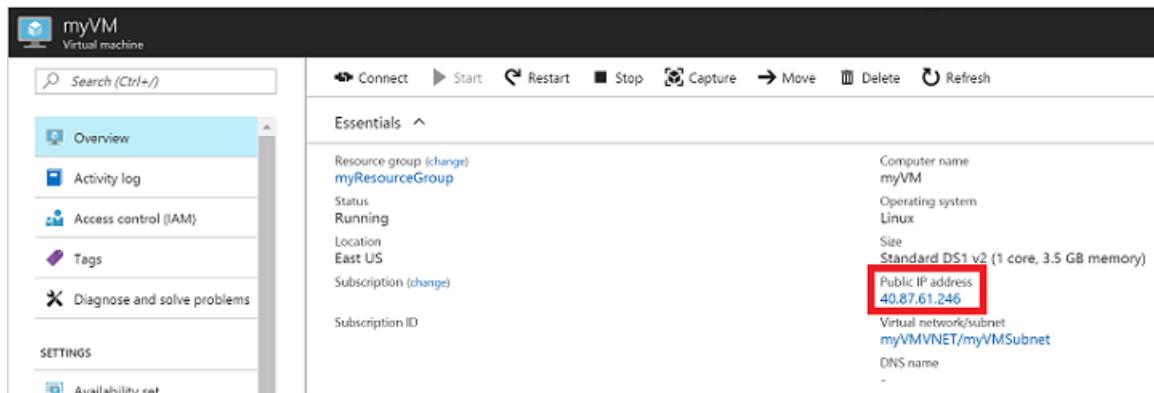
12/14/2017 • 1 min to read • [Edit Online](#)

When you create a virtual machine (VM) in the [Azure portal](#), a public IP resource for the virtual machine is automatically created. You use this IP address to remotely access the VM. Although the portal does not create a [fully qualified domain name](#), or FQDN, you can create one once the VM is created. This article demonstrates the steps to create a DNS name or FQDN.

Create a FQDN

This article assumes that you have already created a VM. If needed, you can [create a VM in the portal](#) or [with Azure PowerShell](#). Follow these steps once your VM is up and running:

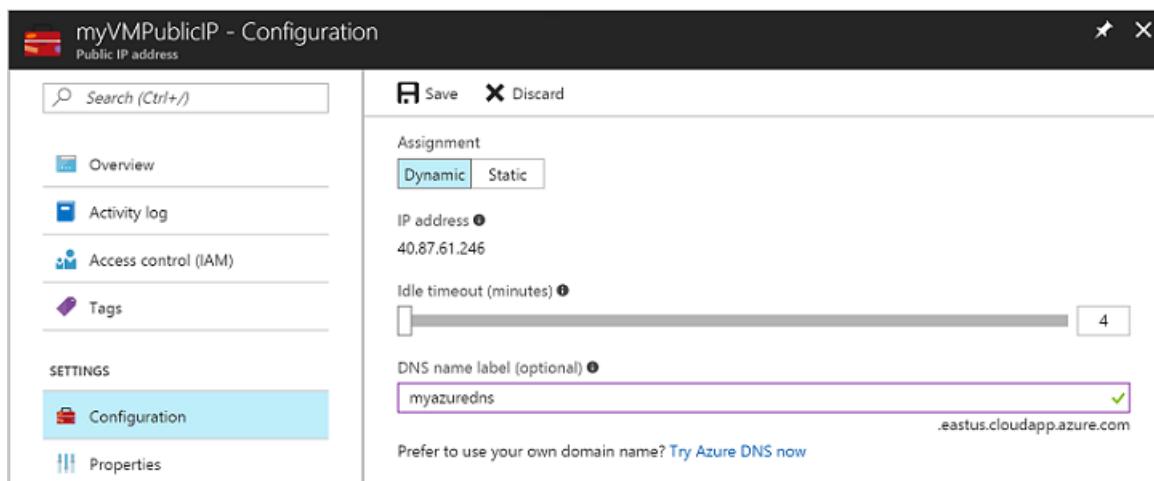
1. Select your VM in the portal. The *DNS name* is currently blank. Select **Public IP address**:



The screenshot shows the Azure portal interface for a virtual machine named "myVM". The left sidebar has "Overview" selected. The main pane displays the "Essentials" section with the following details:

Resource group (change) myResourceGroup	Computer name myVM
Status Running	Operating system Linux
Location East US	Size Standard DS1 v2 (1 core, 3.5 GB memory)
Subscription (change)	Public IP address 40.87.61.246
Subscription ID	Virtual network/subnet myVMVNET/myVMSubnet
	DNS name -

2. Enter the desired DNS name and then select **Save**.



The screenshot shows the "myVMPublicIP - Configuration" blade for the Public IP address. The left sidebar has "Configuration" selected. The main pane shows the following configuration:

Assignment	Dynamic
IP address	40.87.61.246
Idle timeout (minutes)	4
DNS name label (optional)	myazuredns

Below the configuration, there is a note: "Prefer to use your own domain name? [Try Azure DNS now](#)".

3. To return to the VM overview blade, close the *Public IP address* blade. Verify that the *DNS name* is now shown.

The screenshot shows the Azure portal interface for a virtual machine named 'myVM'. The left sidebar has a search bar and links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, SETTINGS, and Availability set. The main area is titled 'Essentials' and shows the following details:

Resource group (change) myResourceGroup	Computer name myVM
Status Running	Operating system Linux
Location East US	Size Standard DS1 v2 (1 core, 3.5 GB memory)
Subscription (change)	Public IP address 40.87.61.246
Subscription ID	Virtual network/subnet myVMVNET/myVMSubnet
	DNS name myazuredns.eastus.cloudapp.azure.com

You can now connect remotely to the VM using this DNS name such as for Remote Desktop Protocol (RDP).

Next steps

Now that your VM has a public IP and DNS name, you can deploy common application frameworks or services such as IIS, SQL, or SharePoint.

You can also read more about [using Resource Manager](#) for tips on building your Azure deployments.

How Azure DNS works with other Azure services

11/29/2017 • 2 min to read • [Edit Online](#)

Azure DNS is a hosted DNS management and name resolution service. This allows you to create public DNS names for the other applications and services you have deployed in Azure. Creating a name for an Azure service in your custom domain is as simple as adding a record of the correct type for your service.

- For dynamically allocated IP addresses, you must create a DNS CNAME record that maps to the DNS name that Azure created for your service. DNS standards prevent you from using a CNAME record for the zone apex.
- For statically allocated IP addresses, you can create a DNS A record using any name, including a *naked domain* name at the zone apex.

The following table outlines the supported record types that can be used for various Azure services. As you can see from this table, Azure DNS only supports DNS records for Internet-facing network resources. Azure DNS cannot be used for name resolution of internal, private addresses.

AZURE SERVICE	NETWORK INTERFACE	DESCRIPTION
Application Gateway	Front-end Public IP	You can create a DNS A or CNAME record.
Load Balancer	Front-end Public IP	You can create a DNS A or CNAME record. Load Balancer can have an IPv6 Public IP address that is dynamically assigned. Therefore, you must create a CNAME record for an IPv6 address.
Traffic Manager	Public name	You can only create a CNAME that maps to the trafficmanager.net name assigned to your Traffic Manager profile. For more information, see How Traffic Manager works .
Cloud Service	Public IP	For statically allocated IP addresses, you can create a DNS A record. For dynamically allocated IP addresses, you must create a CNAME record that maps to the cloudapp.net name.
App Service	External IP	For external IP addresses, you can create a DNS A record. Otherwise, you must create a CNAME record that maps to the azurewebsites.net name. For more information, see Map a custom domain name to an Azure app
Resource Manager VMs	Public IP	Resource Manager VMs can have Public IP addresses. A VM with a Public IP address may also be behind a load balancer. You can create a DNS A or CNAME record for the Public address. This custom name can be used to bypass the VIP on the load balancer.

AZURE SERVICE	NETWORK INTERFACE	DESCRIPTION
Classic VMs	Public IP	Classic VMs created using PowerShell or CLI can be configured with a dynamic or static (reserved) virtual address. You can create a DNS CNAME or A record, respectively.

Virtual machine extensions and features for Windows

11/8/2017 • 7 min to read • [Edit Online](#)

Azure virtual machine extensions are small applications that provide post-deployment configuration and automation tasks on Azure virtual machines. For example, if a virtual machine requires software installation, anti-virus protection, or Docker configuration, a VM extension can be used to complete these tasks. Azure VM extensions can be run by using the Azure CLI, PowerShell, Azure Resource Manager templates, and the Azure portal. Extensions can be bundled with a new virtual machine deployment or run against any existing system.

This document provides an overview of virtual machine extensions, prerequisites for using virtual machine extensions, and guidance on how to detect, manage, and remove virtual machine extensions. This document provides generalized information because many VM extensions are available, each with a potentially unique configuration. Extension-specific details can be found in each document specific to the individual extension.

Use cases and samples

There are many different Azure VM extensions available, each with a specific use case. Some example use cases are:

- Apply PowerShell Desired State configurations to a virtual machine by using the DSC extension for Windows. For more information, see [Azure Desired State Configuration extension](#).
- Configure virtual machine monitoring by using the Microsoft Monitoring Agent VM extension. For more information, see [Connect Azure virtual machines to Log Analytics](#).
- Configure monitoring of your Azure infrastructure with the Datadog extension. For more information, see the [Datadog blog](#).
- Configure an Azure virtual machine by using Chef. For more information, see [Automating Azure virtual machine deployment with Chef](#).

In addition to process-specific extensions, a Custom Script extension is available for both Windows and Linux virtual machines. The Custom Script extension for Windows allows any PowerShell script to be run on a virtual machine. This is useful when you're designing Azure deployments that require configuration beyond what native Azure tooling can provide. For more information, see [Windows VM Custom Script extension](#).

Prerequisites

Each virtual machine extension may have its own set of prerequisites. For instance, the Docker VM extension has a prerequisite of a supported Linux distribution. Requirements of individual extensions are detailed in the extension-specific documentation.

Azure VM agent

The Azure VM agent manages interaction between an Azure virtual machine and the Azure fabric controller. The VM agent is responsible for many functional aspects of deploying and managing Azure virtual machines, including running VM extensions. The Azure VM agent is preinstalled on Azure Marketplace images and can be installed on supported operating systems.

For information on supported operating systems and installation instructions, see [Azure virtual machine agent](#).

Discover VM extensions

Many different VM extensions are available for use with Azure virtual machines. To see a complete list, run the following command with the Azure Resource Manager PowerShell module. Make sure to specify the desired

location when you're running this command.

```
Get-AzureRmVmImagePublisher -Location WestUS | `  
Get-AzureRmVMEExtensionImageType | `  
Get-AzureRmVMEExtensionImage | Select Type, Version
```

Run VM extensions

Azure virtual machine extensions can be run on existing virtual machines, which is useful when you need to make configuration changes or recover connectivity on an already deployed VM. VM extensions can also be bundled with Azure Resource Manager template deployments. By using extensions with Resource Manager templates, you can enable Azure virtual machines to be deployed and configured without the need for post-deployment intervention.

The following methods can be used to run an extension against an existing virtual machine.

PowerShell

Several PowerShell commands exist for running individual extensions. To see a list, run the following PowerShell commands.

```
get-command Set-AzureRM*Extension* -Module AzureRM.Compute
```

This provides output similar to the following:

CommandType	Name	Version	Source
Cmdlet	Set-AzureRmVMAccessExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMADDomainExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMAEMExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMBackupExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMBginfoExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMChefExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMCustomScriptExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMDiagnosticsExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMDiskEncryptionExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMDscExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMEExtension	2.2.0	AzureRM.Compute
Cmdlet	Set-AzureRmVMSqlServerExtension	2.2.0	AzureRM.Compute

The following example uses the Custom Script extension to download a script from a GitHub repository onto the target virtual machine and then run the script. For more information on the Custom Script extension, see [Custom Script extension overview](#).

```
Set-AzureRmVMCustomScriptExtension -ResourceGroupName "myResourceGroup" `  
-VMName "myVM" -Name "myCustomScript" `  
-FileUri "https://raw.githubusercontent.com/neilpeterson/nepeters-azure-templates/master/windows-custom-`  
script-simple/support-scripts/Create-File.ps1" `  
-Run "Create-File.ps1" -Location "West US"
```

In this example, the VM Access extension is used to reset the administrative password of a Windows virtual machine. For more information on the VM Access extension, see [Reset Remote Desktop service in a Windows VM](#).

```
$cred=Get-Credential

Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" -VMName "myVM" -Name "myVMAccess" `

-Location WestUS -UserName $cred.GetNetworkCredential().Username `

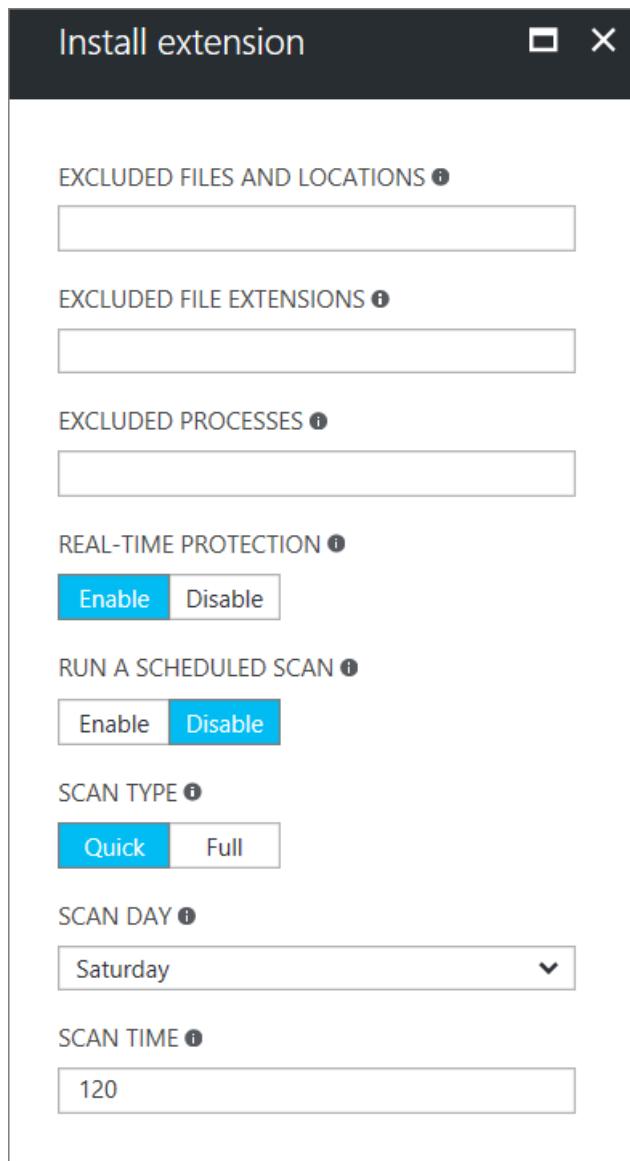
-Password $cred.GetNetworkCredential().Password -typeHandlerVersion "2.0"
```

The `Set-AzureRmVMExtension` command can be used to start any VM extension. For more information, see the [Set-AzureRmVMExtension reference](#).

Azure portal

A VM extension can be applied to an existing virtual machine through the Azure portal. To do so, select the virtual machine you want to use, choose **Extensions**, and click **Add**. This provides a list of available extensions. Select the one you want and follow the steps in the wizard.

The following image shows the installation of the Microsoft Antimalware extension from the Azure portal.



Azure Resource Manager templates

VM extensions can be added to an Azure Resource Manager template and executed with the deployment of the template. Deploying extensions with a template is useful for creating fully configured Azure deployments. For example, the following JSON is taken from a Resource Manager template that deploys a set of load-balanced virtual machines and an Azure SQL database, and then installs a .NET Core application on each VM. The VM extension takes care of the software installation.

For more information, see the [full Resource Manager template](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "extensions",
  "name": "config-app",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'),copyindex())]",
    "[variables('musicstoresqlName')]"
  ],
  "tags": {
    "displayName": "config-app"
  },
  "properties": {
    "publisher": "Microsoft.Compute",
    "type": "CustomScriptExtension",
    "typeHandlerVersion": "1.4",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "fileUris": [
        "https://raw.githubusercontent.com/Microsoft/dotnet-core-sample-templates/master/dotnet-core-music-
windows/scripts/configure-music-app.ps1"
      ]
    },
    "protectedSettings": {
      "commandToExecute": "[concat('powershell -ExecutionPolicy Unrestricted -File configure-music-app.ps1 -',
        'user ',parameters('adminUsername'),' -password ',parameters('adminPassword'),' -sqlserver',
        ',variables('musicstoresqlName'),'.database.windows.net')]"
    }
  }
}
```

For more information, see [Authoring Azure Resource Manager templates with Windows VM extensions](#).

Secure VM extension data

When you're running a VM extension, it may be necessary to include sensitive information such as credentials, storage account names, and storage account access keys. Many VM extensions include a protected configuration that encrypts data and only decrypts it inside the target virtual machine. Each extension has a specific protected configuration schema that will be detailed in extension-specific documentation.

The following example shows an instance of the Custom Script extension for Windows. Notice that the command to execute includes a set of credentials. In this example, the command to execute will not be encrypted.

```
{
  "apiVersion": "2015-06-15",
  "type": "extensions",
  "name": "config-app",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'),copyindex())]",
    "[variables('musicstoresqlName')]"
  ],
  "tags": {
    "displayName": "config-app"
  },
  "properties": {
    "publisher": "Microsoft.Compute",
    "type": "CustomScriptExtension",
    "typeHandlerVersion": "1.4",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "fileUris": [
        "https://raw.githubusercontent.com/Microsoft/dotnet-core-sample-templates/master/dotnet-core-music-
windows/scripts/configure-music-app.ps1"
      ],
      "commandToExecute": "[concat('powershell -ExecutionPolicy Unrestricted -File configure-music-app.ps1 -u
ser ',parameters('adminUsername'),' -password ',parameters('adminPassword'),' -sqlserver '
,variables('musicstoresqlName'),'.database.windows.net')]"
    }
  }
}
```

Secure the execution string by moving the **command to execute** property to the **protected** configuration.

```
{
  "apiVersion": "2015-06-15",
  "type": "extensions",
  "name": "config-app",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'),copyindex())]",
    "[variables('musicstoresqlName')]"
  ],
  "tags": {
    "displayName": "config-app"
  },
  "properties": {
    "publisher": "Microsoft.Compute",
    "type": "CustomScriptExtension",
    "typeHandlerVersion": "1.4",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "fileUris": [
        "https://raw.githubusercontent.com/Microsoft/dotnet-core-sample-templates/master/dotnet-core-music-
windows/scripts/configure-music-app.ps1"
      ]
    },
    "protectedSettings": {
      "commandToExecute": "[concat('powershell -ExecutionPolicy Unrestricted -File configure-music-app.ps1 -u
ser ',parameters('adminUsername'),' -password ',parameters('adminPassword'),' -sqlserver '
,variables('musicstoresqlName'),'.database.windows.net')]"
    }
  }
}
```

Troubleshoot VM extensions

Each VM extension may have specific troubleshooting steps. For instance, when you're using the Custom Script extension, script execution details can be found locally on the virtual machine on which the extension was run. Any extension-specific troubleshooting steps are detailed in extension-specific documentation.

The following troubleshooting steps apply to all virtual machine extensions.

View extension status

After a virtual machine extension has been run against a virtual machine, use the following PowerShell command to return extension status. Replace example parameter names with your own values. The `Name` parameter takes the name given to the extension at execution time.

```
Get-AzureRmVMExtension -ResourceGroupName myResourceGroup -VMName myVM -Name myExtensionName
```

The output looks like the following:

```
ResourceGroupName      : myResourceGroup
VMName                : myVM
Name                  : myExtensionName
Location              : westus
Etag                  : null
Publisher              : Microsoft.Azure.Extensions
ExtensionType         : DockerExtension
TypeHandlerVersion    : 1.0
Id                    :
/subscriptions/mySubscription/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM/extensions/myExtensionName
PublicSettings         :
ProtectedSettings      :
ProvisioningState     : Succeeded
Statuses              :
SubStatuses            :
AutoUpgradeMinorVersion : False
ForceUpdateTag         :
```

Extension execution status can also be found in the Azure portal. To view the status of an extension, select the virtual machine, choose **Extensions**, and select the desired extension.

Rerun VM extensions

There may be cases in which a virtual machine extension needs to be rerun. You can do this by removing the extension and then rerunning the extension with an execution method of your choice. To remove an extension, run the following command with the Azure PowerShell module. Replace example parameter names with your own values.

```
Remove-AzureRmVMExtension -ResourceGroupName myResourceGroup -VMName myVM -Name myExtensionName
```

An extension can also be removed using the Azure portal. To do so:

1. Select a virtual machine.
2. Select **Extensions**.
3. Choose the desired extension.
4. Select **Uninstall**.

Common VM extensions reference

Extension Name	Description	More Information
Custom Script Extension for Windows	Run scripts against an Azure virtual machine	Custom Script Extension for Windows
DSC Extension for Windows	PowerShell DSC (Desired State Configuration) Extension	DSC Extension for Windows
Azure Diagnostics Extension	Manage Azure Diagnostics	Azure Diagnostics Extension
Azure VM Access Extension	Manage users and credentials	VM Access Extension for Linux

Custom Script Extension for Windows

12/7/2017 • 3 min to read • [Edit Online](#)

The Custom Script Extension downloads and executes scripts on Azure virtual machines. This extension is useful for post deployment configuration, software installation, or any other configuration / management task. Scripts can be downloaded from Azure storage or GitHub, or provided to the Azure portal at extension run time. The Custom Script extension integrates with Azure Resource Manager templates, and can also be run using the Azure CLI, PowerShell, Azure portal, or the Azure Virtual Machine REST API.

This document details how to use the Custom Script Extension using the Azure PowerShell module, Azure Resource Manager templates, and details troubleshooting steps on Windows systems.

Prerequisites

Operating System

The Custom Script Extension for Windows can be run against Windows 10 Client, Windows Server 2008 R2, 2012, 2012 R2, and 2016 releases.

Script Location

The script needs to be stored in Azure Blob storage, or any other location accessible through a valid URL.

Internet Connectivity

The Custom Script Extension for Windows requires that the target virtual machine is connected to the internet.

Extension schema

The following JSON shows the schema for the Custom Script Extension. The extension requires a script location (Azure Storage or other location with valid URL), and a command to execute. If using Azure Storage as the script source, an Azure storage account name and account key is required. These items should be treated as sensitive data and specified in the extensions protected setting configuration. Azure VM extension protected setting data is encrypted, and only decrypted on the target virtual machine.

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Compute/virtualMachines/extensions",
  "name": "config-app",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'), copyindex())]",
    "[variables('musicstoresqlName')]"
  ],
  "tags": {
    "displayName": "config-app"
  },
  "properties": {
    "publisher": "Microsoft.Compute",
    "type": "CustomScriptExtension",
    "typeHandlerVersion": "1.9",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "fileUris": [
        "script location"
      ]
    },
    "protectedSettings": {
      "commandToExecute": "myExecutionCommand",
      "storageAccountName": "myStorageAccountName",
      "storageAccountKey": "myStorageAccountKey"
    }
  }
}
```

Property values

NAME	VALUE / EXAMPLE
apiVersion	2015-06-15
publisher	Microsoft.Compute
type	extensions
typeHandlerVersion	1.9
fileUris (e.g.)	https://raw.githubusercontent.com/Microsoft/dotnet-core-sample-templates/master/dotnet-core-music-windows/scripts/configure-music-app.ps1
commandToExecute (e.g.)	powershell -ExecutionPolicy Unrestricted -File configure-music-app.ps1
storageAccountName (e.g.)	examplestorageacct

NAME	VALUE / EXAMPLE
storageAccountKey (e.g.)	TmJK/1N3AbAZ3q/+hOXoi/l73zOqsaxXHqa9Y83/v5UpXQp2DQIBuv2Tifp60cE/OaHsJZmQZ7teQfczQj8hg==

Note – these property names are case sensitive. Use the names as seen above to avoid deployment issues.

Template deployment

Azure VM extensions can be deployed with Azure Resource Manager templates. The JSON schema detailed in the previous section can be used in an Azure Resource Manager template to run the Custom Script Extension during an Azure Resource Manager template deployment. A sample template that includes the Custom Script Extension can be found here, [GitHub](#).

PowerShell deployment

The `Set-AzureRmVMCustomScriptExtension` command can be used to add the Custom Script extension to an existing virtual machine. For more information, see [Set-AzureRmVMCustomScriptExtension](#).

```
Set-AzureRmVMCustomScriptExtension -ResourceGroupName myResourceGroup ` 
    -VMName myVM ` 
    -Location myLocation ` 
    -FileUri myURL ` 
    -Run 'myScript.ps1' ` 
    -Name DemoScriptExtension
```

Troubleshoot and support

Troubleshoot

Data about the state of extension deployments can be retrieved from the Azure portal, and by using the Azure PowerShell module. To see the deployment state of extensions for a given VM, run the following command.

```
Get-AzureRmVMExtension -ResourceGroupName myResourceGroup -VMName myVM -Name myExtensionName
```

Extension execution output is logged to files found under the following directory on the target virtual machine.

```
C:\WindowsAzure\Logs\Plugins\Microsoft.Compute.CustomScriptExtension
```

The specified files are downloaded into the following directory on the target virtual machine.

```
C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1.*\Downloads\<n>
```

where `<n>` is a decimal integer which may change between executions of the extension. The `1.*` value matches the actual, current `typeHandlerVersion` value of the extension. For example, the actual directory could be `C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1.8\Downloads\2`.

When executing the `commandToExecute` command, the extension will have set this directory (e.g., `...\Downloads\2`) as the current working directory. This enables the use of relative paths to locate the files downloaded via the `fileUris` property. See the table below for examples.

Since the absolute download path may vary over time, it is better to opt for relative script/file paths in the `commandToExecute` string, whenever possible. For example:

```
"commandToExecute": "powershell.exe . . . -File './scripts/myscript.ps1'"
```

Path information after the first URI segment is retained for files downloaded via the `fileUris` property list. As shown in the table below, downloaded files are mapped into download subdirectories to reflect the structure of the `fileUris` values.

Examples of Downloaded Files

URI IN FILEURIS	RELATIVE DOWNLOADED LOCATION	ABSOLUTE DOWNLOADED LOCATION *
<code>https://someAcct.blob.core.windows.net/aContainer/script ./scripts/myscript.ps1</code>		<code>C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1</code>
<code>https://someAcct.blob.core.windows.net/aContainer/topLevel ./topLevel.ps1</code>		<code>C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1</code>

* As above, the absolute directory paths will change over the lifetime of the VM, but not within a single execution of the CustomScript extension.

Support

If you need more help at any point in this article, you can contact the Azure experts on the [MSDN Azure and Stack Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select Get support. For information about using Azure Support, read the [Microsoft Azure support FAQ](#).

OMS virtual machine extension for Windows

12/14/2017 • 3 min to read • [Edit Online](#)

Operations Management Suite (OMS) provides monitoring, alerting, and alert remediation capabilities across cloud and on-premises assets. The OMS Agent virtual machine extension for Windows is published and supported by Microsoft. The extension installs the OMS agent on Azure virtual machines, and enrolls virtual machines into an existing OMS workspace. This document details the supported platforms, configurations, and deployment options for the OMS virtual machine extension for Windows.

Prerequisites

Operating system

The OMS Agent extension for Windows can be run against Windows Server 2008 R2, 2012, 2012 R2, and 2016 releases.

Azure Security Center

Azure Security Center automatically provisions the OMS agent and connects it with the default log analytics workspace of the Azure subscription. If you are using Azure Security Center, do not run through the steps in this document. Doing so overwrites the configured workspace and break the connection with Azure Security Center.

Internet connectivity

The OMS Agent extension for Windows requires that the target virtual machine is connected to the internet.

Extension schema

The following JSON shows the schema for the OMS Agent extension. The extension requires the workspace Id and workspace key from the target OMS workspace, these can be found in the OMS portal. Because the workspace key should be treated as sensitive data, it should be stored in a protected setting configuration. Azure VM extension protected setting data is encrypted, and only decrypted on the target virtual machine. Note that **workspaceId** and **workspaceKey** are case-sensitive.

```
{
  "type": "extensions",
  "name": "OMSExtension",
  "apiVersion": "[variables('apiVersion')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
  ],
  "properties": {
    "publisher": "Microsoft.EnterpriseCloud.Monitoring",
    "type": "MicrosoftMonitoringAgent",
    "typeHandlerVersion": "1.0",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "workspaceId": "myWorkSpaceId"
    },
    "protectedSettings": {
      "workspaceKey": "myWorkspaceKey"
    }
  }
}
```

Property values

NAME	VALUE / EXAMPLE
apiVersion	2015-06-15
publisher	Microsoft.EnterpriseCloud.Monitoring
type	MicrosoftMonitoringAgent
typeHandlerVersion	1.0
workspaceId (e.g.)	6f680a37-00c6-41c7-a93f-1437e3462574
workspaceKey (e.g.)	z4bU3p1/GrnWpQkky4gdabWXAhbWSTz70hm4m2Xt92XI+rS RgE8qVvRhsGo9TXffbrTahyrw35W0pOqQAU7uQ==

Template deployment

Azure VM extensions can be deployed with Azure Resource Manager templates. The JSON schema detailed in the previous section can be used in an Azure Resource Manager template to run the OMS Agent extension during an Azure Resource Manager template deployment. A sample template that includes the OMS Agent VM extension can be found on the [Azure Quick Start Gallery](#).

The JSON for a virtual machine extension can be nested inside the virtual machine resource, or placed at the root or top level of a Resource Manager JSON template. The placement of the JSON affects the value of the resource name and type. For more information, see [Set name and type for child resources](#).

The following example assumes the OMS extension is nested inside the virtual machine resource. When nesting the extension resource, the JSON is placed in the `"resources": []` object of the virtual machine.

```
{  
    "type": "extensions",  
    "name": "OMSExtension",  
    "apiVersion": "[variables('apiVersion')]",  
    "location": "[resourceGroup().location]",  
    "dependsOn": [  
        "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"  
    ],  
    "properties": {  
        "publisher": "Microsoft.EnterpriseCloud.Monitoring",  
        "type": "MicrosoftMonitoringAgent",  
        "typeHandlerVersion": "1.0",  
        "autoUpgradeMinorVersion": true,  
        "settings": {  
            "workspaceId": "myWorkSpaceId"  
        },  
        "protectedSettings": {  
            "workspaceKey": "myWorkspaceKey"  
        }  
    }  
}
```

When placing the extension JSON at the root of the template, the resource name includes a reference to the parent virtual machine, and the type reflects the nested configuration.

```
{
  "type": "Microsoft.Compute/virtualMachines/extensions",
  "name": "<parentVmResource>/OMSExtension",
  "apiVersion": "[variables('apiVersion')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
  ],
  "properties": {
    "publisher": "Microsoft.EnterpriseCloud.Monitoring",
    "type": "MicrosoftMonitoringAgent",
    "typeHandlerVersion": "1.0",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "workspaceId": "myWorkSpaceId"
    },
    "protectedSettings": {
      "workspaceKey": "myWorkspaceKey"
    }
  }
}
```

PowerShell deployment

The `Set-AzureRmVMExtension` command can be used to deploy the OMS Agent virtual machine extension to an existing virtual machine. Before running the command, the public and private configurations need to be stored in a PowerShell hash table.

```
$PublicSettings = @{"workspaceId" = "myWorkSpaceId"}
$ProtectedSettings = @{"workspaceKey" = "myWorkspaceKey"}

Set-AzureRmVMExtension -ExtensionName "Microsoft.EnterpriseCloud.Monitoring" ` 
  -ResourceGroupName "myResourceGroup" ` 
  -VMName "myVM" ` 
  -Publisher "Microsoft.EnterpriseCloud.Monitoring" ` 
  -ExtensionType "MicrosoftMonitoringAgent" ` 
  -TypeHandlerVersion 1.0 ` 
  -Settings $PublicSettings ` 
  -ProtectedSettings $ProtectedSettings ` 
  -Location WestUS `
```

Troubleshoot and support

Troubleshoot

Data about the state of extension deployments can be retrieved from the Azure portal, and by using the Azure PowerShell module. To see the deployment state of extensions for a given VM, run the following command using the Azure PowerShell module.

```
Get-AzureRmVMExtension -ResourceGroupName myResourceGroup -VMName myVM -Name myExtensionName
```

Extension execution output is logged to files found in the following directory:

```
C:\WindowsAzure\Logs\Plugins\Microsoft.EnterpriseCloud.Monitoring.MicrosoftMonitoringAgent\
```

Support

If you need more help at any point in this article, you can contact the Azure experts on the [MSDN Azure and Stack](#)

[Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select Get support. For information about using Azure Support, read the [Microsoft Azure support FAQ](#).

Introduction to the Azure Desired State Configuration extension handler

6/27/2017 • 6 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

The Azure VM Agent and associated Extensions are part of the Microsoft Azure Infrastructure Services. VM Extensions are software components that extend the VM functionality and simplify various VM management operations. For example, the VMAccess extension can be used to reset an administrator's password, or the Custom Script extension can be used to execute a script on the VM.

This article introduces the PowerShell Desired State Configuration (DSC) Extension for Azure VMs as part of the Azure PowerShell SDK. You can use new cmdlets to upload and apply a PowerShell DSC configuration on an Azure VM enabled with the PowerShell DSC extension. The PowerShell DSC extension calls into PowerShell DSC to enact the received DSC configuration on the VM. This functionality is also available through the Azure portal.

Prerequisites

Local machine To interact with the Azure VM extension, you need to use either the Azure portal or the Azure PowerShell SDK.

Guest Agent The Azure VM that is configured by the DSC configuration needs to be an OS that supports either Windows Management Framework (WMF) 4.0 or 5.0. The full list of supported OS versions can be found at the [DSC Extension Version History](#).

Terms and concepts

This guide presumes familiarity with the following concepts:

Configuration - A DSC configuration document.

Node - A target for a DSC configuration. In this document, "node" always refers to an Azure VM.

Configuration Data - A .psd1 file containing environmental data for a configuration

Architectural overview

The Azure DSC extension uses the Azure VM Agent framework to deliver, enact, and report on DSC configurations running on Azure VMs. The DSC extension expects a .zip file containing at least a configuration document, and a set of parameters provided either through the Azure PowerShell SDK or through the Azure portal.

When the extension is called for the first time, it runs an installation process. This process installs a version of the Windows Management Framework (WMF) using the following logic:

1. If the Azure VM OS is Windows Server 2016, no action is taken. Windows Server 2016 already has the latest version of PowerShell installed.
2. If the `wmfVersion` property is specified, that version of the WMF is installed unless it is incompatible with the VM's OS.

3. If no `wmfVersion` property is specified, the latest applicable version of the WMF is installed.

Installation of the WMF requires a reboot. After reboot, the extension downloads the .zip file specified in the `modulesUrl` property. If this location is in Azure blob storage, a SAS token can be specified in the `sasToken` property to access the file. After the .zip is downloaded and unpacked, the configuration function defined in `configurationFunction` is run to generate the MOF file. The extension then runs `Start-DscConfiguration -Force` on the generated MOF file. The extension captures output and writes it back out to the Azure Status Channel. From this point on, the DSC LCM handles monitoring and correction as normal.

PowerShell cmdlets

PowerShell cmdlets can be used with Azure Resource Manager or the classic deployment model to package, publish, and monitor DSC extension deployments. The following cmdlets listed are the classic deployment modules, but "Azure" can be replaced with "AzureRm" to use the Azure Resource Manager model. For example, `Publish-AzureVMDscConfiguration` uses the classic deployment model, where `Publish-AzureRmVMDscConfiguration` uses Azure Resource Manager.

`Publish-AzureVMDscConfiguration` takes in a configuration file, scans it for dependent DSC resources, and creates a .zip file containing the configuration and DSC resources needed to enact the configuration. It can also create the package locally using the `-ConfigurationArchivePath` parameter. Otherwise, it publishes the .zip file to Azure blob storage and secures it with a SAS token.

The .zip file created by this cmdlet has the .ps1 configuration script at the root of the archive folder. Resources have the module folder placed in the archive folder.

`Set-AzureVMDscExtension` injects the settings needed by the PowerShell DSC extension into a VM configuration object. In the classic deployment model, the VM changes must be applied to an Azure VM with `Update-AzureVM`.

`Get-AzureVMDscExtension` retrieves the DSC extension status of a particular VM.

`Get-AzureVMDscExtensionStatus` retrieves the status of the DSC configuration enacted by the DSC extension handler. This action can be performed on a single VM, or group of VMs.

`Remove-AzureVMDscExtension` removes the extension handler from a given virtual machine. This cmdlet does **not** remove the configuration, uninstall the WMF, or change the applied settings on the virtual machine. It only removes the extension handler.

Key differences in ASM and Azure Resource Manager cmdlets

- Azure Resource Manager cmdlets are synchronous. ASM cmdlets are asynchronous.
- `ResourceGroupName`, `VMName`, `ArchiveStorageAccountName`, `Version`, and `Location` are all required parameters in Azure Resource Manager.
- `ArchiveResourceGroupName` is a new optional parameter for Azure Resource Manager. You can specify this parameter when your storage account belongs to a different resource group than the one where the virtual machine is created.
- `ConfigurationArchive` is called `ArchiveBlobName` in Azure Resource Manager
- `ContainerName` is called `ArchiveContainerName` in Azure Resource Manager
- `StorageEndpointSuffix` is called `ArchiveStorageEndpointSuffix` in Azure Resource Manager
- The `AutoUpdate` switch has been added to Azure Resource Manager to enable automatic updating of the extension handler to the latest version as and when it is available. Note this parameter has the potential to cause reboots on the VM when a new version of the WMF is released.

Azure portal functionality

Browse to a VM. Under Settings -> General click "Extensions." A new pane is created. Click "Add" and select

PowerShell DSC.

The portal needs input. **Configuration Modules or Script**: This field is mandatory. Requires a .ps1 file containing a configuration script, or a .zip file with a .ps1 configuration script at the root, and all dependent resources in module folders within the .zip. It can be created with the

`Publish-AzureVMdscConfiguration -ConfigurationArchivePath` cmdlet included in the Azure PowerShell SDK. The .zip file is uploaded into your user blob storage secured by a SAS token.

Configuration Data PSD1 File: This field is optional. If your configuration requires a configuration data file in .psd1, use this field to select it and upload it to your user blob storage, where it is secured by a SAS token.

Module-Qualified Name of Configuration: .ps1 files can have multiple configuration functions. Enter the name of the configuration .ps1 script followed by a " and the name of the configuration function. For example, if your .ps1 script has the name "configuration.ps1", and the configuration is "IisInstall", you would enter:

`configuration.ps1\IisInstall`

Configuration Arguments: If the configuration function takes arguments, enter them in here in the format

`argumentName1=value1,argumentName2=value2` . Note this format is a different format than how configuration

arguments are accepted through PowerShell cmdlets or Resource Manager templates.

Getting started

The Azure DSC extension takes in DSC configuration documents and enacts them on Azure VMs. A simple example of a configuration follows. Save it locally as "IisInstall.ps1":

```
configuration IISInstall
{
    node "localhost"
    {
        WindowsFeature IIS
        {
            Ensure = "Present"
            Name = "Web-Server"
        }
    }
}
```

The following steps place the IisInstall.ps1 script on the specified VM, execute the configuration, and report back on status.

Classic model

```

#Azure PowerShell cmdlets are required
Import-Module Azure

#Use an existing Azure Virtual Machine, 'DscDemo1'
$demoVM = Get-AzureVM DscDemo1

#Publish the configuration script into user storage.
Publish-AzureVMDscConfiguration -ConfigurationPath ".\IisInstall.ps1" -StorageContext $storageContext -Verbose -Force

#Set the VM to run the DSC configuration
Set-AzureVMDscExtension -VM $demoVM -ConfigurationArchive "IisInstall.ps1.zip" -StorageContext $storageContext -ConfigurationName "IisInstall" -Verbose

#Update the configuration of an Azure Virtual Machine
$demoVM | Update-AzureVM -Verbose

#check on status
Get-AzureVMDscExtensionStatus -VM $demovm -Verbose

```

Azure Resource Manager model

```

$resourceGroup = "dscVmDemo"
$location = "westus"
$vmName = "myVM"
$storageName = "demostorage"
#Publish the configuration script into user storage
Publish-AzureRmVMDscConfiguration -ConfigurationPath .\iisInstall.ps1 -ResourceGroupName $resourceGroup -StorageAccountName $storageName -force
#Set the VM to run the DSC configuration
Set-AzureRmVmDscExtension -Version 2.21 -ResourceGroupName $resourceGroup -VMName $vmName -ArchiveStorageAccountName $storageName -ArchiveBlobName iisInstall.ps1.zip -AutoUpdate:$true -ConfigurationName "IISInstall"

```

Logging

Logs are placed in:

C:\WindowsAzure\Logs\Plugins\Microsoft.Powershell.DSC[Version Number]

Next steps

For more information about PowerShell DSC, [visit the PowerShell documentation center](#).

Examine the [Azure Resource Manager template for the DSC extension](#).

To find additional functionality you can manage with PowerShell DSC, [browse the PowerShell gallery](#) for more DSC resources.

For details on passing sensitive parameters into configurations, see [Manage credentials securely with the DSC extension handler](#).

Passing credentials to the Azure DSC extension handler

6/27/2017 • 2 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

This article covers the Desired State Configuration extension for Azure. An overview of the DSC extension handler can be found at [Introduction to the Azure Desired State Configuration extension handler](#).

Passing in credentials

As a part of the configuration process, you may need to set up user accounts, access services, or install a program in a user context. To do these things, you need to provide credentials.

DSC allows for parameterized configurations in which credentials are passed into the configuration and securely stored in MOF files. The Azure Extension Handler simplifies credential management by providing automatic management of certificates.

Consider the following DSC configuration script that creates a local user account with the specified password:

user_configuration.ps1

```
configuration Main
{
    param(
        [Parameter(Mandatory=$true)]
        [ValidateNotNullOrEmpty()]
        [PSCredential]
        $Credential
    )
    Node localhost {
        User LocalUserAccount
        {
            Username = $Credential.UserName
            Password = $Credential
            Disabled = $false
            Ensure = "Present"
            FullName = "Local User Account"
            Description = "Local User Account"
            PasswordNeverExpires = $true
        }
    }
}
```

It is important to include *node localhost* as part of the configuration. If this statement is missing, the following steps do not work as the extension handler specifically looks for the node *localhost* statement. It is also important to include the typecast *[PsCredential]*, as this specific type triggers the extension to encrypt the credential.

Publish this script to blob storage:

```
Publish-AzureVMDscConfiguration -ConfigurationPath .\user_configuration.ps1
```

Set the Azure DSC extension and provide the credential:

```
$configurationName = "Main"
$configurationArguments = @{ Credential = Get-Credential }
$configurationArchive = "user_configuration.ps1.zip"
$vm = Get-AzureVM "example-1"

$vm = Set-AzureVMDSCExtension -VM $vm -ConfigurationArchive $configurationArchive
-ConfigurationName $configurationName -ConfigurationArgument @configurationArguments

$vm | Update-AzureVM
```

How credentials are secured

Running this code prompts for a credential. Once it is provided, it is stored in memory briefly. When it is published with `Set-AzureVmDscExtension` cmdlet, it is transmitted over HTTPS to the VM, where Azure stores it encrypted on disk, using the local VM certificate. Then it is briefly decrypted in memory and re-encrypted to pass it to DSC.

This behavior is different than [using secure configurations without the extension handler](#). The Azure environment gives a way to transmit configuration data securely via certificates. When using the DSC extension handler, there is no need to provide `$CertificatePath` or a `$CertificateID` / `$Thumbprint` entry in `ConfigurationData`.

Next steps

For more information on the Azure DSC extension handler, see [Introduction to the Azure Desired State Configuration extension handler](#).

Examine the [Azure Resource Manager template for the DSC extension](#).

For more information about PowerShell DSC, [visit the PowerShell documentation center](#).

To find additional functionality you can manage with PowerShell DSC, [browse the PowerShell gallery](#) for more DSC resources.

Windows VMSS and Desired State Configuration with Azure Resource Manager templates

7/24/2017 • 6 min to read • [Edit Online](#)

This article describes the Resource Manager template for the [Desired State Configuration extension handler](#).

Template example for a Windows VM

The following snippet goes into the Resource section of the template.

```
"name": "Microsoft.Powershell.DSC",
"type": "extensions",
"location": "[resourceGroup().location]",
"apiVersion": "2015-06-15",
"dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
],
"properties": {
    "publisher": "Microsoft.PowerShell",
    "type": "DSC",
    "typeHandlerVersion": "2.20",
    "autoUpgradeMinorVersion": true,
    "forceUpdateTag": "[parameters('dscExtensionUpdateTagVersion')]",
    "settings": {
        "configuration": {
            "url": "[concat(parameters('_artifactsLocation'), '/',
variables('dscExtensionArchiveFolder'), '/', variables('dscExtensionArchiveFileName'))]",
            "script": "dscExtension.ps1",
            "function": "Main"
        },
        "configurationArguments": {
            "nodeName": "[variables('vmName')]"
        }
    },
    "protectedSettings": {
        "configurationUrlSasToken": "[parameters('_artifactsLocationSasToken')]"
    }
}
```

Template example for Windows VMSS

A VMSS node has a "properties" section with the "VirtualMachineProfile", "extensionProfile" attribute. DSC is added under "extensions".

```

"extensionProfile": {
    "extensions": [
        {
            "name": "Microsoft.PowerShell.DSC",
            "properties": {
                "publisher": "Microsoft.PowerShell",
                "type": "DSC",
                "typeHandlerVersion": "2.20",
                "autoUpgradeMinorVersion": true,
                "forceUpdateTag": "[parameters('DscExtensionUpdateTagVersion')]",
                "settings": {
                    "configuration": {
                        "url": "[concat(parameters('_artifactsLocation'), '/',
variables('DscExtensionArchiveFolder'), '/', variables('DscExtensionArchiveFileName'))]",
                        "script": "DscExtension.ps1",
                        "function": "Main"
                    },
                    "configurationArguments": {
                        "nodeName": "localhost"
                    }
                },
                "protectedSettings": {
                    "configurationUrlsAsToken": "[parameters('_artifactsLocationSasToken')]"
                }
            }
        }
    ]
}

```

Detailed Settings Information

The following schema is for the settings portion of the Azure DSC extension in an Azure Resource Manager template.

```

"settings": {
    "wmfVersion": "latest",
    "configuration": {
        "url": "http://validURLToConfigLocation",
        "script": "ConfigurationScript.ps1",
        "function": "ConfigurationFunction"
    },
    "configurationArguments": {
        "argument1": "Value1",
        "argument2": "Value2"
    },
    "configurationData": {
        "url": "https://foo.psd1"
    },
    "privacy": {
        "dataCollection": "enable"
    },
    "advancedOptions": {
        "downloadMappings": {
            "customWmfLocation": "http://myWMFlocation"
        }
    }
},
"protectedSettings": {
    "configurationArguments": {
        "parameterOfTypePSCredential1": {
            "userName": "UsernameValue1",
            "password": "PasswordValue1"
        },
        "parameterOfTypePSCredential2": {
            "userName": "UsernameValue2",
            "password": "PasswordValue2"
        }
    },
    "configurationUrlsAsToken": "?g!bber1sht0k3n",
    "configurationDataUrlsAsToken": "?dataAcC355T0k3N"
}
}

```

Details

PROPERTY NAME	TYPE	DESCRIPTION
settings.wmfVersion	string	Specifies the version of the Windows Management Framework that should be installed on your VM. Setting this property to 'latest' installs the most updated version of WMF. The only current possible values for this property are '4.0' , '5.0' , '5.0PP' , and 'latest' . These possible values are subject to updates. The default value is 'latest'.

PROPERTY NAME	TYPE	DESCRIPTION
settings.configuration.url	string	Specifies the URL location from which to download your DSC configuration zip file. If the URL provided requires a SAS token for access, you need to set the protectedSettings.configurationUrlSasToken property to the value of your SAS token. This property is required if settings.configuration.script and/or settings.configuration.function are defined.
settings.configuration.script	string	Specifies the file name of the script that contains the definition of your DSC configuration. This script must be in the root folder of the zip file downloaded from the URL specified by the configuration.url property. This property is required if settings.configuration.url and/or settings.configuration.script are defined.
settings.configuration.function	string	Specifies the name of your DSC configuration. The configuration named must be contained in the script defined by configuration.script. This property is required if settings.configuration.url and/or settings.configuration.function are defined.
settings.configurationArguments	Collection	Defines any parameters you would like to pass to your DSC configuration. This property is not encrypted.
settings.configurationData.url	string	Specifies the URL from which to download your configuration data (.psd1) file to use as input for your DSC configuration. If the URL provided requires a SAS token for access, you need to set the protectedSettings.configurationDataUrlsAsToken property to the value of your SAS token.
settings.privacy.dataEnabled	string	Enables or disables telemetry collection. The only possible values for this property are 'Enable' , 'Disable' , '' , or \$null . Leaving this property blank or null enables telemetry. The default value is '' . More Information
settings.advancedOptions.downloadMappings	Collection	Defines alternate locations from which to download the WMF. More Information
protectedSettings.configurationArguments	Collection	Defines any parameters you would like to pass to your DSC configuration. This property is encrypted.

PROPERTY NAME	TYPE	DESCRIPTION
protectedSettings.configurationUrlSasToken	string	Specifies the SAS token to access the URL defined by configuration.url. This property is encrypted.
protectedSettings.configurationDataUrlSasToken	string	Specifies the SAS token to access the URL defined by configurationData.url. This property is encrypted.

Settings vs. ProtectedSettings

All settings are saved in a settings text file on the VM. Properties under 'settings' are public properties because they are not encrypted in the settings text file. Properties under 'protectedSettings' are encrypted with a certificate and are not shown in plain text in this file on the VM.

If the configuration needs credentials, they can be included in protectedSettings:

```
"protectedSettings": {
    "configurationArguments": {
        "parameterOfTypePSCredential1": {
            "userName": "UsernameValue1",
            "password": "PasswordValue1"
        }
    }
}
```

Example

The following example derives from the "Getting Started" section of the [DSC Extension Handler Overview page](#). This example uses Resource Manager templates instead of cmdlets to deploy the extension. Save the "IisInstall.ps1" configuration, place it in a .ZIP file, and upload the file in an accessible URL. This example uses Azure blob storage, but it is possible to download .ZIP files from any arbitrary location.

In the Azure Resource Manager template, the following code instructs the VM to download the correct file and run the appropriate PowerShell function:

```
"settings": {
    "configuration": {
        "url": "https://demo.blob.core.windows.net/",
        "script": "IisInstall.ps1",
        "function": "IISInstall"
    }
},
"protectedSettings": {
    "configurationUrlSasToken": "odLPL/U1p9lvcnp..."
}
```

Updating from the Previous Format

Any settings in the previous format (containing the public properties ModulesUrl, ConfigurationFunction, SasToken, or Properties) automatically adapt to the current format and run just as they did before.

The following schema is what the previous settings schema looked like:

```

"settings": {
    "WMFVersion": "latest",
    "ModulesUrl": "https://UrlToZipContainingConfigurationScript.ps1.zip",
    "SasToken": "SAS Token if ModulesUrl points to private Azure Blob Storage",
    "ConfigurationFunction": "ConfigurationScript.ps1\\ConfigurationFunction",
    "Properties": {
        "ParameterToConfigurationFunction1": "Value1",
        "ParameterToConfigurationFunction2": "Value2",
        "ParameterOfTypePSCredential1": {
            "UserName": "UsernameValue1",
            "Password": "PrivateSettingsRef:Key1"
        },
        "ParameterOfTypePSCredential2": {
            "UserName": "UsernameValue2",
            "Password": "PrivateSettingsRef:Key2"
        }
    }
},
"protectedSettings": {
    "Items": {
        "Key1": "PasswordValue1",
        "Key2": "PasswordValue2"
    },
    "DataBlobUri": "https://UrlToConfigurationDataWithOptionalSasToken.psd1"
}
}

```

Here's how the previous format adapts to the current format:

PROPERTY NAME	PREVIOUS SCHEMA EQUIVALENT
settings.wmfVersion	settings.WMFVersion
settings.configuration.url	settings.ModulesUrl
settings.configuration.script	First part of settings.ConfigurationFunction (before '\\')
settings.configuration.function	Second part of settings.ConfigurationFunction (after '\\')
settings.configurationArguments	settings.Properties
settings.configurationData.url	protectedSettings.DataBlobUri (without SAS token)
settings.privacy.dataEnabled	settings.Privacy.DataEnabled
settings.advancedOptions.downloadMappings	settings.AdvancedOptions.DownloadMappings
protectedSettings.configurationArguments	protectedSettings.Properties
protectedSettings.configurationUrlSasToken	settings.SasToken
protectedSettings.configurationDataUrlSasToken	SAS token from protectedSettings.DataBlobUri

Troubleshooting - Error Code 1100

Error Code 1100 indicates that there is a problem with the user input to the DSC extension. The text of these errors is variable and may change. Here are some of the errors you may run into and how you can fix them.

Invalid Values

"Privacy.dataCollection is '{0}'. The only possible values are '', 'Enable', and 'Disable'" "WmfVersion is '{0}'. Only possible values are ... and 'latest'"

Problem: A provided value is not allowed.

Solution: Change the invalid value to a valid value. See the table in the Details section.

Invalid URL

"ConfigurationData.url is '{0}'. This is not a valid URL" "DataBlobUri is '{0}'. This is not a valid URL"

"Configuration.url is '{0}'. This is not a valid URL"

Problem: A provided URL is not valid.

Solution: Check all your provided URLs. Make sure all URLs resolve to valid locations that the extension can access on the remote machine.

Invalid ConfigurationArgument Type

"Invalid configurationArguments type {0}"

Problem: The ConfigurationArguments property cannot resolve to a Hashtable object.

Solution: Make your ConfigurationArguments property a Hashtable. Follow the format provided in the preceding example. Watch out for quotes, commas, and braces.

Duplicate ConfigurationArguments

"Found duplicate arguments '{0}' in both public and protected configurationArguments"

Problem: The ConfigurationArguments in public settings and the ConfigurationArguments in protected settings contain properties with the same name.

Solution: Remove one of the duplicate properties.

Missing Properties

"Configuration.function requires that configuration.url or configuration.module is specified"

"Configuration.url requires that configuration.script is specified"

"Configuration.script requires that configuration.url is specified"

"Configuration.url requires that configuration.function is specified"

"ConfigurationUrlSasToken requires that configuration.url is specified"

"ConfigurationDataUrlSasToken requires that configurationData.url is specified"

Problem: A defined property needs another property that is missing.

Solutions:

- Provide the missing property.
- Remove the property that needs the missing property.

Next Steps

Learn about DSC and virtual machine scale sets in [Using Virtual Machine Scale Sets with the Azure DSC Extension](#)

Find more details on [DSC's secure credential management](#).

For more information on the Azure DSC extension handler, see [Introduction to the Azure Desired State Configuration extension handler](#).

For more information about PowerShell DSC, [visit the PowerShell documentation center](#).

AzureLogCollector Extension

6/27/2017 • 10 min to read • [Edit Online](#)

Diagnosing issues with an Microsoft Azure cloud service requires collecting the service's log files on virtual machines as the issues occur. You can use the AzureLogCollector extension on-demand to perform one-time collection of logs from one or more Cloud Service VMs (from both web roles and worker roles) and transfer the collected files to an Azure storage account – all without remotely logging on to any of the VMs.

NOTE

Descriptions for most of the logged information can be found at
<http://blogs.msdn.com/b/kwill/archive/2013/08/09/windows-azure-paas-compute-diagnostics-data.asp>.

There are two modes of collection dependent on the types of files to be collected.

- Azure Guest Agent Logs only (GA). This collection mode includes all the logs related to Azure guest agents and other Azure components.
- All Logs (Full). This collection mode will collect all files in GA mode plus:
 - system and application event logs
 - HTTP error logs
 - IIS Logs
 - Setup logs
 - other system logs

In both collection modes, additional data collection folders can be specified by using a collection of the following structure:

- **Name:** The name of the collection, which will be used as the name of subfolder inside the zip file to be collected.
- **Location:** The path to the folder on the virtual machine where file will be collected.
- **SearchPattern:** The pattern of the names of files to be collected. Default is “*”
- **Recursive:** if the files will be collected recursively under the folder.

Prerequisites

- You need to have a storage account for extension to save generated zip files.
- You must make sure that you are using Azure PowerShell Cmdlets V0.8.0 or above. For more information, see [Azure Downloads](#).

Add the extension

You can use [Microsoft Azure PowerShell](#) cmdlets or [Service Management REST APIs](#) to add the AzureLogCollector extension.

For Cloud Services, the existing Azure Powershell cmdlet, **Set-AzureServiceExtension**, can be used to enable the extension on Cloud Service role instances. Every time this extension is enabled through this cmdlet, log collection is triggered on the selected role instances of selected roles.

For Virtual Machines, the existing Azure Powershell cmdlet, **Set-AzureVMExtension**, can be used to enable the extension on Virtual Machines. Every time this extension is enabled through the cmdlets, log collection is triggered

on each instance.

Internally, this extension uses the JSON-based PublicConfiguration and PrivateConfiguration. The following is the layout of a sample JSON for public and private configuration.

PublicConfiguration

```
{  
    "Instances": "*",
    "Mode": "Full",
    "SasUri": "SasUri to your storage account with sp=wl",
    "AdditionalData": [
        {
            "Name": "StorageData",
            "Location": "%roleroot%storage",
            "SearchPattern": "*.*",
            "Recursive": "true"
        },
        {
            "Name": "CustomDataFolder2",
            "Location": "c:\customFolder",
            "SearchPattern": "*.log",
            "Recursive": "false"
        }
    ]
}
```

PrivateConfiguration

```
{  
}
```

NOTE

This extension doesn't need **privateConfiguration**. You can just provide an empty structure for the **-PrivateConfiguration** argument.

You can follow one of the two following steps to add the AzureLogCollector to one or more instances of a Cloud Service or Virtual Machine of selected roles, which triggers the collections on each VM to run and send the collected files to Azure account specified.

Adding as a Service Extension

1. Follow the instructions to connect Azure PowerShell to your subscription.
2. Specify the service name, slot, roles, and role instances to which you want to add and enable the AzureLogCollector extension.

```

#Specify your cloud service name
$ServiceName = 'extensiontest2'

#Specify the slot. 'Production' or 'Staging'
$slot = 'Production'

#Specified the roles on which the extension will be installed and enabled
$roles = @("WorkerRole1","WebRole1")

#Specify the instances on which extension will be installed and enabled. Use wildcard * for all
instances
$instance = @("*")

#Specify the collection mode, "Full" or "GA"
$mode = "GA"

```

3. Specify the additional data folder for which files will be collected (this step is optional).

```

#add one location
$a1 = New-Object PSObject

$a1 | Add-Member -MemberType NoteProperty -Name "Name" -Value "StorageData"
$a1 | Add-Member -MemberType NoteProperty -Name "SearchPattern" -Value "*"
$a1 | Add-Member -MemberType NoteProperty -Name "Location" -Value "%roleroot%storage"  #%%roleroot% is
normally E: or F: drive
$a1 | Add-Member -MemberType NoteProperty -Name "Recursive" -Value "true"

$AdditionalDataList+= $a1
#more locations can be added....

```

NOTE

You can use token `%roleroot%` to specify the role root drive since it doesn't use a fixed drive.

4. Provide the Azure storage account name and key to which collected files will be uploaded.

```

$StorageAccountName = 'YourStorageAccountName'
$StorageAccountKey = 'YourStorageAccountKey'

```

5. Call the SetAzureServiceLogCollector.ps1 (included at the end of the article) as follows to enable the AzureLogCollector extension for a Cloud Service. Once the execution is completed, you can find the uploaded file under <https://YourStorageAccountName.blob.core.windows.net/vmlogs>

```

.\SetAzureServiceLogCollector.ps1 -ServiceName YourCloudServiceName -Roles $roles -Instances
$instance -Mode $mode -StorageAccountName $StorageAccountName -StorageAccountKey $StorageAccountKey -
AdditionDataLocationList $AdditionalDataList

```

The following is the definition of the parameters passed to the script. (This is copied below as well.)

```
[CmdletBinding(SupportsShouldProcess = $true)]  
  
param (  
    [Parameter(Mandatory=$true)]  
    [string] $ServiceName,  
  
    [Parameter(Mandatory=$false)]  
    [string[]] $Roles ,  
  
    [Parameter(Mandatory=$false)]  
    [string[]] $Instances,  
  
    [Parameter(Mandatory=$false)]  
    [string] $Slot = 'Production',  
  
    [Parameter(Mandatory=$false)]  
    [string] $Mode = 'Full',  
  
    [Parameter(Mandatory=$false)]  
    [string] $StorageAccountName,  
  
    [Parameter(Mandatory=$false)]  
    [string] $StorageAccountKey,  
  
    [Parameter(Mandatory=$false)]  
    [PSObject[]] $AdditionalDataLocationList = $null  
)
```

- *ServiceName*: Your cloud service name.
- *Roles*: A list of roles, such as "WebRole1" or "WorkerRole1".
- *Instances*: A list of the names of role instances separated by comma -- use the wildcard string ("*") for all role instances.
- *Slot*: Slot name. "Production" or "Staging".
- *Mode*: Collection mode. "Full" or "GA".
- *StorageAccountName*: Name of Azure storage account for storing collected data.
- *StorageAccountKey*: Name of Azure storage account key.
- *AdditionalDataLocationList*: A list of the following structure:

```
{  
    String Name,  
    String Location,  
    String SearchPattern,  
    Bool   Recursive  
}
```

Adding as a VM Extension

Follow the instructions to connect Azure PowerShell to your subscription.

1. Specify the service name, VM, and the collection mode.

```

#Specify your cloud service name
$ServiceName = 'YourCloudServiceName'

#Specify the VM name
$VMName = "'YourVMName"

#Specify the collection mode, "Full" or "GA"
$mode = "GA"

Specify the additional data folder for which files will be collected (this step is optional).

#add one location
$a1 = New-Object PSObject

$a1 | Add-Member -MemberType NoteProperty -Name "Name" -Value "StorageData"
$a1 | Add-Member -MemberType NoteProperty -Name "SearchPattern" -Value "*"
$a1 | Add-Member -MemberType NoteProperty -Name "Location" -Value "%roleroot%storage" #%roleroot% is
normally E: or F: drive
$a1 | Add-Member -MemberType NoteProperty -Name "Recursive" -Value "true"

$AdditionalDataList+= $a1
#more locations can be added....

```

2. Provide the Azure storage account name and key to which collected files will be uploaded.

```

$StorageAccountName = 'YourStorageAccountName'
$StorageAccountKey = 'YourStorageAccountKey'

```

3. Call the SetAzureVMLogCollector.ps1 (included at the end of the article) as follows to enable the AzureLogCollector extension for a Cloud Service. Once the execution is completed, you can find the uploaded file under <https://YourStorageAccountName.blob.core.windows.net/vmlogs>

The following is the definition of the parameters passed to the script. (This is copied below as well.)

```

[CmdletBinding(SupportsShouldProcess = $true)]

param (
    [Parameter(Mandatory=$true)]
    [string] $ServiceName,

    [Parameter(Mandatory=$false)]
    [string] $VMName ,

    [Parameter(Mandatory=$false)]
    [string] $Mode = 'Full',

    [Parameter(Mandatory=$false)]
    [string] $StorageAccountName,

    [Parameter(Mandatory=$false)]
    [string] $StorageAccountKey,

    [Parameter(Mandatory=$false)]
    [PSObject[]] $AdditionalDataLocationList = $null
)

```

- ServiceName: Your cloud service name.
- VMName The name of the VM.
- Mode: Collection mode. "Full" or "GA".
- StorageAccountName: Name of Azure storage account for storing collected data.

- StorageAccountKey: Name of Azure storage account key.
- AdditionalDataLocationList: A list of the following structure:

```
{
    String Name,
    String Location,
    String SearchPattern,
    Bool   Recursive
}
```

Extention PowerShell Script files

SetAzureServiceLogCollector.ps1

```
[CmdletBinding(SupportsShouldProcess = $true)]

param (
    [Parameter(Mandatory=$true)]
    [string]    $ServiceName,

    [Parameter(Mandatory=$false)]
    [string[]]  $Roles ,

    [Parameter(Mandatory=$false)]
    [string[]]  $Instances = '*',

    [Parameter(Mandatory=$false)]
    [string]    $Slot = 'Production',

    [Parameter(Mandatory=$false)]
    [string]    $Mode = 'Full',

    [Parameter(Mandatory=$false)]
    [string]    $StorageAccountName,

    [Parameter(Mandatory=$false)]
    [string]    $StorageAccountKey,

    [Parameter(Mandatory=$false)]
    [PSObject[]] $AdditionDataLocationList = $null
)

$publicConfig = New-Object PSObject

if ($Instances -ne $null -and $Instances.Count -gt 0) #Instances should be seperated by ,
{
    $instanceText = $Instances[0]
    for ($i = 1;$i -lt $Instances.Count;$i++)
    {
        $instanceText = $instanceText+ "," + $Instances[$i]
    }
    $publicConfig | Add-Member -MemberType NoteProperty -Name "Instances" -Value $instanceText
}
else #For all instances if not specified. The value should be a space or *
{
    $publicConfig | Add-Member -MemberType NoteProperty -Name "Instances" -Value " "
}

if ($Mode -ne $null )
{
    $publicConfig | Add-Member -MemberType NoteProperty -Name "Mode" -Value $Mode
}
else
{
```

```

$publicConfig | Add-Member -MemberType NoteProperty -Name "Mode" -Value "Full"
}

#
#we need to get the Sasuri from StorageAccount and containers
#
$context = New-AzureStorageContext -Protocol https -StorageAccountName $StorageAccountName -StorageAccountKey
$StorageAccountKey

$ContainerName = "azurelogcollectordata"
$existingContainer = Get-AzureStorageContainer -Context $context | Where-Object { $_.Name -like
$ContainerName}
if ($existingContainer -eq $null)
{
    "Container ($ContainerName) doesn't exist. Creating it now.."
    New-AzureStorageContainer -Context $context -Name $ContainerName -Permission off
}

$ExpiryTime = [DateTime]::Now.AddMinutes(120).ToString("o")
$SasUri = New-AzureStorageContainerSASToken -ExpiryTime $ExpiryTime -FullUri -Name $ContainerName -Permission
rwl -Context $context
$publicConfig | Add-Member -MemberType NoteProperty -Name "SasUri" -Value $SasUri

#
#Add AdditionalData to collect data from additional folders
#
if ($AdditionDataLocationList -ne $null )
{
    $publicConfig | Add-Member -MemberType NoteProperty -Name "AdditionalData" -Value $AdditionDataLocationList
}

#
# Convert it to JSON format
#
$publicConfigJSON = $publicConfig | ConvertTo-Json
"publicConfig is: $publicConfigJSON"

#we just provide a empty privateConfig object
$privateconfig = "{

}"
if ($Roles -ne $null)
{
    Set-AzureServiceExtension -Service $ServiceName -Slot $Slot -Role $Roles -ExtensionName
'AzureLogCollector' -ProviderNamespace Microsoft.WindowsAzure.Compute -PublicConfiguration $publicConfigJSON -
PrivateConfiguration $privateconfig -Version 1.0 -Verbose
}
else
{
    Set-AzureServiceExtension -Service $ServiceName -Slot $Slot -ExtensionName 'AzureLogCollector' -
ProviderNamespace Microsoft.WindowsAzure.Compute -PublicConfiguration $publicConfigJSON -PrivateConfiguration
$privateconfig -Version 1.0 -Verbose
}

#
#This is an optional step: generate a sasUri to the container so it can be shared with other people if nened
#
$SasExpireTime = [DateTime]::Now.AddMinutes(120).ToString("o")
$SasUri = New-AzureStorageContainerSASToken -ExpiryTime $ExpiryTime -FullUri -Name $ContainerName -Permission
rwl -Context $context
$SasUri = $SasUri + "&restype=container&comp=list"
Write-Output "The container for uploaded file can be accessed using this link:`r`n$sasuri"

```

SetAzureVMLogCollector.ps1

```
[CmdletBinding(SupportsShouldProcess = $true)]
```

```

param (
    [Parameter(Mandatory=$true)]
    [string] $ServiceName,
    [Parameter(Mandatory=$false)]
    [string] $VMName ,
    [Parameter(Mandatory=$false)]
    [string] $Mode = 'Full',
    [Parameter(Mandatory=$false)]
    [string] $StorageAccountName,
    [Parameter(Mandatory=$false)]
    [string] $StorageAccountKey,
    [Parameter(Mandatory=$false)]
    [PSObject[]] $AdditionDataLocationList = $null
)

$publicConfig = New-Object PSObject
$publicConfig | Add-Member -MemberType NoteProperty -Name "Instances" -Value "*"

if ($Mode -ne $null )
{
    $publicConfig | Add-Member -MemberType NoteProperty -Name "Mode" -Value $Mode
}
else
{
    $publicConfig | Add-Member -MemberType NoteProperty -Name "Mode" -Value "Full"
}

#
#we need to get the Sasuri from StorageAccount and containers
#


```

```

#
$privateconfig = "{

if ($VMName -ne $null )
{
    $VM = Get-AzureVM -ServiceName $ServiceName -Name $VMName
    $VM.VM.OSVirtualHardDisk.OS

    if ($VM.VM.OSVirtualHardDisk.OS -like '*Windows*')
    {
        Set-AzureVMExtension -VM $VM -ExtensionName "AzureLogCollector" -Publisher
Microsoft.WindowsAzure.Compute -PublicConfiguration $publicConfigJSON -PrivateConfiguration $privateconfig -
Version 1.* | Update-AzureVM -Verbose

        #
        #We will check the VM status to find if operation by extension has been completed or not. The
completion of the operation,either succeed or fail, can be indicated by
        #the presence of SubstatusList field.
        #
        $Completed = $false
        while ($Completed -ne $true)
        {
            $VM = Get-AzureVM -ServiceName $ServiceName -Name $VMName
            $status = $VM.ResourceExtensionStatusList | Where-Object {$_.HandlerName -eq
"Microsoft.WindowsAzure.Compute.AzureLogCollector"}

            if ( ($status.Code -ne 0) -and ($status.Status -like '*error*'))
            {
                Write-Output "Error status is returned:
 $($status.ExtensionSettingStatus.FormattedMessage.Message)."
                $Completed = $true
            }
            elseif (( $status.ExtensionSettingStatus.SubstatusList -eq $null -or
$status.ExtensionSettingStatus.SubstatusList.Count -lt 1))
            {
                $Completed = $false
                Write-Output "Waiting for operation to complete..."
            }
            else
            {
                $Completed = $true
                Write-Output "Operation completed.

                $UploadedFileUri = $status.ExtensionSettingStatus.SubstatusList[0].FormattedMessage.Message
                $blob = New-Object
Microsoft.WindowsAzure.Storage.Blob.CloudBlockBlob($UploadedFileUri)

                #
                # This is an optional step: For easier access to the file, we can generate a read-only
SasUri directly to the file
                #
                $ExpiryTimeRead = [DateTime]::Now.AddMinutes(120).ToString("o")
                $ReadSasUri = New-AzureStorageBlobSASToken -ExpiryTime $ExpiryTimeRead -FullUri -
Blob $blob.name -Container $blob.Container.Name -Permission r -Context $context

                Write-Output "The uploaded file can be accessed using this link: $ReadSasUri"

                #
                #This is an optional step: Remove the extension after we are done
                #
                Get-AzureVM -ServiceName $ServiceName -Name $VMName | Set-AzureVMExtension -Publisher
Microsoft.WindowsAzure.Compute -ExtensionName "AzureLogCollector" -Version 1.* -Uninstall | Update-AzureVM -
Verbose

            }
            Start-Sleep -s 5
        }
    }
}

```

```
    else
    {
        Write-Output "VM OS Type is not Windows, the extension cannot be enabled"
    }

}
else
{
    Write-Output "VM name is not specified, the extension cannot be enabled"
}
```

Next Steps

Now you can examine or copy your logs from one very simple location.

Use PowerShell to enable Azure Diagnostics in a virtual machine running Windows

6/28/2017 • 6 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

Azure Diagnostics is the capability within Azure that enables the collection of diagnostic data on a deployed application. You can use the diagnostics extension to collect diagnostic data like application logs or performance counters from an Azure virtual machine (VM) that is running Windows. This article describes how to use Windows PowerShell to enable the diagnostics extension for a VM. See [How to install and configure Azure PowerShell](#) for the prerequisites needed for this article.

Enable the diagnostics extension if you use the Resource Manager deployment model

You can enable the diagnostics extension while you create a Windows VM through the Azure Resource Manager deployment model by adding the extension configuration to the Resource Manager template. See [Create a Windows virtual machine with monitoring and diagnostics by using the Azure Resource Manager template](#).

To enable the diagnostics extension on an existing VM that was created through the Resource Manager deployment model, you can use the [Set-AzureRMVMDiagnosticExtension](#) PowerShell cmdlet as shown below.

```
$vm_resourcegroup = "myvmresourcegroup"
$vm_name = "myvm"
$diagnosticsconfig_path = "DiagnosticsPubConfig.xml"

Set-AzureRmVMDiagnosticExtension -ResourceGroupName $vm_resourcegroup -VMName $vm_name -
DiagnosticsConfigurationPath $diagnosticsconfig_path
```

\$diagnosticsconfig_path is the path to the file that contains the diagnostics configuration in XML, as described in the [sample](#) below.

If the diagnostics configuration file specifies a **StorageAccount** element with a storage account name, then the *Set-AzureRMVMDiagnosticExtension* script will automatically set the diagnostics extension to send diagnostic data to that storage account. For this to work, the storage account needs to be in the same subscription as the VM.

If no **StorageAccount** was specified in the diagnostics configuration, then you need to pass in the *StorageAccountName* parameter to the cmdlet. If the *StorageAccountName* parameter is specified, then the cmdlet will always use the storage account that is specified in the parameter and not the one that is specified in the diagnostics configuration file.

If the diagnostics storage account is in a different subscription from the VM, then you need to explicitly pass in the *StorageAccountName* and *StorageAccountKey* parameters to the cmdlet. The *StorageAccountKey* parameter is not needed when the diagnostics storage account is in the same subscription, as the cmdlet can automatically query and set the key value when enabling the diagnostics extension. However, if the diagnostics storage account is in a different subscription, then the cmdlet might not be able to get the key automatically and you need to explicitly specify the key through the *StorageAccountKey* parameter.

```
Set-AzureRmVMDiagnosticExtension -ResourceGroupName $vm_resourcegroup -VMName $vm_name -  
DiagnosticsConfigurationPath $diagnosticsconfig_path -StorageAccountName $diagnosticssstorage_name -  
StorageAccountKey $diagnosticssstorage_key
```

Once the diagnostics extension is enabled on a VM, you can get the current settings by using the [Get-AzureRMVmDiagnosticExtension](#) cmdlet.

```
Get-AzureRmVMDiagnosticExtension -ResourceGroupName $vm_resourcegroup -VMName $vm_name
```

The cmdlet returns *PublicSettings*, which contains the diagnostics configuration. There are two kinds of configuration supported, *WadCfg* and *xmlCfg*. *WadCfg* is JSON configuration, and *xmlCfg* is XML configuration in a Base64-encoded format. To read the XML, you need to decode it.

```
$publicsettings = (Get-AzureRmVMDiagnosticExtension -ResourceGroupName $vm_resourcegroup -VMName  
$vm_name).PublicSettings  
$encodedconfig = (ConvertFrom-Json -InputObject $publicsettings).xmlCfg  
$xmlconfig = [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($encodedconfig))  
Write-Host $xmlconfig
```

The [Remove-AzureRMVmDiagnosticExtension](#) cmdlet can be used to remove the diagnostics extension from the VM.

Enable the diagnostics extension if you use the classic deployment model

You can use the [Set-AzureVMDiagnosticExtension](#) cmdlet to enable a diagnostics extension on a VM that you create through the classic deployment model. The following example shows how to create a new VM through the classic deployment model with the diagnostics extension enabled.

```
$VM = New-AzureVMConfig -Name $VM -InstanceSize Small -ImageName $VMImage  
$VM = Add-AzureProvisioningConfig -VM $VM -AdminUsername $Username -Password $Password -Windows  
$VM = Set-AzureVMDiagnosticExtension -DiagnosticsConfigurationPath $Config_Path -VM $VM -StorageContext  
$Storage_Context  
New-AzureVM -Location $Location -ServiceName $Service_Name -VM $VM
```

To enable the diagnostics extension on an existing VM that was created through the classic deployment model, first use the [Get-AzureVM](#) cmdlet to get the VM configuration. Then update the VM configuration to include the diagnostics extension by using the [Set-AzureVMDiagnosticExtension](#) cmdlet. Finally, apply the updated configuration to the VM by using [Update-AzureVM](#).

```
$VM = Get-AzureVM -ServiceName $Service_Name -Name $VM_Name  
$VM_Update = Set-AzureVMDiagnosticExtension -DiagnosticsConfigurationPath $Config_Path -VM $VM -  
StorageContext $Storage_Context  
Update-AzureVM -ServiceName $Service_Name -Name $VM_Name -VM $VM_Update.VM
```

Sample diagnostics configuration

The following XML can be used for the diagnostics public configuration with the above scripts. This sample configuration will transfer various performance counters to the diagnostics storage account, along with errors from the application, security, and system channels in the Windows event logs and any errors from the diagnostics infrastructure logs.

The configuration needs to be updated to include the following:

- The `resourceID` attribute of the **Metrics** element needs to be updated with the resource ID for the VM.
 - The resource ID can be constructed by using the following pattern: `"/subscriptions/{subscription ID for the subscription with the VM}/resourceGroups/{The resourcegroup name for the VM}/providers/Microsoft.Compute/virtualMachines/{The VM Name}"`.
 - For example, if the subscription ID for the subscription where the VM is running is **11111111-1111-1111-1111-111111111111**, the resource group name for the resource group is **MyResourceGroup**, and the VM Name is **MyWindowsVM**, then the value for `resourceID` would be:

```
<Metrics resourceId="/subscriptions/11111111-1111-1111-1111-  
111111111111/resourceGroups/MyResourceGroup/providers/Microsoft.Compute/virtualMachines/MyWindows  
VM" >
```

- For more information on how metrics are generated based on the performance counters and metrics configuration, see [Azure Diagnostics metrics table in storage](#).
- The **StorageAccount** element needs to be updated with the name of the diagnostics storage account.

```
<?xml version="1.0" encoding="utf-8"?>  
<PublicConfig xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">  
    <WadCfg>  
        <DiagnosticMonitorConfiguration overallQuotaInMB="4096">  
            <DiagnosticInfrastructureLogs scheduledTransferLogLevelFilter="Error"/>  
            <PerformanceCounters scheduledTransferPeriod="PT1M">  
                <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time"  
                    sampleRate="PT15S" unit="Percent">  
                    <annotation displayName="CPU utilization" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Privileged Time"  
                    sampleRate="PT15S" unit="Percent">  
                    <annotation displayName="CPU privileged time" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% User Time"  
                    sampleRate="PT15S" unit="Percent">  
                    <annotation displayName="CPU user time" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Processor Information(_Total)\Processor  
Frequency" sampleRate="PT15S" unit="Count">  
                    <annotation displayName="CPU frequency" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\System\Processes" sampleRate="PT15S"  
                    unit="Count">  
                    <annotation displayName="Processes" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Process(_Total)\Thread Count"  
                    sampleRate="PT15S" unit="Count">  
                    <annotation displayName="Threads" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Process(_Total)\Handle Count"  
                    sampleRate="PT15S" unit="Count">  
                    <annotation displayName="Handles" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Memory\% Committed Bytes In Use"  
                    sampleRate="PT15S" unit="Percent">  
                    <annotation displayName="Memory usage" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Memory\Available Bytes" sampleRate="PT15S"  
                    unit="Bytes">  
                    <annotation displayName="Memory available" locale="en-us"/>  
                </PerformanceCounterConfiguration>  
                <PerformanceCounterConfiguration counterSpecifier="\Memory\Committed Bytes" sampleRate="PT15S"  
                    unit="Bytes">
```

```

        <annotation displayName="Memory committed" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\Memory\Commit Limit" sampleRate="PT15S"
unit="Bytes">
        <annotation displayName="Memory commit limit" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\Memory\Pool Paged Bytes" sampleRate="PT15S"
unit="Bytes">
        <annotation displayName="Memory paged pool" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\Memory\Pool Nonpaged Bytes"
sampleRate="PT15S" unit="Bytes">
        <annotation displayName="Memory non-paged pool" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\% Disk Time"
sampleRate="PT15S" unit="Percent">
        <annotation displayName="Disk active time" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\% Disk Read Time"
sampleRate="PT15S" unit="Percent">
        <annotation displayName="Disk active read time" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\% Disk Write Time"
sampleRate="PT15S" unit="Percent">
        <annotation displayName="Disk active write time" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Disk Transfers/sec"
sampleRate="PT15S" unit="CountPerSecond">
        <annotation displayName="Disk operations" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Disk Reads/sec"
sampleRate="PT15S" unit="CountPerSecond">
        <annotation displayName="Disk read operations" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Disk Writes/sec"
sampleRate="PT15S" unit="CountPerSecond">
        <annotation displayName="Disk write operations" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Disk Bytes/sec"
sampleRate="PT15S" unit="BytesPerSecond">
        <annotation displayName="Disk speed" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Disk Read Bytes/sec"
sampleRate="PT15S" unit="BytesPerSecond">
        <annotation displayName="Disk read speed" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Disk Write Bytes/sec"
sampleRate="PT15S" unit="BytesPerSecond">
        <annotation displayName="Disk write speed" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Avg. Disk Queue Length"
sampleRate="PT15S" unit="Count">
        <annotation displayName="Disk average queue length" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Avg. Disk Read Queue
Length" sampleRate="PT15S" unit="Count">
        <annotation displayName="Disk average read queue length" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\PhysicalDisk(_Total)\Avg. Disk Write Queue
Length" sampleRate="PT15S" unit="Count">
        <annotation displayName="Disk average write queue length" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\LogicalDisk(_Total)\% Free Space"
sampleRate="PT15S" unit="Percent">
        <annotation displayName="Disk free space (percentage)" locale="en-us"/>
    </PerformanceCounterConfiguration>
    <PerformanceCounterConfiguration counterSpecifier="\LogicalDisk(_Total)\Free Megabytes"
sampleRate="PT15S" unit="Count">
        <annotation displayName="Disk free space (MB)" locale="en-us"/>
    </PerformanceCounterConfiguration>

```

```
</PerformanceCounterConfiguration>
</PerformanceCounters>
<Metrics resourceId="(Update with resource ID for the VM)" >
    <MetricAggregation scheduledTransferPeriod="PT1H"/>
    <MetricAggregation scheduledTransferPeriod="PT1M"/>
</Metrics>
<WindowsEventLog scheduledTransferPeriod="PT1M">
    <DataSource name="Application!*[System[(Level = 1 or Level = 2)]]"/>
    <DataSource name="Security!*[System[(Level = 1 or Level = 2)]]"/>
    <DataSource name="System!*[System[(Level = 1 or Level = 2)]]"/>
</WindowsEventLog>
</DiagnosticMonitorConfiguration>
</WadCfg>
<StorageAccount>(Update with diagnostics storage account name)</StorageAccount>
</PublicConfig>
```

Next steps

- For additional guidance on using the Azure Diagnostics capability and other techniques to troubleshoot problems, see [Enabling Diagnostics in Azure Cloud Services and Virtual Machines](#).
- [Diagnostics configurations schema](#) explains the various XML configurations options for the diagnostics extension.

Exporting Resource Groups that contain VM extensions

11/8/2017 • 3 min to read • [Edit Online](#)

Azure Resource Groups can be exported into a new Resource Manager template that can then be redeployed. The export process interprets existing resources, and creates a Resource Manager template that when deployed results in a similar Resource Group. When using the Resource Group export option against a Resource Group containing Virtual Machine extensions, several items need to be considered such as extension compatibility and protected settings.

This document details how the Resource Group export process works regarding virtual machine extensions, including a list of supported extensions, and details on handling secured data.

Supported Virtual Machine Extensions

Many Virtual Machine extensions are available. Not all extensions can be exported into a Resource Manager template using the "Automation Script" feature. If a virtual machine extension is not supported, it needs to be manually placed back into the exported template.

The following extensions can be exported with the automation script feature.

EXTENSION			
Acronis Backup	Datadog Windows Agent	OS Patching For Linux	VM Snapshot Linux
Acronis Backup Linux	Docker Extension	Puppet Agent	
Bg Info	DSC Extension	Site 24x7 Apm Insight	
BMC CTM Agent Linux	Dynatrace Linux	Site 24x7 Linux Server	
BMC CTM Agent Windows	Dynatrace Windows	Site 24x7 Windows Server	
Chef Client	HPE Security Application Defender	Trend Micro DSA	
Custom Script	IaaS Antimalware	Trend Micro DSA Linux	
Custom Script Extension	IaaS Diagnostics	VM Access For Linux	
Custom Script for Linux	Linux Chef Client	VM Access For Linux	
Datadog Linux Agent	Linux Diagnostic	VM Snapshot	

Export the Resource Group

To export a Resource Group into a reusable template, complete the following steps:

1. Sign in to the Azure portal

2. On the Hub Menu, click Resource Groups
3. Select the target resource group from the list
4. In the Resource Group blade, click Automation Script

```

1 {
2   "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
3   "contentVersion": "1.0.0.0",
4   "parameters": {
5     "virtualMachines_MyWindowsVM_adminPassword": {
6       "defaultValue": null,
7       "type": "SecureString"
8     },
9     "extensions_Microsoft.Insights.VERBOSEiagnosticsSettings_protectedSettings": {
10       "defaultValue": null,
11       "type": "SecureObject"
12     },
13     "virtualMachines_MyWindowsVM_name": {
14       "defaultValue": "MyWindowsVM",
15       "type": "String"
16     },
17     "networkInterfaces_myVMNic_name": {
18       "defaultValue": "myVMNic",
19       "type": "String"
20     },
21     "publicIPAddresses_myPublicIP_name": {
22       "defaultValue": "myPublicIP",
23       "type": "String"
24     },
25     "virtualNetworks_MyVNET_name": {
26       "defaultValue": "MyVNET",
27     }
28   }
29 }

```

The Azure Resource Manager automations script produces a Resource Manager template, a parameters file, and several sample deployment scripts such as PowerShell and Azure CLI. At this point, the exported template can be downloaded using the download button, added as a new template to the template library, or redeployed using the deploy button.

Configure protected settings

Many Azure virtual machine extensions include a protected settings configuration, that encrypts sensitive data such as credentials and configuration strings. Protected settings are not exported with the automation script. If necessary, protected settings need to be reinserted into the exported templated.

Step 1 - Remove template parameter

When the Resource Group is exported, a single template parameter is created to provide a value to the exported protected settings. This parameter can be removed. To remove the parameter, look through the parameter list and delete the parameter that looks similar to this JSON example.

```

"extensions_extensionname_protectedSettings": {
  "defaultValue": null,
  "type": "SecureObject"
}

```

Step 2 - Get protected settings properties

Because each protected setting has a set of required properties, a list of these properties need to be gathered. Each parameter of the protected settings configuration can be found in the [Azure Resource Manager schema on GitHub](#). This schema only includes the parameter sets for the extensions listed in the overview section of this document.

From within the schema repository, search for the desired extension, for this example `IaaS Diagnostics`. Once the extensions `protectedSettings` object has been located, take note of each parameter. In the example of the `IaaS Diagnostic` extension, the required parameters are `storageAccountName`, `storageAccountKey`, and `storageAccountEndPoint`.

```
"protectedSettings": {
    "type": "object",
    "properties": {
        "storageAccountName": {
            "type": "string"
        },
        "storageAccountKey": {
            "type": "string"
        },
        "storageAccountEndPoint": {
            "type": "string"
        }
    },
    "required": [
        "storageAccountName",
        "storageAccountKey",
        "storageAccountEndPoint"
    ]
}
```

Step 3 - Re-create the protected configuration

On the exported template, search for `protectedSettings` and replace the exported protected setting object with a new one that includes the required extension parameters and a value for each one.

In the example of the `IaaSDiagnostic` extension, the new protected setting configuration would look like the following example:

```
"protectedSettings": {
    "storageAccountName": "[parameters('storageAccountName')]",
    "storageAccountKey": "[parameters('storageAccountKey')]",
    "storageAccountEndPoint": "https://core.windows.net"
}
```

The final extension resource looks similar to the following JSON example:

```
{
  "name": "Microsoft.Insights.VMDiagnosticsSettings",
  "type": "extensions",
  "location": "[resourceGroup().location]",
  "apiVersion": "[variables('apiVersion')]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
  ],
  "tags": {
    "displayName": "AzureDiagnostics"
  },
  "properties": {
    "publisher": "Microsoft.Azure.Diagnostics",
    "type": "IaaS.Diagnostics",
    "typeHandlerVersion": "1.5",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "xmlCfg": "[base64(concat(variables('wadcfgxstart'), variables('wadmetricsresourceid'),
variables('vmName'), variables('wadcfgxend')))]",
      "storageAccount": "[parameters('existingdiagnosticsStorageAccountName')]"
    },
    "protectedSettings": {
      "storageAccountName": "[parameters('storageAccountName')]",
      "storageAccountKey": "[parameters('storageAccountKey')]",
      "storageAccountEndPoint": "https://core.windows.net"
    }
  }
}
```

If using template parameters to provide property values, these need to be created. When creating template parameters for protected setting values, make sure to use the `SecureString` parameter type so that sensitive values are secured. For more information on using parameters, see [Authoring Azure Resource Manager templates](#).

In the example of the `IaaSDiagnostic` extension, the following parameters would be created in the parameters section of the Resource Manager template.

```
"storageAccountName": {
  "defaultValue": null,
  "type": "SecureString"
},
"storageAccountKey": {
  "defaultValue": null,
  "type": "SecureString"
}
```

At this point, the template can be deployed using any template deployment method.

Troubleshooting Azure Windows VM extension failures

6/27/2017 • 1 min to read • [Edit Online](#)

Overview of Azure Resource Manager templates

Azure Resource Manager templates allows you to declaratively specify the Azure IaaS infrastructure in JSON language by defining the dependencies between resources.

See [Authoring extension templates](#) to learn more about authoring templates for using extensions.

In this article we'll learn about troubleshooting some of the common VM extension failures.

Viewing extension status

Azure Resource Manager templates can be executed from Azure PowerShell. Once the template is executed, the extension status can be viewed from Azure Resource Explorer or the command line tools.

Here is an example:

Azure PowerShell:

```
Get-AzureRmVM -ResourceGroupName $RGName -Name $vmName -Status
```

Here is the sample output:

```
Extensions: {
  "ExtensionType": "Microsoft.Compute.CustomScriptExtension",
  "Name": "myCustomScriptExtension",
  "SubStatuses": [
    {
      "Code": "ComponentStatus/StdOut/succeeded",
      "DisplayStatus": "Provisioning succeeded",
      "Level": "Info",
      "Message": "Directory: C:\\\\temp\\\\n\\\\n\\\\nMode
      \\n----",
      "Time": null
    },
    {
      "Code": "ComponentStatus/StdErr/succeeded",
      "DisplayStatus": "Provisioning succeeded",
      "Level": "Info",
      "Message": "",
      "Time": null
    }
  ]
}
```

Troubleshooting extension failures

Re-running the extension on the VM

If you are running scripts on the VM using Custom Script Extension, you could sometimes run into an error where VM was created successfully but the script has failed. Under these conditions, the recommended way to recover from this error is to remove the extension and rerun the template again. Note: In future, this functionality would be enhanced to remove the need for uninstalling the extension.

Remove the extension from Azure PowerShell

```
Remove-AzureRmVMExtension -ResourceGroupName $RGName -VMName $vmName -Name "myCustomScriptExtension"
```

Once the extension has been removed, the template can be re-executed to run the scripts on the VM.

Use monitoring and diagnostics with a Windows VM and Azure Resource Manager templates

8/10/2017 • 8 min to read • [Edit Online](#)

The Azure Diagnostics Extension provides the monitoring and diagnostics capabilities on a Windows based Azure virtual machine. You can enable these capabilities on the virtual machine by including the extension as part of the azure resource manager template. See [Authoring Azure Resource Manager Templates with VM Extensions](#) for more information on including any extension as part of a virtual machine template. This article describes how you can add the Azure Diagnostics extension to a windows virtual machine template.

Add the Azure Diagnostics extension to the VM resource definition

To enable the diagnostics extension on a Windows Virtual Machine you need to add the extension as a VM resource in the Resource manager template.

For a simple Resource Manager based Virtual Machine add the extension configuration to the *resources* array for the Virtual Machine:

```
"resources": [
    {
        "name": "Microsoft.Insights.VMDiagnosticsSettings",
        "type": "extensions",
        "location": "[resourceGroup().location]",
        "apiVersion": "2015-06-15",
        "dependsOn": [
            "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
        ],
        "tags": {
            "displayName": "AzureDiagnostics"
        },
        "properties": {
            "publisher": "Microsoft.Azure.Diagnostics",
            "type": "IaaSDiagnostics",
            "typeHandlerVersion": "1.5",
            "autoUpgradeMinorVersion": true,
            "settings": {
                "xmlCfg": "[base64(concat(variables('wadcfgxstart'),
variables('wadmetricsresourceid'), variables('vmName'), variables('wadcfgxend')))]",
                "storageAccount": "[parameters('existingdiagnosticsStorageAccountName')]"
            },
            "protectedSettings": {
                "storageAccountName": "[parameters('existingdiagnosticsStorageAccountName')]",
                "storageAccountKey": "[listkeys(variables('accountid'), '2015-05-01-preview').key1]",
                "storageAccountEndPoint": "https://core.windows.net"
            }
        }
    }
]
```

Another common convention is add the extension configuration at the root resources node of the template instead of defining it under the virtual machine's resources node. With this approach you have to explicitly specify a hierarchical relation between the extension and the virtual machine with the *name* and *type* values. For example:

```
"name": "[concat(variables('vmName'), 'Microsoft.Insights.VMDiagnosticsSettings')]",  
"type": "Microsoft.Compute/virtualMachines/extensions",
```

The extension is always associated with the virtual machine, you can either directly define it under the virtual machine's resource node directly or define it at the base level and use the hierarchical naming convention to associate it with the virtual machine.

For Virtual Machine Scale Sets the extensions configuration is specified in the *extensionProfile* property of the *VirtualMachineProfile*.

The *publisher* property with the value of **Microsoft.Azure.Diagnostics** and the *type* property with the value of **IaaSDiagnostics** uniquely identify the Azure Diagnostics extension.

The value of the *name* property can be used to refer to the extension in the resource group. Setting it specifically to **Microsoft.Insights.VMDiagnosticsSettings** will enable it to be easily identified by the Azure portal ensuring that the monitoring charts show up correctly in the Azure portal.

The *typeHandlerVersion* specifies the version of the extension you would like to use. Setting *autoUpgradeMinorVersion* minor version to **true** ensures that you will get the latest Minor version of the extension that is available. It is highly recommended that you always set *autoUpgradeMinorVersion* to always be **true** so that you always get to use the latest available diagnostics extension with all the new features and bug fixes.

The *settings* element contains configurations properties for the extension that can be set and read back from the extension (sometimes referred to as public configuration). The *xmlcfg* property contains xml based configuration for the diagnostics logs, performance counters etc that will be collected by the diagnostics agent. See [Diagnostics Configuration Schema](#) for more information about the xml schema itself. A common practice is to store the actual xml configuration as a variable in the Azure Resource Manager template and then concatenate and base64 encode them to set the value for *xmlcfg*. See the section on [diagnostics configuration variables](#) to understand more about how to store the xml in variables. The *storageAccount* property specifies the name of the storage account to which diagnostics data will be transferred.

The properties in *protectedSettings* (sometimes referred to as private configuration) can be set but cannot be read back after being set. The write-only nature of *protectedSettings* makes it useful for storing secrets like the storage account key where the diagnostics data will be written.

Specifying diagnostics storage account as parameters

The diagnostics extension json snippet above assumes two parameters *existingdiagnosticsStorageAccountName* and *existingdiagnosticsStorageResourceGroup* to specify the diagnostics storage account where diagnostics data will be stored. Specifying the diagnostics storage account as a parameter makes it easy to change the diagnostics storage account across different environments e.g. you may want to use a different diagnostics storage account for testing and a different one for your production deployment.

```
"existingdiagnosticsStorageAccountName": {  
    "type": "string",  
    "metadata": {  
        "description": "The name of an existing storage account to which diagnostics data will be transferred."  
    },  
    "existingdiagnosticsStorageResourceGroup": {  
        "type": "string",  
        "metadata": {  
            "description": "The resource group for the storage account specified in  
existingdiagnosticsStorageAccountName"  
        },  
    }  
}
```

It is best practice to specify a diagnostics storage account in a different resource group than the resource group for the virtual machine. A resource group can be considered to be a deployment unit with its own lifetime, a virtual machine can be deployed and redeployed as new configurations updates are made to it but you may want to continue storing the diagnostics data in the same storage account across those virtual machine deployments. Having the storage account in a different resource enables the storage account to accept data from various virtual machine deployments making it easy to troubleshoot issues across the various versions.

NOTE

If you create a windows virtual machine template from Visual Studio the default storage account might be set to use the same storage account where the virtual machine VHD is uploaded. This is to simplify initial setup of the VM. You should re-factor the template to use a different storage account that can be passed in as a parameter.

Diagnostics configuration variables

The diagnostics extension json snippet above defines an *accountid* variable to simplify getting the storage account key for the diagnostics storage:

```
"accountid": "[concat('/subscriptions/', subscription().subscriptionId,  
'/resourceGroups/',parameters('existingdiagnosticsStorageResourceGroup'),  
'/providers/', 'Microsoft.Storage/storageAccounts/', parameters('existingdiagnosticsStorageAccountName'))]"
```

The *xmlcfg* property for the diagnostics extension is defined using multiple variables that are concatenated together. The values of these variables are in xml so they need to be escaped correctly when setting the json variables.

The following describes the diagnostics configuration xml that collects standard system level performance counters along with some windows event logs and diagnostics infrastructure logs. It has been escaped and formatted correctly so that the configuration can directly be pasted into the variables section of your template. See the [Diagnostics Configuration Schema](#) for a more human readable example of the configuration xml.

```

"wadlogs": "<WadCfg> <DiagnosticMonitorConfiguration overallQuotaInMB=\"4096\" xmlns=\"http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration\">
<DiagnosticInfrastructureLogs scheduledTransferLogLevelFilter=\"Error\"/> <WindowsEventLog
scheduledTransferPeriod=\"PT1M\"> <DataSource name=\"Application!*[System[(Level = 1 or Level = 2)]]\" />
<DataSource name=\"Security!*[System[(Level = 1 or Level = 2)]]\" /> <DataSource name=\"System!*[System[(Level
= 1 or Level = 2)]]\" /></WindowsEventLog>",

"wadperfcounters1": "<PerformanceCounters scheduledTransferPeriod=\"PT1M\">
<PerformanceCounterConfiguration counterSpecifier=\"\\Processor(_Total)\\% Processor Time\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU utilization\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Processor(_Total)\\% Privileged Time\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU privileged time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Processor(_Total)\\% User Time\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"CPU user time\" locale=\"en-us\"/></PerformanceCounterConfiguration>
<PerformanceCounterConfiguration counterSpecifier=\"\\Processor Information(_Total)\\Processor Frequency\" sampleRate=\"PT15S\" unit=\"Count\"><annotation displayName=\"CPU frequency\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\System\\Processes\\\" sampleRate=\"PT15S\" unit=\"Count\"><annotation displayName=\"Processes\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Process(_Total)\\Thread Count\" sampleRate=\"PT15S\" unit=\"Count\"><annotation displayName=\"Threads\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Process(_Total)\\Handle Count\" sampleRate=\"PT15S\" unit=\"Count\"><annotation displayName=\"Handles\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\% Committed Bytes In Use\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Memory usage\" locale=\"en-us\"/></PerformanceCounterConfiguration>
<PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\Available Bytes\" sampleRate=\"PT15S\" unit=\"Bytes\"><annotation displayName=\"Memory available\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\Committed Bytes\" sampleRate=\"PT15S\" unit=\"Bytes\"><annotation displayName=\"Memory committed\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\Memory\\Commit Limit\" sampleRate=\"PT15S\" unit=\"Bytes\"><annotation displayName=\"Memory commit limit\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\% Disk Time\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Disk active time\" locale=\"en-us\"/></PerformanceCounterConfiguration>",

"wadperfcounters2": "<PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\% Disk Read Time\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Disk active read time\" locale=\"en-us\"/></PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\% Disk Write Time\" sampleRate=\"PT15S\" unit=\"Percent\">
<annotation displayName=\"Disk active write time\" locale=\"en-us\"/></PerformanceCounterConfiguration>
<PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\Disk Transfers/sec\" sampleRate=\"PT15S\" unit=\"CountPerSecond\"><annotation displayName=\"Disk operations\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\Disk Reads/sec\" sampleRate=\"PT15S\" unit=\"CountPerSecond\">
<annotation displayName=\"Disk read operations\" locale=\"en-us\"/></PerformanceCounterConfiguration>
<PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\Disk Writes/sec\" sampleRate=\"PT15S\" unit=\"CountPerSecond\"><annotation displayName=\"Disk write operations\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\Disk Bytes/sec\" sampleRate=\"PT15S\" unit=\"BytesPerSecond\">
<annotation displayName=\"Disk speed\" locale=\"en-us\"/></PerformanceCounterConfiguration>
<PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\Disk Read Bytes/sec\" sampleRate=\"PT15S\" unit=\"BytesPerSecond\"><annotation displayName=\"Disk read speed\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\PhysicalDisk(_Total)\\Disk Write Bytes/sec\" sampleRate=\"PT15S\" unit=\"BytesPerSecond\"><annotation displayName=\"Disk write speed\" locale=\"en-us\"/>
</PerformanceCounterConfiguration><PerformanceCounterConfiguration counterSpecifier=\"\\LogicalDisk(_Total)\\% Free Space\" sampleRate=\"PT15S\" unit=\"Percent\"><annotation displayName=\"Disk free space (percentage)\" locale=\"en-us\"/></PerformanceCounterConfiguration></PerformanceCounters>",

"wadcfgxstart": "[concat(variables('wadlogs'), variables('wadperfcounters1'), variables('wadperfcounters2'), '<Metrics resourceId=\"\">')]",

"wadmetricsresourceid": "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/', 'Microsoft.Compute/virtualMachines/')]",
"wadcfgxend": "\"><MetricAggregation scheduledTransferPeriod=\"PT1H\"/><MetricAggregation scheduledTransferPeriod=\"PT1M\"/></Metrics></DiagnosticMonitorConfiguration></WadCfg>"

```

The Metrics definition xml node in the above configuration is an important configuration element as it defines how the performance counters defined earlier in the xml in *PerformanceCounter* node will be aggregated and stored.

IMPORTANT

These metrics drive the monitoring charts and alerts in the Azure portal. The **Metrics** node with the *resourceID* and **MetricAggregation** must be included in the diagnostics configuration for your VM if you want to see the VM monitoring data in the Azure portal.

The following is an example of the xml for metrics definitions:

```
<Metrics  
resourceId="/subscriptions/subscription().subscriptionId/resourceGroups/resourceGroup().name/providers/Microsoft.Compute/virtualMachines/vmName">  
    <MetricAggregation scheduledTransferPeriod="PT1H"/>  
    <MetricAggregation scheduledTransferPeriod="PT1M"/>  
</Metrics>
```

The *resourceID* attribute uniquely identifies the virtual machine in your subscription. Make sure to use the `subscription()` and `resourceGroup()` functions so that the template automatically updates those values based on the subscription and resource group you are deploying to.

If you are creating multiple Virtual Machines in a loop then you will have to populate the *resourceID* value with an `copyIndex()` function to correctly differentiate each individual VM. The *xmlCfg* value can be updated to support this as follows:

```
"xmlCfg": "[base64(concat(variables('wadcfgxstart'), variables('wadmetricsresourceid'),  
concat(parameters('vmNamePrefix'), copyindex()), variables('wadcfgxend')))]",
```

The MetricAggregation value of *PT1H* and *PT1M* signify an aggregation over a minute and an aggregation over an hour.

WADMetrics tables in storage

The Metrics configuration above will generate tables in your diagnostics storage account with the following naming conventions:

- **WADMetrics** : Standard prefix for all WADMetrics tables
- **PT1H** or **PT1M** : Signifies that the table contains aggregate data over 1 hour or 1 minute
- **P10D** : Signifies the table will contain data for 10 days from when the table started collecting data
- **V2S** : String constant
- **yyyymmdd** : The date at which the table started collecting data

Example: *WADMetricsPT1HP10DV2S20151108* will contain metrics data aggregated over an hour for 10 days starting on 11-Nov-2015

Each WADMetrics table will contain the following columns:

- **PartitionKey**: The partitionkey is constructed based on the *resourceID* value to uniquely identify the VM resource. for e.g. :
`002Fsubscriptions::002FresourceGroups:002F:002Fproviders:002FMicrosoft:002ECompute:002FvirtualMachines:002F`
- **RowKey** : Follows the format .. The descending time tick calculation is max time ticks minus the time of the beginning of the aggregation period. E.g. if the sample period started on 10-Nov-2015 and 00:00Hrs UTC then the calculation would be: `DateTime.MaxValue.Ticks - (new DateTime(2015,11,10,0,0,0,DateTimeKind.Utc).Ticks)`.
For the memory available bytes performance counter the row key will look like:
`251955187199999999__:005CMemory:005CAvailable:0020Bytes`
- **CounterName** : Is the name of the performance counter. This matches the *counterSpecifier* defined in the xml

config.

- **Maximum** : The maximum value of the performance counter over the aggregation period.
- **Minimum** : The minimum value of the performance counter over the aggregation period.
- **Total** : The sum of all values of the performance counter reported over the aggregation period.
- **Count** : The total number of values reported for the performance counter.
- **Average** : The average (total/count) value of the performance counter over the aggregation period.

Next Steps

- For a complete sample template of a Windows virtual machine with diagnostics extension see [201-vm-monitoring-diagnostics-extension](#)
- Deploy the resource manager template using [Azure PowerShell](#) or [Azure Command Line](#)
- Learn more about [authoring Azure Resource Manager templates](#)

Network Watcher Agent virtual machine extension for Windows

12/20/2017 • 2 min to read • [Edit Online](#)

Overview

[Azure Network Watcher](#) is a network performance monitoring, diagnostic, and analytics service that allows monitoring of Azure networks. The Network Watcher Agent virtual machine extension is a requirement for capturing network traffic on demand, and other advanced functionality on Azure virtual machines.

This document details the supported platforms and deployment options for the Network Watcher Agent virtual machine extension for Windows.

Prerequisites

Operating system

The Network Watcher Agent extension for Windows can be run against Windows Server 2008 R2, 2012, 2012 R2, and 2016 releases. Nano Server is not supported.

Internet connectivity

Some of the Network Watcher Agent functionality requires that the target virtual machine be connected to the Internet. Without the ability to establish outgoing connections, the Network Watcher Agent will not be able to upload packet captures to your storage account. For more details, please see the [Network Watcher documentation](#).

Extension schema

The following JSON shows the schema for the Network Watcher Agent extension. The extension neither requires, nor supports, any user-supplied settings, and relies on its default configuration.

```
{  
    "type": "extensions",  
    "name": "Microsoft.Azure.NetworkWatcher",  
    "apiVersion": "[variables('apiVersion')]",  
    "location": "[resourceGroup().location]",  
    "dependsOn": [  
        "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"  
    ],  
    "properties": {  
        "publisher": "Microsoft.Azure.NetworkWatcher",  
        "type": "NetworkWatcherAgentWindows",  
        "typeHandlerVersion": "1.4",  
        "autoUpgradeMinorVersion": true  
    }  
}
```

Property values

NAME	VALUE / EXAMPLE
apiVersion	2015-06-15

NAME	VALUE / EXAMPLE
publisher	Microsoft.Azure.NetworkWatcher
type	NetworkWatcherAgentWindows
typeHandlerVersion	1.4

Template deployment

You can deploy Azure VM extensions with Azure Resource Manager templates. You can use the JSON schema detailed in the previous section in an Azure Resource Manager template to run the Network Watcher Agent extension during an Azure Resource Manager template deployment.

PowerShell deployment

Use the `Set-AzureRmVMExtension` command to deploy the Network Watcher Agent virtual machine extension to an existing virtual machine:

```
Set-AzureRmVMExtension ` 
-ResourceGroupName "myResourceGroup1" ` 
-Location "WestUS" ` 
-VMName "myVM1" ` 
-Name "networkWatcherAgent" ` 
-Publisher "Microsoft.Azure.NetworkWatcher" ` 
-Type "NetworkWatcherAgentWindows" ` 
-TypeHandlerVersion "1.4"
```

Troubleshooting and support

Troubleshooting

You can retrieve data about the state of extension deployments from the Azure portal and PowerShell. To see the deployment state of extensions for a given VM, run the following command using the Azure PowerShell module:

```
Get-AzureRmVMExtension -ResourceGroupName myResourceGroup1 -VMName myVM1 -Name networkWatcherAgent
```

Extension execution output is logged to files found in the following directory:

```
C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.NetworkWatcher.NetworkWatcherAgentWindows\
```

Support

If you need more help at any point in this article, you can refer to the Network Watcher User Guide documentation or contact the Azure experts on the [MSDN Azure and Stack Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select Get support. For information about using Azure Support, read the [Microsoft Azure support FAQ](#).

Migrate from Amazon Web Services (AWS) and other platforms to Managed Disks in Azure

12/7/2017 • 4 min to read • [Edit Online](#)

You can upload VHD files from AWS or on-premises virtualization solutions to Azure to create VMs that take advantage of Managed Disks. Azure Managed Disks removes the need to manage storage accounts for Azure IaaS VMs. You have to only specify the type (Premium or Standard) and size of disk you need, and Azure creates and manages the disk for you.

You can upload either generalized and specialized VHDs.

- **Generalized VHD** - has had all of your personal account information removed using Sysprep.
- **Specialized VHD** - maintains the user accounts, applications, and other state data from your original VM.

IMPORTANT

Before uploading any VHD to Azure, you should follow [Prepare a Windows VHD or VHDX to upload to Azure](#)

SCENARIO	DOCUMENTATION
You have existing AWS EC2 instances that you would like to migrate to Azure VMs using managed disks	Move a VM from Amazon Web Services (AWS) to Azure
You have a VM from another virtualization platform that you would like to use as an image to create multiple Azure VMs.	Upload a generalized VHD and use it to create a new VM in Azure
You have a uniquely customized VM that you would like to recreate in Azure.	Upload a specialized VHD to Azure and create a new VM

Overview of Managed Disks

Azure Managed Disks simplifies VM management by removing the need to manage storage accounts. Managed Disks also benefit from better reliability of VMs in an Availability Set. It ensures that the disks of different VMs in an Availability Set are sufficiently isolated from each other to avoid a single point of failure. It automatically places disks of different VMs in an Availability Set in different Storage scale units (stamps) which limits the impact of single Storage scale unit failures caused due to hardware and software failures. Based on your needs, you can choose from two types of storage options:

- [Premium Managed Disks](#) are Solid State Drive (SSD) based storage media, which delivers high performance, low-latency disk support for virtual machines running I/O-intensive workloads. You can take advantage of the speed and performance of these disks by migrating to Premium Managed Disks.
- [Standard Managed Disks](#) use Hard Disk Drive (HDD) based storage media and are best suited for Dev/Test and other infrequent access workloads that are less sensitive to performance variability.

Plan for the migration to Managed Disks

This section helps you to make the best decision on VM and disk types.

Location

Pick a location where Azure Managed Disks are available. If you are migrating to Premium Managed Disks, also ensure that Premium storage is available in the region where you are planning to migrate to. See [Azure Services by Region](#) for up-to-date information on available locations.

VM sizes

If you are migrating to Premium Managed Disks, you have to update the size of the VM to Premium Storage capable size available in the region where VM is located. Review the VM sizes that are Premium Storage capable. The Azure VM size specifications are listed in [Sizes for virtual machines](#). Review the performance characteristics of virtual machines that work with Premium Storage and choose the most appropriate VM size that best suits your workload. Make sure that there is sufficient bandwidth available on your VM to drive the disk traffic.

Disk sizes

Premium Managed Disks

There are three types of Premium Managed disks that can be used with your VM and each has specific IOPs and throughput limits. Consider these limits when choosing the Premium disk type for your VM based on the needs of your application in terms of capacity, performance, scalability, and peak loads.

PREMIUM DISKS TYPE	P10	P20	P30
Disk size	128 GB	512 GB	1024 GB (1 TB)
IOPS per disk	500	2300	5000
Throughput per disk	100 MB per second	150 MB per second	200 MB per second

Standard Managed Disks

There are five types of Standard Managed disks that can be used with your VM. Each of them have different capacity but have same IOPS and throughput limits. Choose the type of Standard Managed disks based on the capacity needs of your application.

STANDARD DISK TYPE	S4	S6	S10	S20	S30
Disk size	30 GB	64 GB	128 GB	512 GB	1024 GB (1 TB)
IOPS per disk	500	500	500	500	500
Throughput per disk	60 MB per second				

Disk caching policy

Premium Managed Disks

By default, disk caching policy is *Read-Only* for all the Premium data disks, and *Read-Write* for the Premium operating system disk attached to the VM. This configuration setting is recommended to achieve the optimal performance for your application's IOs. For write-heavy or write-only data disks (such as SQL Server log files), disable disk caching so that you can achieve better application performance.

Pricing

Review the [pricing for Managed Disks](#). Pricing of Premium Managed Disks is same as the Premium Unmanaged Disks. But pricing for Standard Managed Disks is different than Standard Unmanaged Disks.

Next Steps

- Before uploading any VHD to Azure, you should follow [Prepare a Windows VHD or VHDX to upload to Azure](#)

Upload a generalized VHD and use it to create new VMs in Azure

8/21/2017 • 7 min to read • [Edit Online](#)

This topic walks you through using PowerShell to upload a VHD of a generalized VM to Azure, create an image from the VHD and create a new VM from that image. You can upload a VHD exported from an on-premises virtualization tool or from another cloud. Using [Managed Disks](#) for the new VM simplifies the VM management and provides better availability when the VM is placed in an availability set.

If you want to use a sample script, see [Sample script to upload a VHD to Azure and create a new VM](#)

Before you begin

- Before uploading any VHD to Azure, you should follow [Prepare a Windows VHD or VHDX to upload to Azure](#)
- Review [Plan for the migration to Managed Disks](#) before starting your migration to [Managed Disks](#).
- Make sure that you have the latest version of the AzureRM.Compute PowerShell module. Run the following command to install it.

```
Install-Module AzureRM.Compute -RequiredVersion 2.6.0
```

For more information, see [Azure PowerShell Versioning](#).

Generalize the Windows VM using Sysprep

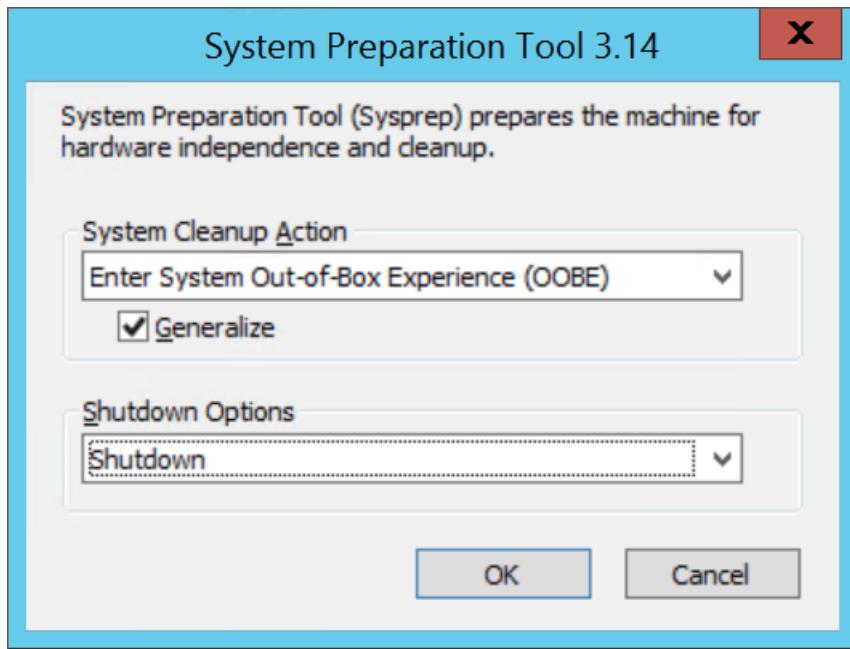
Sysprep removes all your personal account information, among other things, and prepares the machine to be used as an image. For details about Sysprep, see [How to Use Sysprep: An Introduction](#).

Make sure the server roles running on the machine are supported by Sysprep. For more information, see [Sysprep Support for Server Roles](#)

IMPORTANT

If you are running Sysprep before uploading your VHD to Azure for the first time, make sure you have [prepared your VM](#) before running Sysprep.

1. Sign in to the Windows virtual machine.
2. Open the Command Prompt window as an administrator. Change the directory to `%windir%\system32\sysprep`, and then run `sysprep.exe`.
3. In the **System Preparation Tool** dialog box, select **Enter System Out-of-Box Experience (OOBE)**, and make sure that the **Generalize** check box is selected.
4. In **Shutdown Options**, select **Shutdown**.
5. Click **OK**.



- When Sysprep completes, it shuts down the virtual machine. Do not restart the VM.

Log in to Azure

If you don't already have PowerShell version 1.4 or above installed, read [How to install and configure Azure PowerShell](#).

- Open Azure PowerShell and sign in to your Azure account. A pop-up window opens for you to enter your Azure account credentials.

```
Login-AzureRmAccount
```

- Get the subscription IDs for your available subscriptions.

```
Get-AzureRmSubscription
```

- Set the correct subscription using the subscription ID. Replace with the ID of the correct subscription.

```
Select-AzureRmSubscription -SubscriptionId "<subscriptionID>"
```

Get the storage account

You need a storage account in Azure to store the uploaded VM image. You can either use an existing storage account or create a new one.

If you will be using the VHD to create a managed disk for a VM, the storage account location must be same the location where you will be creating the VM.

To show the available storage accounts, type:

```
Get-AzureRmStorageAccount
```

If you want to use an existing storage account, proceed to the [Upload the VM image](#) section.

If you need to create a storage account, follow these steps:

1. You need the name of the resource group where the storage account should be created. To find out all the resource groups that are in your subscription, type:

```
Get-AzureRmResourceGroup
```

To create a resource group named **myResourceGroup** in the **East US** region, type:

```
New-AzureRmResourceGroup -Name myResourceGroup -Location "East US"
```

2. Create a storage account named **mystorageaccount** in this resource group by using the [New-AzureRmStorageAccount](#) cmdlet:

```
New-AzureRmStorageAccount -ResourceGroupName myResourceGroup -Name mystorageaccount -Location "East US"  
-SkuName "Standard_LRS" -Kind "Storage"
```

Valid values for -SkuName are:

- **Standard_LRS** - Locally redundant storage.
- **Standard_ZRS** - Zone redundant storage.
- **Standard_GRS** - Geo redundant storage.
- **Standard_RAGRS** - Read access geo redundant storage.
- **Premium_LRS** - Premium locally redundant storage.

Upload the VHD to your storage account

Use the [Add-AzureRmVhd](#) cmdlet to upload the VHD to a container in your storage account. This example uploads the file *myVHD.vhd* from "C:\Users\Public\Documents\Virtual hard disks" to a storage account named *mystorageaccount* in the *myResourceGroup* resource group. The file will be placed into the container named *mycontainer* and the new file name will be *myUploadedVHD.vhd*.

```
$rgName = "myResourceGroup"  
$urlOfUploadedImageVhd = "https://mystorageaccount.blob.core.windows.net/mycontainer/myUploadedVHD.vhd"  
Add-AzureRmVhd -ResourceGroupName $rgName -Destination $urlOfUploadedImageVhd  
-LocalFilePath "C:\Users\Public\Documents\Virtual hard disks\myVHD.vhd"
```

If successful, you get a response that looks similar to this:

```
MD5 hash is being calculated for the file C:\Users\Public\Documents\Virtual hard disks\myVHD.vhd.  
MD5 hash calculation is completed.  
Elapsed time for the operation: 00:03:35  
Creating new page blob of size 53687091712...  
Elapsed time for upload: 01:12:49  
  
LocalFilePath      DestinationUri  
-----  
C:\Users\Public\Doc... https://mystorageaccount.blob.core.windows.net/mycontainer/myUploadedVHD.vhd
```

Depending on your network connection and the size of your VHD file, this command may take a while to complete.

Save the **Destination Uri** path to use later if you are going to create a managed disk or a new VM using the uploaded VHD.

Other options for uploading a VHD

You can also upload a VHD to your storage account using one of the following:

- [AzCopy](#)
- [Azure Storage Copy Blob API](#)
- [Azure Storage Explorer Uploading Blobs](#)
- [Storage Import/Export Service REST API Reference](#)
- We recommend using Import/Export Service if estimated uploading time is longer than 7 days. You can use [DataTransferSpeedCalculator](#) to estimate the time from data size and transfer unit. Import/Export can be used to copy to a standard storage account. You will need to copy from standard storage to premium storage account using a tool like AzCopy.

Create a managed image from the uploaded VHD

Create a managed image using your generalized OS VHD. Replace the values with your own information.

1. First, set the common parameters:

```
$vmName = "myVM"  
$computerName = "myComputer"  
$vmSize = "Standard_DS1_v2"  
$location = "East US"  
$imageName = "yourImageName"
```

2. Create the image using your generalized OS VHD.

```
$imageConfig = New-AzureRmImageConfig -Location $location  
$imageConfig = Set-AzureRmImageOsDisk -Image $imageConfig -OsType Windows -OsState Generalized -BlobUri  
$urlOfUploadedImageVhd  
$image = New-AzureRmImage -ImageName $imageName -ResourceGroupName $rgName -Image $imageConfig
```

Create a virtual network

Create the vNet and subnet of the [virtual network](#).

1. Create the subnet. This example creates a subnet named *mySubnet* with the address prefix of *10.0.0.0/24*.

```
$subnetName = "mySubnet"  
$singleSubnet = New-AzureRmVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix 10.0.0.0/24
```

2. Create the virtual network. This example creates a virtual network named *myVnet* with the address prefix of *10.0.0.0/16*.

```
$vnetName = "myVnet"  
$vnet = New-AzureRmVirtualNetwork -Name $vnetName -ResourceGroupName $rgName -Location $location `  
-AddressPrefix 10.0.0.0/16 -Subnet $singleSubnet
```

Create a public IP address and network interface

To enable communication with the virtual machine in the virtual network, you need a [public IP address](#) and a network interface.

1. Create a public IP address. This example creates a public IP address named *myPip*.

```
$ipName = "myPip"
$ip = New-AzureRmPublicIpAddress -Name $ipName -ResourceGroupName $rgName -Location $location ` -AllocationMethod Dynamic
```

2. Create the NIC. This example creates a NIC named **myNic**.

```
$nicName = "myNic"
$nic = New-AzureRmNetworkInterface -Name $nicName -ResourceGroupName $rgName -Location $location ` -SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $ip.Id
```

Create the network security group and an RDP rule

To be able to log in to your VM using RDP, you need to have a network security rule (NSG) that allows RDP access on port 3389.

This example creates an NSG named *myNsg* that contains a rule called *myRdpRule* that allows RDP traffic over port 3389. For more information about NSGs, see [Opening ports to a VM in Azure using PowerShell](#).

```
$nsgName = "myNsg"
$ruleName = "myRdpRule"
$rdpRule = New-AzureRmNetworkSecurityRuleConfig -Name $ruleName -Description "Allow RDP" ` -Access Allow -Protocol Tcp -Direction Inbound -Priority 110 ` -SourceAddressPrefix Internet -SourcePortRange * ` -DestinationAddressPrefix * -DestinationPortRange 3389

$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $rgName -Location $location ` -Name $nsgName -SecurityRules $rdpRule
```

Create a variable for the virtual network

Create a variable for the completed virtual network.

```
$vnet = Get-AzureRmVirtualNetwork -ResourceGroupName $rgName -Name $vnetName
```

Get the credentials for the VM

The following cmdlet will open a window where you will enter a new user name and password to use as the local administrator account for remotely accessing the VM.

```
$cred = Get-Credential
```

Add the VM name and size to the VM configuration.

```
$vm = New-AzureRmVMConfig -VMName $vmName -VMSize $vmSize
```

Set the VM image as source image for the new VM

Set the source image using the ID of the managed VM image.

```
$vm = Set-AzureRmVMSourceImage -VM $vm -Id $image.Id
```

Set the OS configuration and add the NIC.

Enter the storage type (PremiumLRS or StandardLRS) and the size of the OS disk. This example sets the account type to *PremiumLRS*, the disk size to 128 GB and disk caching to *ReadWrite*.

```
$vm = Set-AzureRmVMOSDisk -VM $vm -DiskSizeInGB 128 `  
-CreateOption FromImage -Caching ReadWrite  
  
$vm = Set-AzureRmVMOperatingSystem -VM $vm -Windows -ComputerName $computerName `  
-Credential $cred -ProvisionVMAgent -EnableAutoUpdate  
  
$vm = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id
```

Create the VM

Create the new VM using the configuration stored in the **\$vm** variable.

```
New-AzureRmVM -VM $vm -ResourceGroupName $rgName -Location $location
```

Verify that the VM was created

When complete, you should see the newly created VM in the [Azure portal](#) under **Browse > Virtual machines**, or by using the following PowerShell commands:

```
$vmList = Get-AzureRmVM -ResourceGroupName $rgName  
$vmList.Name
```

Next steps

To sign in to your new virtual machine, browse to the VM in the [portal](#), click **Connect**, and open the Remote Desktop RDP file. Use the account credentials of your original virtual machine to sign in to your new virtual machine. For more information, see [How to connect and log on to an Azure virtual machine running Windows](#).

Move a Windows VM from Amazon Web Services (AWS) to Azure using PowerShell

7/25/2017 • 2 min to read • [Edit Online](#)

If you are evaluating Azure virtual machines for hosting your workloads, you can export an existing Amazon Web Services (AWS) EC2 Windows VM instance then upload the virtual hard disk (VHD) to Azure. Once the VHD is uploaded, you can create a new VM in Azure from the VHD.

This topic covers moving a single VM from AWS to Azure. If you want to move VMs from AWS to Azure at scale, see [Migrate virtual machines in Amazon Web Services \(AWS\) to Azure with Azure Site Recovery](#).

Prepare the VM

You can upload both generalized and specialized VHDs to Azure. Each type requires that you prepare the VM before exporting from AWS.

- **Generalized VHD** - a generalized VHD has had all of your personal account information removed using Sysprep. If you intend to use the VHD as an image to create new VMs from, you should:
 - [Prepare a Windows VM](#).
 - Generalize the virtual machine using Sysprep.
- **Specialized VHD** - a specialized VHD maintains the user accounts, applications and other state data from your original VM. If you intend to use the VHD as-is to create a new VM, ensure the following steps are completed.
 - [Prepare a Windows VHD to upload to Azure](#). **Do not** generalize the VM using Sysprep.
 - Remove any guest virtualization tools and agents that are installed on the VM (i.e. VMware tools).
 - Ensure the VM is configured to pull its IP address and DNS settings via DHCP. This ensures that the server obtains an IP address within the VNet when it starts up.

Export and download the VHD

Export the EC2 instance to a VHD in an Amazon S3 bucket. Follow the steps described in the Amazon documentation topic [Exporting an Instance as a VM Using VM Import/Export](#) and run the `create-instance-export-task` command to export the EC2 instance to a VHD file.

The exported VHD file is saved in the Amazon S3 bucket you specify. The basic syntax for exporting the VHD is below, just replace the placeholder text in with your information.

```
aws ec2 create-instance-export-task --instance-id <instanceID> --target-environment Microsoft \
--export-to-s3-task DiskImageFormat=VHD,ContainerFormat=ova,S3Bucket=<bucket>,S3Prefix=<prefix>
```

Once the VHD has been exported, follow the instructions in [How Do I Download an Object from an S3 Bucket?](#) to download the VHD file from the S3 bucket.

IMPORTANT

AWS charges data transfer fees for downloading the VHD. See [Amazon S3 Pricing](#) for more information.

Next steps

Now you can upload the VHD to Azure and create a new VM.

- If you ran Sysprep on your source to **generalize** it before exporting, see [Upload a generalized VHD and use it to create a new VMs in Azure](#)
- If you did not run Sysprep before exporting, the VHD is considered **specialized**, see [Upload a specialized VHD to Azure and create a new VM](#)

Migrate to Azure with Site Recovery

10/31/2017 • 2 min to read • [Edit Online](#)

Read this article to learn about using the [Azure Site Recovery](#) service to migrate on-premises virtual machines (VMs) and physical servers, to Azure VMs.

Before you start

Watch this video for a quick overview of the steps required to migrate to Azure.

What do we mean by migration?

You can deploy Site Recovery to replication on-premises VMs and physical servers, and to migrate them.

- When you replicate, you configure on-premises machines to replicate on a regular basis to Azure. Then when an outage occurs, you fail the machines over from the on-premises site to Azure, and access them from there. When the on-premises site is available again, you fail back from Azure.
- When you use Site Recovery for migration, you replicate on-premises machines to Azure. Then you fail them over from your on-premises site to Azure, and finish up the migration process. There's no failback involved.

What can Site Recovery migrate?

You can:

- **Migrate from on-premises:** Migrate on-premises Hyper-V VMs, VMware VMs, and physical servers to Azure. After the migration, workloads running on the on-premises machines will be running on Azure VMs.
- **Migrate within Azure:** Migrate Azure VMs between Azure regions.
- **Migrate AWS:** Migrate AWS Windows instances to Azure IaaS VMs.

Migrate from on-premises to Azure

To migrate on-premises VMware VMs, Hyper-V VMs, and physical servers, you follow almost the same steps as you would for full replication.

Migrate between Azure regions

To migrate Azure VMs between regions, you follow almost the same steps as you would for full migration.

1. You [enable replication](#)) for the machines you want to migrate.
2. You [run a quick test failover](#) to make sure everything's working.
3. Then, you [run an unplanned failover](#) with the **Complete Migration** option.
4. After you've completed the migration, you can [set up replication for disaster recovery](#), from the Azure region to which you migrated, to a secondary region.

Migrate AWS to Azure

You can migrate AWS instances to Azure VMs.

- In this scenario only migration is supported. In other words, you can replicate the AWS instances and fail them over to Azure, but you can't fail back.
- AWS instances are handled in the same way as physical servers for migration purposes. You set up a Recovery Services vault, deploy an on-premises configuration server to manage replication, add it to the vault, and specify replication settings.
- You enable replication for the machines you want to migrate, and run a quick test failover. Then you run an unplanned failover with the **Complete Migration** option.

Next steps

- [Migrate on-premises machines to Azure](#)
- [Migrate VMs from one Azure region to another](#)
- [Migrate AWS to Azure](#)

Platform-supported migration of IaaS resources from classic to Azure Resource Manager

10/11/2017 • 7 min to read • [Edit Online](#)

In this article, we describe how we're enabling migration of infrastructure as a service (IaaS) resources from the Classic to Resource Manager deployment models. You can read more about [Azure Resource Manager features and benefits](#). We detail how to connect resources from the two deployment models that coexist in your subscription by using virtual network site-to-site gateways.

Goal for migration

Resource Manager enables deploying complex applications through templates, configures virtual machines by using VM extensions, and incorporates access management and tagging. Azure Resource Manager includes scalable, parallel deployment for virtual machines into availability sets. The new deployment model also provides lifecycle management of compute, network, and storage independently. Finally, there's a focus on enabling security by default with the enforcement of virtual machines in a virtual network.

Almost all the features from the classic deployment model are supported for compute, network, and storage under Azure Resource Manager. To benefit from the new capabilities in Azure Resource Manager, you can migrate existing deployments from the Classic deployment model.

Supported resources for migration

These classic IaaS resources are supported during migration

- Virtual Machines
- Availability Sets
- Cloud Services
- Storage Accounts
- Virtual Networks
- VPN Gateways
- Express Route Gateways (*in the same subscription as Virtual Network only*)
- Network Security Groups
- Route Tables
- Reserved IPs

Supported scopes of migration

There are 4 different ways to complete migration of compute, network, and storage resources. These are

- Migration of virtual machines (NOT in a virtual network)
- Migration of virtual machines (in a virtual network)
- Storage accounts migration
- Unattached resources (Network Security Groups, Route Tables & Reserved IPs)

Migration of virtual machines (NOT in a virtual network)

In the Resource Manager deployment model, security is enforced for your applications by default. All VMs need to be in a virtual network in the Resource Manager model. The Azure platform restarts (`stop`, `Deallocate`, and

Start) the VMs as part of the migration. You have two options for the virtual networks that the Virtual Machines will be migrated to:

- You can request the platform to create a new virtual network and migrate the virtual machine into the new virtual network.
- You can migrate the virtual machine into an existing virtual network in Resource Manager.

NOTE

In this migration scope, both the management-plane operations and the data-plane operations may not be allowed for a period of time during the migration.

Migration of virtual machines (in a virtual network)

For most VM configurations, only the metadata is migrating between the Classic and Resource Manager deployment models. The underlying VMs are running on the same hardware, in the same network, and with the same storage. The management-plane operations may not be allowed for a certain period of time during the migration. However, the data plane continues to work. That is, your applications running on top of VMs (classic) do not incur downtime during the migration.

The following configurations are not currently supported. If support is added in the future, some VMs in this configuration might incur downtime (go through stop, deallocate, and restart VM operations).

- You have more than one availability set in a single cloud service.
- You have one or more availability sets and VMs that are not in an availability set in a single cloud service.

NOTE

In this migration scope, the management plane may not be allowed for a period of time during the migration. For certain configurations as described earlier, data-plane downtime occurs.

Storage accounts migration

To allow seamless migration, you can deploy Resource Manager VMs in a classic storage account. With this capability, compute and network resources can and should be migrated independently of storage accounts. Once you migrate over your Virtual Machines and Virtual Network, you need to migrate over your storage accounts to complete the migration process.

NOTE

The Resource Manager deployment model doesn't have the concept of Classic images and disks. When the storage account is migrated, Classic images and disks are not visible in the Resource Manager stack but the backing VHDs remain in the storage account.

Unattached resources (Network Security Groups, Route Tables & Reserved IPs)

Network Security Groups, Route Tables & Reserved IPs that are not attached to any Virtual Machines and Virtual Networks can be migrated independently.

Unsupported features and configurations

We do not currently support some features and configurations. The following sections describe our recommendations around them.

Unsupported features

The following features are not currently supported. You can optionally remove these settings, migrate the VMs, and then re-enable the settings in the Resource Manager deployment model.

RESOURCE PROVIDER	FEATURE	RECOMMENDATION
Compute	Unassociated virtual machine disks.	The VHD blobs behind these disks will get migrated when the Storage Account is migrated
Compute	Virtual machine images.	The VHD blobs behind these disks will get migrated when the Storage Account is migrated
Network	Endpoint ACLs.	Remove Endpoint ACLs and retry migration.
Network	Application Gateway	Remove the Application Gateway before beginning migration and then recreate the Application Gateway once migration is complete.
Network	Virtual networks using VNet Peering.	Migrate Virtual Network to Resource Manager, then peer. Learn more about VNet Peering .

Unsupported configurations

The following configurations are not currently supported.

SERVICE	CONFIGURATION	RECOMMENDATION
Resource Manager	Role Based Access Control (RBAC) for classic resources	Because the URI of the resources is modified after migration, it is recommended that you plan the RBAC policy updates that need to happen after migration.
Compute	Multiple subnets associated with a VM	Update the subnet configuration to reference only subnets.
Compute	Virtual machines that belong to a virtual network but don't have an explicit subnet assigned	You can optionally delete the VM.
Compute	Virtual machines that have alerts, Autoscale policies	The migration goes through and these settings are dropped. It is highly recommended that you evaluate your environment before you do the migration. Alternatively, you can reconfigure the alert settings after migration is complete.

Service	Configuration	Recommendation
Compute	XML VM extensions (BGInfo 1.* , Visual Studio Debugger, Web Deploy, and Remote Debugging)	This is not supported. It is recommended that you remove these extensions from the virtual machine to continue migration or they will be dropped automatically during the migration process.
Compute	Boot diagnostics with Premium storage	Disable Boot Diagnostics feature for the VMs before continuing with migration. You can re-enable boot diagnostics in the Resource Manager stack after the migration is complete. Additionally, blobs that are being used for screenshot and serial logs should be deleted so you are no longer charged for those blobs.
Compute	Cloud services that contain web/worker roles	This is currently not supported.
Compute	Cloud services that contain more than one availability set or multiple availability sets.	This is currently not supported. Please move the Virtual Machines to the same availability set before migrating.
Compute	VM with Azure Security Center extension	Azure Security Center automatically installs extensions on your Virtual Machines to monitor their security and raise alerts. These extensions usually get installed automatically if the Azure Security Center policy is enabled on the subscription. To migrate the Virtual Machines, please disable the security center policy on the subscription which will remove the Security Center monitoring extension from the Virtual Machines.
Compute	VM with backup or snapshot extension	These extensions are installed on a Virtual Machine configured with the Azure Backup service. To migrate these Virtual Machines, follow the guidance here .
Network	Virtual networks that contain virtual machines and web/worker roles	This is currently not supported. Please move the Web/Worker roles to their own Virtual Network before migrating. Once the classic Virtual Network is migrated, the migrated Azure Resource Manager Virtual Network can be peered with the classic Virtual Network to achieve similar configuration as before.

Service	Configuration	Recommendation
Network	Classic Express Route circuits	This is currently not supported. These circuits need to be migrated to Azure Resource Manager before beginning IaaS migration. To learn more about this see Moving ExpressRoute circuits from the classic to the Resource Manager deployment model .
Azure App Service	Virtual networks that contain App Service environments	This is currently not supported.
Azure HDInsight	Virtual networks that contain HDInsight services	This is currently not supported.
Microsoft Dynamics Lifecycle Services	Virtual networks that contain virtual machines that are managed by Dynamics Lifecycle Services	This is currently not supported.
Azure AD Domain Services	Virtual networks that contain Azure AD Domain services	This is currently not supported.
Azure RemoteApp	Virtual networks that contain Azure RemoteApp deployments	This is currently not supported.
Azure API Management	Virtual networks that contain Azure API Management deployments	This is currently not supported. To migrate the IaaS VNET, please change the VNET of the API Management deployment which is a no downtime operation.

Next steps

- [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#)
- [Planning for migration of IaaS resources from classic to Azure Resource Manager](#)
- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager](#)
- [Review most common migration errors](#)
- [Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager](#)

Technical deep dive on platform-supported migration from classic to Azure Resource Manager

6/27/2017 • 13 min to read • [Edit Online](#)

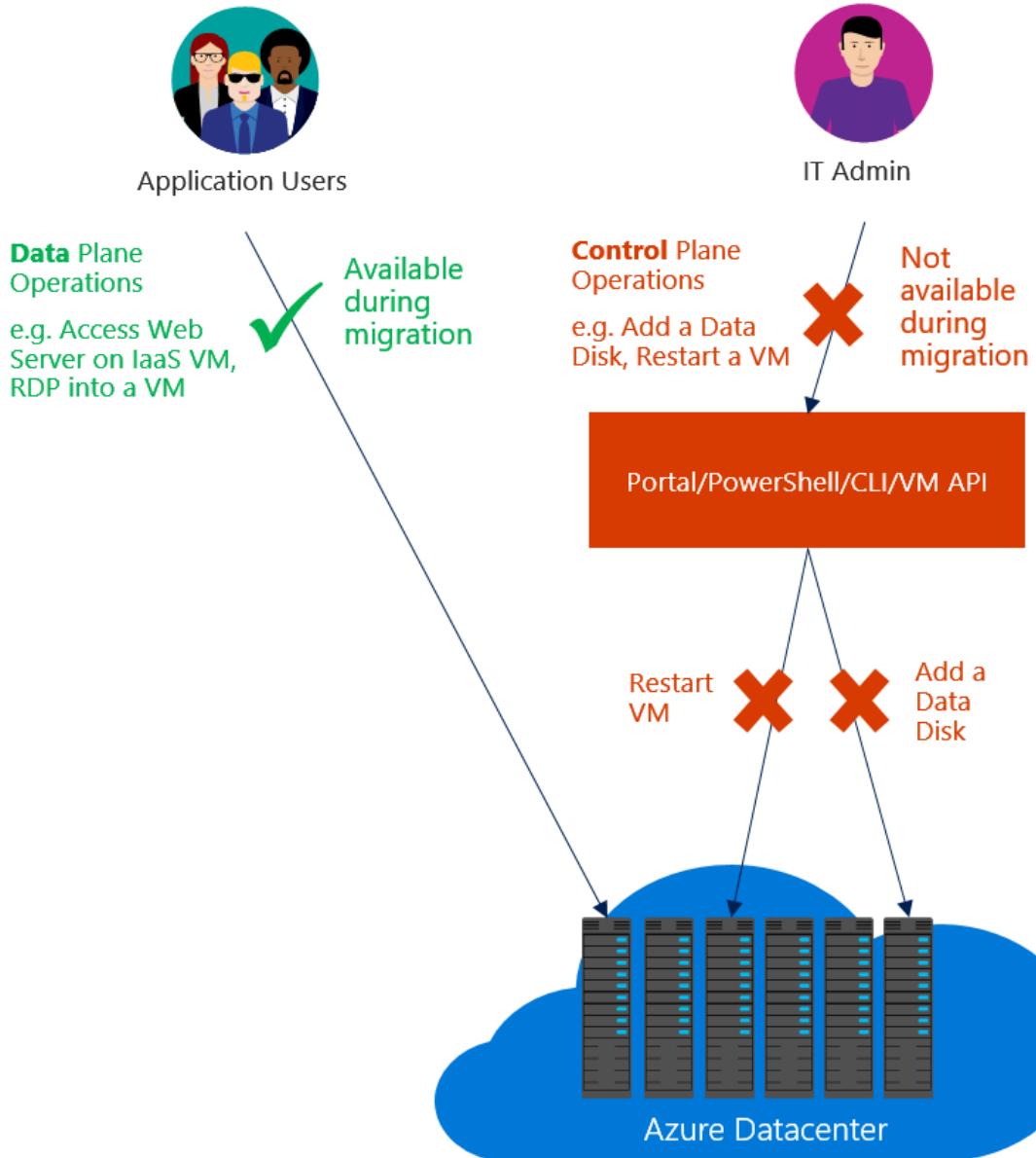
Let's take a deep-dive on migrating from the Azure classic deployment model to the Azure Resource Manager deployment model. We look at resources at a resource and feature level to help you understand how the Azure platform migrates resources between the two deployment models. For more information, please read the service announcement article: [Platform-supported migration of IaaS resources from classic to Azure Resource Manager](#).

Migrate IaaS resources from the classic deployment model to Azure Resource Manager

First, it's important to understand the difference between data-plane and management-plane operations on the infrastructure as a service (IaaS) resources.

- *Management/control plane* describes the calls that come into the management/control plane or the API for modifying resources. For example, operations like creating a VM, restarting a VM, and updating a virtual network with a new subnet manage the running resources. They don't directly affect connecting to the VMs.
- *Data plane* (application) describes the runtime of the application itself, and involves interaction with instances that don't go through the Azure API. For example, accessing your website, or pulling data from a running SQL Server instance or a MongoDB server, are data plane or application interactions. Other examples include copying a blob from a storage account, and accessing a public IP address to use Remote Desktop Protocol (RDP) or Secure Shell (SSH) into the virtual machine. These operations keep the application running across compute, networking, and storage.

The data plane is the same between the classic deployment model and Resource Manager stacks. The difference is that during the migration process, Microsoft translates the representation of the resources from the classic deployment model to that in the Resource Manager stack. As a result, you need to use new tools, APIs, and SDKs to manage your resources in the Resource Manager stack.



NOTE

In some migration scenarios, the Azure platform stops, deallocates, and restarts your virtual machines. This causes a brief data-plane downtime.

The migration experience

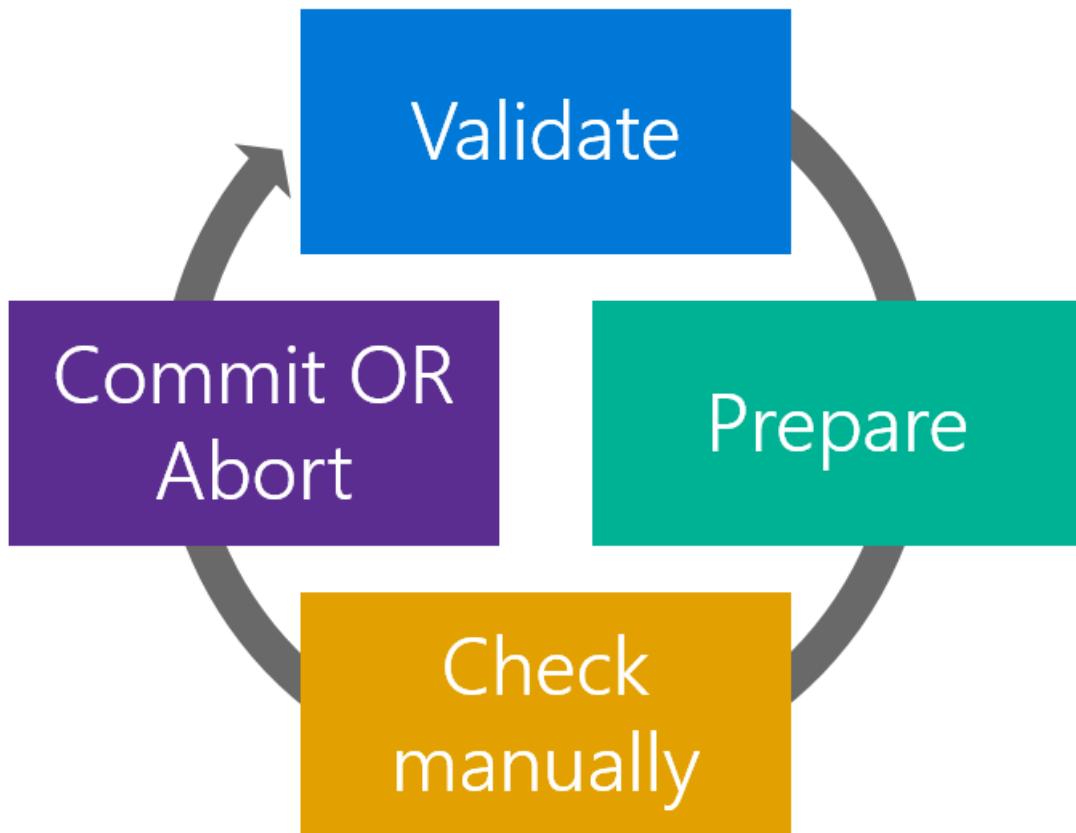
Before you start the migration:

- Ensure that the resources that you want to migrate don't use any unsupported features or configurations. Usually the platform detects these issues and generates an error.
- If you have VMs that are not in a virtual network, they are stopped and deallocated as part of the prepare operation. If you don't want to lose the public IP address, consider reserving the IP address before triggering the prepare operation. If the VMs are in a virtual network, they are not stopped and deallocated.
- Plan your migration during non-business hours to accommodate for any unexpected failures that might happen during migration.
- Download the current configuration of your VMs by using PowerShell, command-line interface (CLI) commands, or REST APIs to make it easier for validation after the prepare step is complete.
- Update your automation and operationalization scripts to handle the Resource Manager deployment model,

before you start the migration. You can optionally do GET operations when the resources are in the prepared state.

- Evaluate the Role-Based Access Control (RBAC) policies that are configured on the IaaS resources in the classic deployment model, and plan for after the migration is complete.

The migration workflow is as follows:



NOTE

The operations described in the following sections are all idempotent. If you have a problem other than an unsupported feature or a configuration error, retry the prepare, abort, or commit operation. Azure tries the action again.

Validate

The validate operation is the first step in the migration process. The goal of this step is to analyze the state of the resources you want to migrate in the classic deployment model. The operation evaluates whether the resources are capable of migration (success or failure).

You select the virtual network or a cloud service (if it's not in a virtual network) that you want to validate for migration. If the resource is not capable of migration, Azure lists the reasons why.

Checks not done in the validate operation

The validate operation only analyzes the state of the resources in the classic deployment model. It can check for all failures and unsupported scenarios due to various configurations in the classic deployment model. It is not possible to check for all issues that the Azure Resource Manager stack might impose on the resources during migration. These issues are only checked when the resources undergo transformation in the next step of migration (the prepare operation). The following table lists all the issues not checked in the validate operation:

NETWORKING CHECKS NOT IN THE VALIDATE OPERATION

A virtual network having both ER and VPN gateways.

A virtual network gateway connection in a disconnected state.

All ER circuits are pre-migrated to Azure Resource Manager stack.

Azure Resource Manager quota checks for networking resources. For example: static public IP, dynamic public IPs, load balancer, network security groups, route tables, and network interfaces.

All load balancer rules are valid across deployment and the virtual network.

Conflicting private IPs between stop-deallocated VMs in the same virtual network.

Prepare

The prepare operation is the second step in the migration process. The goal of this step is to simulate the transformation of the IaaS resources from the classic deployment model to Resource Manager resources. Further, the prepare operation presents this side-by-side for you to visualize.

NOTE

Your resources in the classic deployment model are not modified during this step. It's a safe step to run if you're trying out migration.

You select the virtual network or the cloud service (if it's not a virtual network) that you want to prepare for migration.

- If the resource is not capable of migration, Azure stops the migration process and lists the reason why the prepare operation failed.
- If the resource is capable of migration, Azure locks down the management-plane operations for the resources under migration. For example, you are not able to add a data disk to a VM under migration.

Azure then starts the migration of metadata from the classic deployment model to Resource Manager for the migrating resources.

After the prepare operation is complete, you have the option of visualizing the resources in both the classic deployment model and Resource Manager. For every cloud service in the classic deployment model, the Azure platform creates a resource group name that has the pattern `cloud-service-name>-Migrated`.

NOTE

It is not possible to select the name of a resource group created for migrated resources (that is, "-Migrated"). After migration is complete, however, you can use the move feature of Azure Resource Manager to move resources to any resource group you want. For more information, see [Move resources to new resource group or subscription](#).

The following two screenshots show the result after a successful prepare operation. The first one shows a resource group that contains the original cloud service. The second one shows the new "-Migrated" resource group that contains the equivalent Azure Resource Manager resources.

portalmigrate
Resource group

Search (Ctrl+ /)

Overview

Activity log

Access control (IAM)

Tags

SETTINGS

Quickstart

Resource costs

Deployments

Properties

Locks

Automation script

Essentials

Subscription name (change) Subscription ID

Deployments No deployments Location East US

Filter by name...

2 items

NAME	TYPE	LOCATION
portalmigrate	Cloud service (class...)	East US
portalmigrate	Virtual machine (cl...)	East US

portalmigrate-Migrated
Resource group

Search (Ctrl+ /)

Overview

Activity log

Access control (IAM)

Tags

SETTINGS

Quickstart

Resource costs

Deployments

Properties

Locks

Automation script

Essentials

Subscription name (change) Subscription ID

Deployments 2 Succeeded Location East US

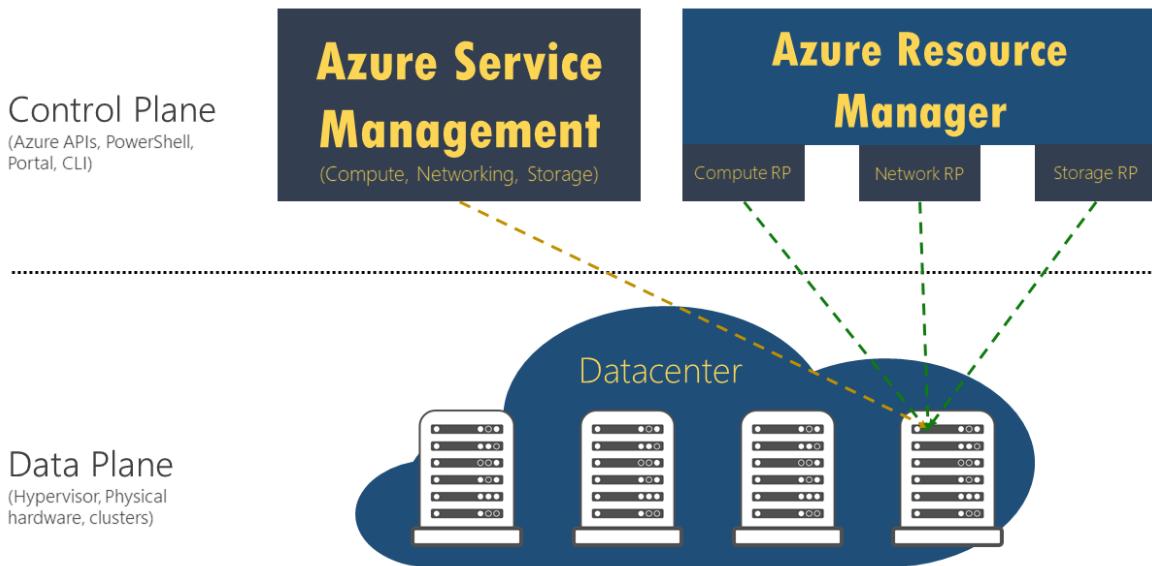
Filter by name...

5 items

NAME	TYPE	LOCATION
portalmigrate	Virtual machine	East US
portalmigrate-PrimaryNic	Network interface	East US
portalmigrate-PrimaryVirtualIP	Public IP address	East US
portalmigrate-PublicLoadBalancer	Load balancer	East US
portalmigrate-VirtualNetwork	Virtual network	East US

Here is a behind-the-scenes look at your resources after the completion of the prepare phase. Note that the resource in the data plane is the same. It's represented in both the management plane (classic deployment model) and the control plane (Resource Manager).

Prepare



NOTE

VMs that are not in a virtual network in the classic deployment model are stopped and deallocated in this phase of migration.

Check (manual or scripted)

In the check step, you have the option to use the configuration that you downloaded earlier to validate that the migration looks correct. Alternatively, you can sign in to the portal, and spot check the properties and resources to validate that metadata migration looks good.

If you are migrating a virtual network, most configuration of virtual machines is not restarted. For applications on those VMs, you can validate that the application is still running.

You can test your monitoring and operational scripts to see if the VMs are working as expected, and if your updated scripts work correctly. Only GET operations are supported when the resources are in the prepared state.

There is no set window of time before which you need to commit the migration. You can take as much time as you want in this state. However, the management plane is locked for these resources until you either abort or commit.

If you see any issues, you can always abort the migration and go back to the classic deployment model. After you go back, Azure opens the management-plane operations on the resources, so that you can resume normal operations on those VMs in the classic deployment model.

Abort

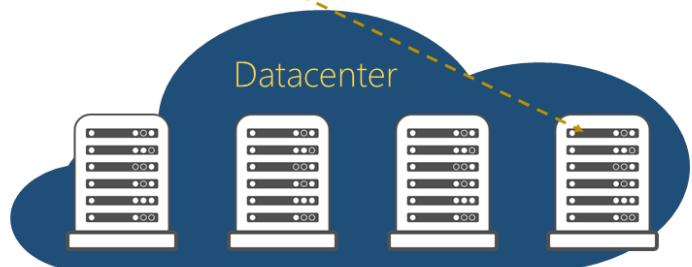
This is an optional step if you want to revert your changes to the classic deployment model and stop the migration. This operation deletes the Resource Manager metadata (created in the prepare step) for your resources.

Abort

Control Plane
(Azure APIs, PowerShell, Portal, CLI)



Data Plane
(Hypervisor, Physical hardware, clusters)



NOTE

This operation can't be done after you have triggered the commit operation.

Commit

After you finish the validation, you can commit the migration. Resources do not appear anymore in the classic deployment model, and are available only in the Resource Manager deployment model. The migrated resources can be managed only in the new portal.

NOTE

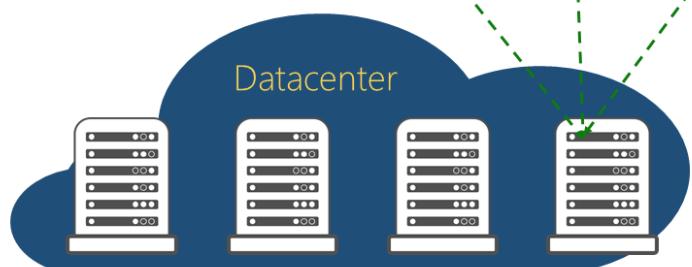
This is an idempotent operation. If it fails, retry the operation. If it continues to fail, create a support ticket or create a forum post with a "ClassiclaaSMigration" tag on our [VM forum](#).

Commit

Control Plane
(Azure APIs, PowerShell, Portal, CLI)



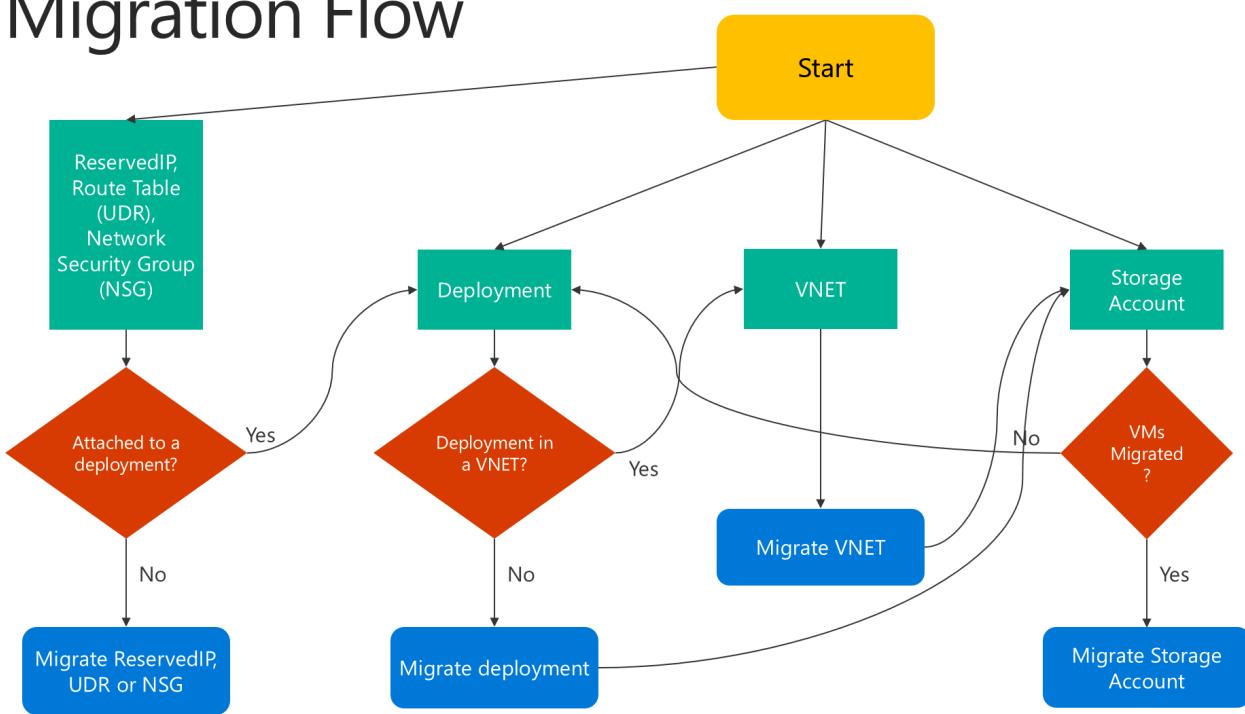
Data Plane
(Hypervisor, Physical hardware, clusters)



Migration flowchart

Here is a flowchart that shows how to proceed with migration:

Migration Flow



Translation of the classic deployment model to Resource Manager resources

You can find the classic deployment model and Resource Manager representations of the resources in the following table. Other features and resources are not currently supported.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	NOTES
Cloud service name	DNS name	During migration, a new resource group is created for every cloud service with the naming pattern <code><cloudservicename>-migrated</code> . This resource group contains all your resources. The cloud service name becomes a DNS name that is associated with the public IP address.
Virtual machine	Virtual machine	VM-specific properties are migrated unchanged. Certain osProfile information, like computer name, is not stored in the classic deployment model, and remains empty after migration.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	NOTES
Disk resources attached to VM	Implicit disks attached to VM	Disks are not modeled as top-level resources in the Resource Manager deployment model. They are migrated as implicit disks under the VM. Only disks that are attached to a VM are currently supported. Resource Manager VMs can now use storage accounts in the classic deployment model, which allows the disks to be easily migrated without any updates.
VM extensions	VM extensions	All the resource extensions, except XML extensions, are migrated from the classic deployment model.
Virtual machine certificates	Certificates in Azure Key Vault	If a cloud service contains service certificates, the migration creates a new Azure key vault per cloud service, and moves the certificates into the key vault. The VMs are updated to reference the certificates from the key vault. Do not delete the key vault. This can cause the VM to go into a failed state.
WinRM configuration	WinRM configuration under osProfile	Windows Remote Management configuration is moved unchanged, as part of the migration.
Availability-set property	Availability-set resource	Availability-set specification is a property on the VM in the classic deployment model. Availability sets become a top-level resource as part of the migration. The following configurations are not supported: multiple availability sets per cloud service, or one or more availability sets along with VMs that are not in any availability set in a cloud service.
Network configuration on a VM	Primary network interface	Network configuration on a VM is represented as the primary network interface resource after migration. For VMs that are not in a virtual network, the internal IP address changes during migration.
Multiple network interfaces on a VM	Network interfaces	If a VM has multiple network interfaces associated with it, each network interface becomes a top-level resource as part of the migration, along with all the properties.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	NOTES
Load-balanced endpoint set	Load balancer	In the classic deployment model, the platform assigned an implicit load balancer for every cloud service. During migration, a new load-balancer resource is created, and the load-balancing endpoint set becomes load-balancer rules.
Inbound NAT rules	Inbound NAT rules	Input endpoints defined on the VM are converted to inbound network address translation rules under the load balancer during the migration.
VIP address	Public IP address with DNS name	The virtual IP address becomes a public IP address, and is associated with the load balancer. A virtual IP can only be migrated if there is an input endpoint assigned to it.
Virtual network	Virtual network	The virtual network is migrated, with all its properties, to the Resource Manager deployment model. A new resource group is created with the name <code>-migrated</code> .
Reserved IPs	Public IP address with static allocation method	Reserved IPs associated with the load balancer are migrated, along with the migration of the cloud service or the virtual machine. Unassociated reserved IP migration is not currently supported.
Public IP address per VM	Public IP address with dynamic allocation method	The public IP address associated with the VM is converted as a public IP address resource, with the allocation method set to static.
NSGs	NSGs	Network security groups associated with a subnet are cloned as part of the migration to the Resource Manager deployment model. The NSG in the classic deployment model is not removed during the migration. However, the management-plane operations for the NSG are blocked when the migration is in progress.
DNS servers	DNS servers	DNS servers associated with a virtual network or the VM are migrated as part of the corresponding resource migration, along with all the properties.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	NOTES
UDRs	UDRs	User-defined routes associated with a subnet are cloned as part of the migration to the Resource Manager deployment model. The UDR in the classic deployment model is not removed during the migration. The management-plane operations for the UDR are blocked when the migration is in progress.
IP forwarding property on a VM's network configuration	IP forwarding property on the NIC	The IP forwarding property on a VM is converted to a property on the network interface during the migration.
Load balancer with multiple IPs	Load balancer with multiple public IP resources	Every public IP associated with the load balancer is converted to a public IP resource, and associated with the load balancer after migration.
Internal DNS names on the VM	Internal DNS names on the NIC	During migration, the internal DNS suffixes for the VMs are migrated to a read-only property named "InternalDomainNameSuffix" on the NIC. The suffix remains unchanged after migration, and VM resolution should continue to work as previously.
Virtual network gateway	Virtual network gateway	Virtual network gateway properties are migrated unchanged. The VIP associated with the gateway does not change either.
Local network site	Local network gateway	Local network site properties are migrated unchanged to a new resource called a local network gateway. This represents on-premises address prefixes and the remote gateway IP.
Connections references	Connection	Connectivity references between the gateway and the local network site in network configuration is represented by a new resource called Connection. All properties of connectivity reference in network configuration files are copied unchanged to the Connection resource. Connectivity between virtual networks in the classic deployment model is achieved by creating two IPsec tunnels to local network sites representing the virtual networks. This is transformed to the virtual-network-to-virtual-network connection type in the Resource Manager model, without requiring local network gateways.

Changes to your automation and tooling after migration

As part of migrating your resources from the classic deployment model to the Resource Manager deployment

model, you must update your existing automation or tooling to ensure that it continues to work after the migration.

Next steps

- [Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager](#)
- [Planning for migration of IaaS resources from classic to Azure Resource Manager](#)
- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager](#)
- [Review most common migration errors](#)
- [Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager](#)

Planning for migration of IaaS resources from classic to Azure Resource Manager

11/15/2017 • 13 min to read • [Edit Online](#)

While Azure Resource Manager offers many amazing features, it is critical to plan out your migration journey to make sure things go smoothly. Spending time on planning will ensure that you do not encounter issues while executing migration activities.

NOTE

The following guidance was heavily contributed to by the Azure Customer Advisory team and Cloud Solution architects working with customers on migrating large environments. As such this document will continue to get updated as new patterns of success emerge, so check back from time to time to see if there are any new recommendations.

There are four general phases of the migration journey:



Plan

Technical considerations and tradeoffs

Depending on your technical requirements size, geographies and operational practices, you might want to consider:

1. Why is Azure Resource Manager desired for your organization? What are the business reasons for a migration?
2. What are the technical reasons for Azure Resource Manager? What (if any) additional Azure services would you like to leverage?
3. Which application (or sets of virtual machines) is included in the migration?
4. Which scenarios are supported with the migration API? Review the [unsupported features and configurations](#).
5. Will your operational teams now support applications/VMs in both Classic and Azure Resource Manager?
6. How (if at all) does Azure Resource Manager change your VM deployment, management, monitoring, and reporting processes? Do your deployment scripts need to be updated?
7. What is the communications plan to alert stakeholders (end users, application owners, and infrastructure owners)?
8. Depending on the complexity of the environment, should there be a maintenance period where the application is unavailable to end users and to application owners? If so, for how long?
9. What is the training plan to ensure stakeholders are knowledgeable and proficient in Azure Resource Manager?
10. What is the program management or project management plan for the migration?
11. What are the timelines for the Azure Resource Manager migration and other related technology road maps? Are they optimally aligned?

Patterns of success

Successful customers have detailed plans where the preceding questions are discussed, documented and governed. Ensure the migration plans are broadly communicated to sponsors and stakeholders. Equip yourself with knowledge about your migration options; reading through this migration document set below is highly

recommended.

- Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager
- Technical deep dive on platform-supported migration from classic to Azure Resource Manager
- Planning for migration of IaaS resources from classic to Azure Resource Manager
- Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager
- Use CLI to migrate IaaS resources from classic to Azure Resource Manager
- Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager
- Review most common migration errors
- Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager

Pitfalls to avoid

- Failure to plan. The technology steps of this migration are proven and the outcome is predictable.
- Assumption that the platform supported migration API will account for all scenarios. Read the [unsupported features and configurations](#) to understand what scenarios are supported.
- Not planning potential application outage for end users. Plan enough buffer to adequately warn end users of potentially unavailable application time.

Lab Test

Replicate your environment and do a test migration

NOTE

Exact replication of your existing environment is executed by using a community-contributed tool which is not officially supported by Microsoft Support. Therefore, it is an **optional** step but it is the best way to find out issues without touching your production environments. If using a community-contributed tool is not an option, then read about the Validate/Prepare/Abort Dry Run recommendation below.

Conducting a lab test of your exact scenario (compute, networking, and storage) is the best way to ensure a smooth migration. This will help ensure:

- A wholly separate lab or an existing non-production environment to test. We recommend a wholly separate lab that can be migrated repeatedly and can be destructively modified. Scripts to collect/hydrate metadata from the real subscriptions are listed below.
- It's a good idea to create the lab in a separate subscription. The reason is that the lab will be torn down repeatedly, and having a separate, isolated subscription will reduce the chance that something real will get accidentally deleted.

This can be accomplished by using the AsmMetadataParser tool. [Read more about this tool here](#)

Patterns of success

The following were issues discovered in many of the larger migrations. This is not an exhaustive list and you should refer to the [unsupported features and configurations](#) for more detail. You may or may not encounter these technical issues but if you do solving these before attempting migration will ensure a smoother experience.

- **Do a Validate/Prepare/Abort Dry Run** - This is perhaps the most important step to ensure Classic to Azure Resource Manager migration success. The migration API has three main steps: Validate, Prepare and Commit. Validate will read the state of your classic environment and return a result of all issues. However, because some issues might exist in the Azure Resource Manager stack, Validate will not catch everything. The next step in migration process, Prepare will help expose those issues. Prepare will move the metadata from Classic to Azure Resource Manager, but will not commit the move, and will not remove or change

anything on the Classic side. The dry run involves preparing the migration, then aborting (**not committing**) the migration prepare. The goal of validate/prepare/abort dry run is to see all of the metadata in the Azure Resource Manager stack, examine it (*programmatically or in Portal*), and verify that everything migrates correctly, and work through technical issues. It will also give you a sense of migration duration so you can plan for downtime accordingly. A validate/prepare/abort does not cause any user downtime; therefore, it is non-disruptive to application usage.

- The items below will need to be solved before the dry run, but a dry run test will also safely flush out these preparation steps if they are missed. During enterprise migration, we've found the dry run to be a safe and invaluable way to ensure migration readiness.
- When prepare is running, the control plane (Azure management operations) will be locked for the whole virtual network, so no changes can be made to VM metadata during validate/prepare/abort. But otherwise any application function (RD, VM usage, etc.) will be unaffected. Users of the VMs will not know that the dry run is being executed.

- **Express Route Circuits and VPN.** Currently Express Route Gateways with authorization links cannot be migrated without downtime. For the workaround, see [Migrate ExpressRoute circuits and associated virtual networks from the classic to the Resource Manager deployment model](#).

- **VM Extensions** - Virtual Machine extensions are potentially one of the biggest roadblocks to migrating running VMs. Remediation of VM Extensions could take upwards of 1-2 days, so plan accordingly. A working Azure agent is needed to report back VM Extension status of running VMs. If the status comes back as bad for a running VM, this will halt migration. The agent itself does not need to be in working order to enable migration, but if extensions exist on the VM, then both a working agent AND outbound internet connectivity (with DNS) will be needed for migration to move forward.

- If connectivity to a DNS server is lost during migration, all VM Extensions except BGInfo version 1.* need to first be removed from every VM before migration prepare, and subsequently re-added back to the VM after Azure Resource Manager migration. **This is only for VMs that are running.** If the VMs are stopped deallocated, VM Extensions do not need to be removed.

NOTE

Many extensions like Azure diagnostics and security center monitoring will reinstall themselves after migration, so removing them is not a problem.

- In addition, make sure Network Security Groups are not restricting outbound internet access. This can happen with some Network Security Groups configurations. Outbound internet access (and DNS) is needed for VM Extensions to be migrated to Azure Resource Manager.
- Two versions of the BGInfo extension exist and are called versions 1 and 2.
 - If the VM is using the BGInfo version 1 extension, you can leave this extension as is. The migration API skips this extension. The BGInfo extension can be added after migration.
 - If the VM is using the JSON-based BGInfo version 2 extension, the VM was created using the Azure portal. The migration API includes this extension in the migration to Azure Resource Manager, provided the agent is working and has outbound internet access (and DNS).
- **Remediation Option 1.** If you know your VMs will not have outbound internet access, a working DNS service, and working Azure agents on the VMs, then uninstall all VM extensions as part of the migration before Prepare, then reinstall the VM Extensions after migration.
- **Remediation Option 2.** If VM extensions are too big of a hurdle, another option is to shutdown/deallocate all VMs before migration. Migrate the deallocated VMs, then restart them on the Azure Resource Manager side. The benefit here is that VM extensions will migrate. The downside is that all public facing Virtual IPs will be lost (this may be a non-starter), and obviously the VMs will shut down causing a much greater impact on working applications.

NOTE

If an Azure Security Center policy is configured against the running VMs being migrated, the security policy needs to be stopped before removing extensions, otherwise the security monitoring extension will be reinstalled automatically on the VM after removing it.

- **Availability Sets** - For a virtual network (vNet) to be migrated to Azure Resource Manager, the Classic deployment (i.e. cloud service) contained VMs must all be in one availability set, or the VMs must all not be in any availability set. Having more than one availability set in the cloud service is not compatible with Azure Resource Manager and will halt migration. Additionally, there cannot be some VMs in an availability set, and some VMs not in an availability set. To resolve this, you will need to remediate or reshuffle your cloud service. Plan accordingly as this might be time consuming.
- **Web/Worker Role Deployments** - Cloud Services containing web and worker roles cannot migrate to Azure Resource Manager. The web/worker roles must first be removed from the virtual network before migration can start. A typical solution is to just move web/worker role instances to a separate Classic virtual network that is also linked to an ExpressRoute circuit, or to migrate the code to newer PaaS App Services (this discussion is beyond the scope of this document). In the former redeploy case, create a new Classic virtual network, move/redeploy the web/worker roles to that new virtual network, then delete the deployments from the virtual network being moved. No code changes required. The new [Virtual Network Peering](#) capability can be used to peer together the classic virtual network containing the web/worker roles and other virtual networks in the same Azure region such as the virtual network being migrated (**after virtual network migration is completed as peered virtual networks cannot be migrated**), hence providing the same capabilities with no performance loss and no latency/bandwidth penalties. Given the addition of [Virtual Network Peering](#), web/worker role deployments can now easily be mitigated and not block the migration to Azure Resource Manager.
- **Azure Resource Manager Quotas** - Azure regions have separate quotas/limits for both Classic and Azure Resource Manager. Even though in a migration scenario new hardware isn't being consumed (*we're swapping existing VMs from Classic to Azure Resource Manager*), Azure Resource Manager quotas still need to be in place with enough capacity before migration can start. Listed below are the major limits we've seen cause problems. Open a quota support ticket to raise the limits.

NOTE

These limits need to be raised in the same region as your current environment to be migrated.

- Network Interfaces
- Load Balancers
- Public IPs
- Static Public IPs
- Cores
- Network Security Groups
- Route Tables

You can check your current Azure Resource Manager quotas using the following commands with the latest version of Azure PowerShell.

Compute (Cores, Availability Sets)

```
Get-AzureRmVMUsage -Location <azure-region>
```

Network (*Virtual Networks, Static Public IPs, Public IPs, Network Security Groups, Network Interfaces, Load Balancers, Route Tables*)

```
Get-AzureRmUsage /subscriptions/<subscription-id>/providers/Microsoft.Network/locations/<azure-region> -ApiVersion 2016-03-30 | Format-Table
```

Storage (Storage Account)

```
Get-AzureRmStorageUsage
```

- **Azure Resource Manager API throttling limits** - If you have a large enough environment (eg. > 400 VMs in a VNET), you might hit the default API throttling limits for writes (currently `1200 writes/hour`) in Azure Resource Manager. Before starting migration, you should raise a support ticket to increase this limit for your subscription.
- **Provisioning Timed Out VM Status** - If any VM has the status of `provisioning timed out`, this needs to be resolved pre-migration. The only way to do this is with downtime by deprovisioning/reprovisioning the VM (delete it, keep the disk, and recreate the VM).
- **RoleStateUnknown VM Status** - If migration halts due to a `role state unknown` error message, inspect the VM using the portal and ensure it is running. This error will typically go away on its own (no remediation required) after a few minutes and is often a transient type often seen during a Virtual Machine `start`, `stop`, `restart` operations. **Recommended practice:** re-try migration again after a few minutes.
- **Fabric Cluster does not exist** - In some cases, certain VMs cannot be migrated for various odd reasons. One of these known cases is if the VM was recently created (within the last week or so) and happened to land an Azure cluster that is not yet equipped for Azure Resource Manager workloads. You will get an error that says `fabric cluster does not exist` and the VM cannot be migrated. Waiting a couple of days will usually resolve this particular problem as the cluster will soon get Azure Resource Manager enabled. However, one immediate workaround is to `stop-deallocate` the VM, then continue forward with migration, and start the VM back up in Azure Resource Manager after migrating.

Pitfalls to avoid

- Do not take shortcuts and omit the validate/prepare/abort dry run migrations.
- Most, if not all, of your potential issues will surface during the validate/prepare/abort steps.

Migration

Technical considerations and tradeoffs

Now you are ready because you have worked through the known issues with your environment.

For the real migrations, you might want to consider:

1. Plan and schedule the virtual network (smallest unit of migration) with increasing priority. Do the simple virtual networks first, and progress with the more complicated virtual networks.
2. Most customers will have non-production and production environments. Schedule production last.
3. **(OPTIONAL)** Schedule a maintenance downtime with plenty of buffer in case unexpected issues arise.
4. Communicate with and align with your support teams in case issues arise.

Patterns of success

The technical guidance from the *Lab Test* section should be considered and mitigated prior to a real migration. With adequate testing, the migration is actually a non-event. For production environments, it might be helpful to have additional support, such as a trusted Microsoft partner or Microsoft Premier services.

Pitfalls to avoid

Not fully testing may cause issues and delay in the migration.

Beyond Migration

Technical considerations and tradeoffs

Now that you are in Azure Resource Manager, maximize the platform. Read the [overview of Azure Resource Manager](#) to find out about additional benefits.

Things to consider:

- Bundling the migration with other activities. Most customers opt for an application maintenance window. If so, you might want to use this downtime to enable other Azure Resource Manager capabilities like encryption and migration to Managed Disks.
- Revisit the technical and business reasons for Azure Resource Manager; enable the additional services available only on Azure Resource Manager that apply to your environment.
- Modernize your environment with PaaS services.

Patterns of success

Be purposeful on what services you now want to enable in Azure Resource Manager. Many customers find the below compelling for their Azure environments:

- [Role Based Access Control](#).
- [Azure Resource Manager templates for easier and more controlled deployment](#).
- [Tags](#).
- [Activity Control](#)
- [Azure Policies](#)

Pitfalls to avoid

Remember why you started this Classic to Azure Resource Manager migration journey. What were the original business reasons? Did you achieve the business reason?

Next steps

- [Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager](#)
- [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#)
- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager](#)
- [Review most common migration errors](#)
- [Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager](#)

Migrate IaaS resources from classic to Azure Resource Manager by using Azure PowerShell

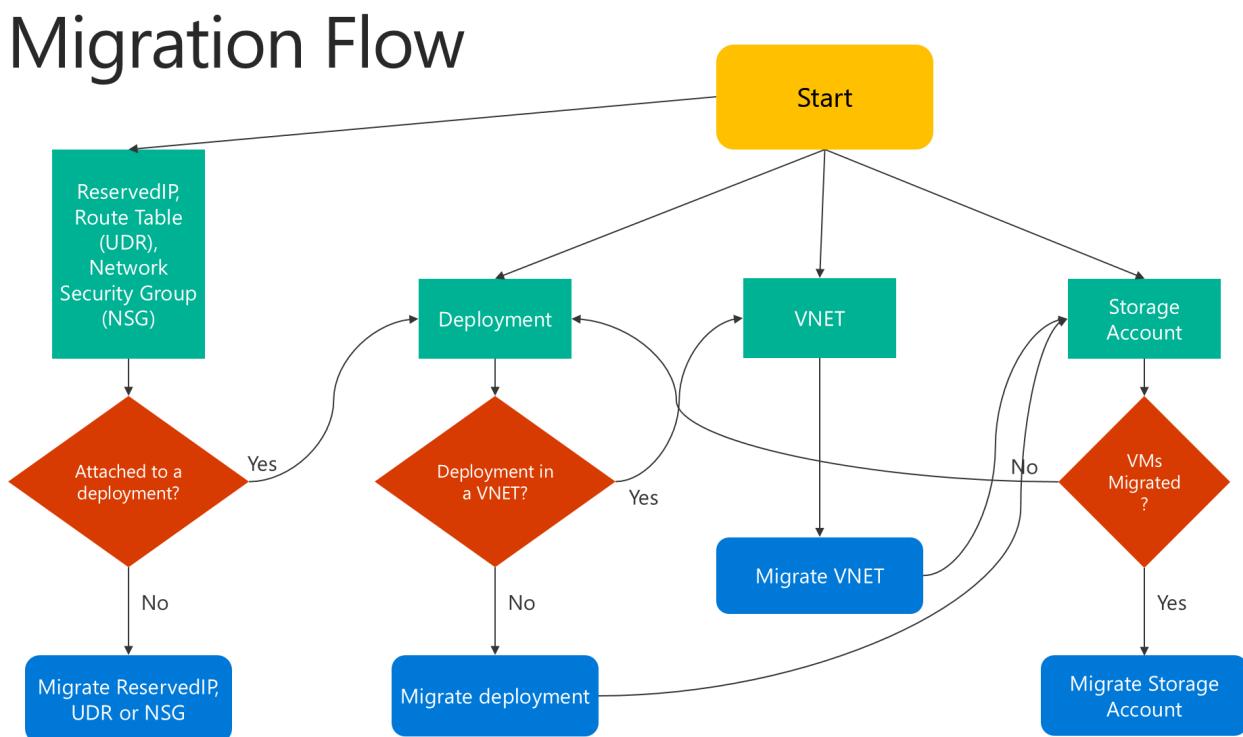
11/16/2017 • 10 min to read • [Edit Online](#)

These steps show you how to use Azure PowerShell commands to migrate infrastructure as a service (IaaS) resources from the classic deployment model to the Azure Resource Manager deployment model.

If you want, you can also migrate resources by using the [Azure Command Line Interface \(Azure CLI\)](#).

- For background on supported migration scenarios, see [Platform-supported migration of IaaS resources from classic to Azure Resource Manager](#).
- For detailed guidance and a migration walkthrough, see [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#).
- [Review most common migration errors](#)

Here is a flowchart to identify the order in which steps need to be executed during a migration process



Step 1: Plan for migration

Here are a few best practices that we recommend as you evaluate migrating IaaS resources from classic to Resource Manager:

- Read through the [supported and unsupported features and configurations](#). If you have virtual machines that use unsupported configurations or features, we recommend that you wait for the configuration/feature support to be announced. Alternatively, if it suits your needs, remove that feature or move out of that configuration to enable migration.
- If you have automated scripts that deploy your infrastructure and applications today, try to create a similar test setup by using those scripts for migration. Alternatively, you can set up sample environments by using the Azure portal.

IMPORTANT

Application Gateways are not currently supported for migration from classic to Resource Manager. To migrate a classic virtual network with an Application gateway, remove the gateway before running a Prepare operation to move the network. After you complete the migration, reconnect the gateway in Azure Resource Manager.

ExpressRoute gateways connecting to ExpressRoute circuits in another subscription cannot be migrated automatically. In such cases, remove the ExpressRoute gateway, migrate the virtual network and recreate the gateway. Please see [Migrate ExpressRoute circuits and associated virtual networks from the classic to the Resource Manager deployment model](#) for more information.

Step 2: Install the latest version of Azure PowerShell

There are two main options to install Azure PowerShell: [PowerShell Gallery](#) or [Web Platform Installer \(WebPI\)](#). WebPI receives monthly updates. PowerShell Gallery receives updates on a continuous basis. This article is based on Azure PowerShell version 2.1.0.

For installation instructions, see [How to install and configure Azure PowerShell](#).

Step 3: Ensure that you are an administrator for the subscription in Azure portal

To perform this migration, you must be added as a co-administrator for the subscription in the [Azure portal](#).

1. Sign into the [Azure portal](#).
2. On the Hub menu, select **Subscription**. If you don't see it, select **More services**.
3. Find the appropriate subscription entry, then look at the **MY ROLE** field. For a co-administrator, the value should be *Account admin*.

If you are not able to add a co-administrator, then contact a service administrator or co-administrator for the subscription to get yourself added.

Step 4: Set your subscription and sign up for migration

First, start a PowerShell prompt. For migration, you need to set up your environment for both classic and Resource Manager.

Sign in to your account for the Resource Manager model.

```
Login-AzureRmAccount
```

Get the available subscriptions by using the following command:

```
Get-AzureRMSubscription | Sort Name | Select Name
```

Set your Azure subscription for the current session. This example sets the default subscription name to **My Azure Subscription**. Replace the example subscription name with your own.

```
Select-AzureRmSubscription -SubscriptionName "My Azure Subscription"
```

NOTE

Registration is a one-time step, but you must do it once before attempting migration. Without registering, you see the following error message:

BadRequest : Subscription is not registered for migration.

Register with the migration resource provider by using the following command:

```
Register-AzureRmResourceProvider -ProviderNamespace Microsoft.ClassicInfrastructureMigrate
```

Please wait five minutes for the registration to finish. You can check the status of the approval by using the following command:

```
Get-AzureRmResourceProvider -ProviderNamespace Microsoft.ClassicInfrastructureMigrate
```

Make sure that RegistrationState is `Registered` before you proceed.

Now sign in to your account for the classic model.

```
Add-AzureAccount
```

Get the available subscriptions by using the following command:

```
Get-AzureSubscription | Sort SubscriptionName | Select SubscriptionName
```

Set your Azure subscription for the current session. This example sets the default subscription to **My Azure Subscription**. Replace the example subscription name with your own.

```
Select-AzureSubscription -SubscriptionName "My Azure Subscription"
```

Step 5: Make sure you have enough Azure Resource Manager Virtual Machine vCPUs in the Azure region of your current deployment or VNET

You can use the following PowerShell command to check the current number of vCPUs you have in Azure Resource Manager. To learn more about vCPU quotas, see [Limits and the Azure Resource Manager](#).

This example checks the availability in the **West US** region. Replace the example region name with your own.

```
Get-AzureRmVMUsage -Location "West US"
```

Step 6: Run commands to migrate your IaaS resources

NOTE

All the operations described here are idempotent. If you have a problem other than an unsupported feature or a configuration error, we recommend that you retry the prepare, abort, or commit operation. The platform then tries the action again.

Step 6.1: Option 1 - Migrate virtual machines in a cloud service (not in a virtual network)

Get the list of cloud services by using the following command, and then pick the cloud service that you want to migrate. If the VMs in the cloud service are in a virtual network or if they have web or worker roles, the command returns an error message.

```
Get-AzureService | ft Servicename
```

Get the deployment name for the cloud service. In this example, the service name is **My Service**. Replace the example service name with your own service name.

```
$serviceName = "My Service"  
$deployment = Get-AzureDeployment -ServiceName $serviceName  
$deploymentName = $deployment.DeploymentName
```

Prepare the virtual machines in the cloud service for migration. You have two options to choose from.

- **Option 1. Migrate the VMs to a platform-created virtual network**

First, validate if you can migrate the cloud service using the following commands:

```
$validate = Move-AzureService -Validate -ServiceName $serviceName `  
    -DeploymentName $deploymentName -CreateNewVirtualNetwork  
$validate.ValidationMessages
```

The preceding command displays any warnings and errors that block migration. If validation is successful, then you can move on to the **Prepare** step:

```
Move-AzureService -Prepare -ServiceName $serviceName `  
    -DeploymentName $deploymentName -CreateNewVirtualNetwork
```

- **Option 2. Migrate to an existing virtual network in the Resource Manager deployment model**

This example sets the resource group name to **myResourceGroup**, the virtual network name to **myVirtualNetwork** and the subnet name to **mySubNet**. Replace the names in the example with the names of your own resources.

```
$existingVnetRGName = "myResourceGroup"  
$vnetName = "myVirtualNetwork"  
$subnetName = "mySubNet"
```

First, validate if you can migrate the virtual network using the following command:

```
$validate = Move-AzureService -Validate -ServiceName $serviceName `  
    -DeploymentName $deploymentName -UseExistingVirtualNetwork -VirtualNetworkResourceGroupName  
$existingVnetRGName -VirtualNetworkName $vnetName -SubnetName $subnetName  
$validate.ValidationMessages
```

The preceding command displays any warnings and errors that block migration. If validation is successful, then you can proceed with the following Prepare step:

```
Move-AzureService -Prepare -ServiceName $serviceName -DeploymentName $deploymentName `  
    -UseExistingVirtualNetwork -VirtualNetworkResourceGroupName $existingVnetRGName `  
    -VirtualNetworkName $vnetName -SubnetName $subnetName
```

After the Prepare operation succeeds with either of the preceding options, query the migration state of the VMs. Ensure that they are in the **Prepared** state.

This example sets the VM name to **myVM**. Replace the example name with your own VM name.

```
$vmName = "myVM"  
$vm = Get-AzureVM -ServiceName $serviceName -Name $vmName  
$vm.VM.MigrationState
```

Check the configuration for the prepared resources by using either PowerShell or the Azure portal. If you are not ready for migration and you want to go back to the old state, use the following command:

```
Move-AzureService -Abort -ServiceName $serviceName -DeploymentName $deploymentName
```

If the prepared configuration looks good, you can move forward and commit the resources by using the following command:

```
Move-AzureService -Commit -ServiceName $serviceName -DeploymentName $deploymentName
```

Step 6.1: Option 2 - Migrate virtual machines in a virtual network

To migrate virtual machines in a virtual network, you migrate the virtual network. The virtual machines automatically migrate with the virtual network. Pick the virtual network that you want to migrate.

NOTE

Migrate [single classic virtual machine](#) by creating a new Resource Manager virtual machine with Managed Disks using the VHD (OS and data) files of the virtual machine.

NOTE

The virtual network name might be different from what is shown in the new Portal. The new Azure Portal displays the name as `[vnet-name]` but the actual virtual network name is of type `Group [resource-group-name] [vnet-name]`. Before migrating, lookup the actual virtual network name using the command `Get-AzureVnetSite | Select -Property Name` or view it in the old Azure Portal.

This example sets the virtual network name to **myVnet**. Replace the example virtual network name with your own.

```
$vnetName = "myVnet"
```

NOTE

If the virtual network contains web or worker roles, or VMs with unsupported configurations, you get a validation error message.

First, validate if you can migrate the virtual network by using the following command:

```
Move-AzureVirtualNetwork -Validate -VirtualNetworkName $vnetName
```

The preceding command displays any warnings and errors that block migration. If validation is successful, then you can proceed with the following Prepare step:

```
Move-AzureVirtualNetwork -Prepare -VirtualNetworkName $vnetName
```

Check the configuration for the prepared virtual machines by using either Azure PowerShell or the Azure portal. If you are not ready for migration and you want to go back to the old state, use the following command:

```
Move-AzureVirtualNetwork -Abort -VirtualNetworkName $vnetName
```

If the prepared configuration looks good, you can move forward and commit the resources by using the following command:

```
Move-AzureVirtualNetwork -Commit -VirtualNetworkName $vnetName
```

Step 6.2 Migrate a storage account

Once you're done migrating the virtual machines, we recommend you migrate the storage accounts.

Before you migrate the storage account, please perform preceding prerequisite checks:

- **Migrate classic virtual machines whose disks are stored in the storage account**

Preceding command returns RoleName and DiskName properties of all the classic VM disks in the storage account. RoleName is the name of the virtual machine to which a disk is attached. If preceding command returns disks then ensure that virtual machines to which these disks are attached are migrated before migrating the storage account.

```
$storageAccountName = 'yourStorageAccountName'  
Get-AzureDisk | where-Object {$_.MediaLink.Host.Contains($storageAccountName)} | Select-Object -  
ExpandProperty AttachedTo -Property `'  
DiskName | Format-List -Property RoleName, DiskName
```

- **Delete unattached classic VM disks stored in the storage account**

Find unattached classic VM disks in the storage account using following command:

```
$storageAccountName = 'yourStorageAccountName'  
Get-AzureDisk | where-Object {$_._MediaLink.Host.Contains($storageAccountName)} | Where-Object -  
Property AttachedTo -EQ $null | Format-List -Property DiskName
```

If above command returns disks then delete these disks using following command:

```
Remove-AzureDisk -DiskName 'yourDiskName'
```

- **Delete VM images stored in the storage account**

Preceding command returns all the VM images with OS disk stored in the storage account.

```
Get-AzureVmImage | Where-Object { $_._OSDiskConfiguration.MediaLink -ne $null -and  
$_._OSDiskConfiguration.MediaLink.Host.Contains($storageAccountName)`  
} | Select-Object -Property ImageName, ImageLabel
```

Preceding command returns all the VM images with data disks stored in the storage account.

```
Get-AzureVmImage | Where-Object {$_._DataDiskConfigurations -ne $null `  
-and ($_.DataDiskConfigurations | Where-Object {$_._MediaLink -ne  
$null -and $_._MediaLink.Host.Contains($storageAccountName)}).Count -gt 0 `  
} | Select-Object -Property ImageName, ImageLabel
```

Delete all the VM images returned by above commands using preceding command:

```
Remove-AzureVMImage -ImageName 'yourImageName'
```

Validate each storage account for migration by using the following command. In this example, the storage account name is **myStorageAccount**. Replace the example name with the name of your own storage account.

```
$storageAccountName = "myStorageAccount"  
Move-AzureStorageAccount -Validate -StorageAccountName $storageAccountName
```

Next step is to Prepare the storage account for migration

```
$storageAccountName = "myStorageAccount"  
Move-AzureStorageAccount -Prepare -StorageAccountName $storageAccountName
```

Check the configuration for the prepared storage account by using either Azure PowerShell or the Azure portal. If you are not ready for migration and you want to go back to the old state, use the following command:

```
Move-AzureStorageAccount -Abort -StorageAccountName $storageAccountName
```

If the prepared configuration looks good, you can move forward and commit the resources by using the following command:

```
Move-AzureStorageAccount -Commit -StorageAccountName $storageAccountName
```

Next steps

- Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager
- Technical deep dive on platform-supported migration from classic to Azure Resource Manager
- Planning for migration of IaaS resources from classic to Azure Resource Manager
- Use CLI to migrate IaaS resources from classic to Azure Resource Manager
- Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager
- Review most common migration errors
- Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager

Common errors during Classic to Azure Resource Manager migration

6/27/2017 • 8 min to read • [Edit Online](#)

This article catalogs the most common errors and mitigations during the migration of IaaS resources from Azure classic deployment model to the Azure Resource Manager stack.

List of errors

ERROR STRING	MITIGATION
Internal server error	<p>In some cases, this is a transient error that goes away with a retry. If it continues to persist, contact Azure support as it needs investigation of platform logs.</p> <p>NOTE: Once the incident is tracked by the support team, please do not attempt any self-mitigation as this might have unintended consequences on your environment.</p>
Migration is not supported for Deployment {deployment-name} in HostedService {hosted-service-name} because it is a PaaS deployment (Web/Worker).	<p>This happens when a deployment contains a web/worker role. Since migration is only supported for Virtual Machines, please remove the web/worker role from the deployment and try migration again.</p>
Template {template-name} deployment failed. CorrelationId={guid}	<p>In the backend of migration service, we use Azure Resource Manager templates to create resources in the Azure Resource Manager stack. Since templates are idempotent, usually you can safely retry the migration operation to get past this error. If this error continues to persist, please contact Azure support and give them the CorrelationId.</p> <p>NOTE: Once the incident is tracked by the support team, please do not attempt any self-mitigation as this might have unintended consequences on your environment.</p>
The virtual network {virtual-network-name} does not exist.	<p>This can happen if you created the Virtual Network in the new Azure portal. The actual Virtual Network name follows the pattern "Group * "</p>
VM {vm-name} in HostedService {hosted-service-name} contains Extension {extension-name} which is not supported in Azure Resource Manager. It is recommended to uninstall it from the VM before continuing with migration.	<p>XML extensions such as BGInfo 1.* are not supported in Azure Resource Manager. Therefore, these extensions cannot be migrated. If these extensions are left installed on the virtual machine, they are automatically uninstalled before completing the migration.</p>
VM {vm-name} in HostedService {hosted-service-name} contains Extension VMSnapshot/VMSnapshotLinux, which is currently not supported for Migration. Uninstall it from the VM and add it back using Azure Resource Manager after the Migration is Complete	<p>This is the scenario where the virtual machine is configured for Azure Backup. Since this is currently an unsupported scenario, please follow the workaround at https://aka.ms/vmbakcupmigration</p>

ERROR STRING	MITIGATION
<p>VM {vm-name} in HostedService {hosted-service-name} contains Extension {extension-name} whose Status is not being reported from the VM. Hence, this VM cannot be migrated. Ensure that the Extension status is being reported or uninstall the extension from the VM and retry migration.</p> <p>VM {vm-name} in HostedService {hosted-service-name} contains Extension {extension-name} reporting Handler Status: {handler-status}. Hence, the VM cannot be migrated. Ensure that the Extension handler status being reported is {handler-status} or uninstall it from the VM and retry migration.</p>	<p>Azure guest agent & VM Extensions need outbound internet access to the VM storage account to populate their status. Common causes of status failure include</p> <ul style="list-style-type: none"> • a Network Security Group that blocks outbound access to the internet • If the VNET has on-prem DNS servers and DNS connectivity is lost <p>If you continue to see an unsupported status, you can uninstall the extensions to skip this check and move forward with migration.</p>
<p>VM Agent for VM {vm-name} in HostedService {hosted-service-name} is reporting the overall agent status as Not Ready. Hence, the VM may not be migrated, if it has a migratable extension. Ensure that the VM Agent is reporting overall agent status as Ready. Refer to https://aka.ms/classicasmigrationfaqs.</p>	
<p>Migration is not supported for Deployment {deployment-name} in HostedService {hosted-service-name} because it has multiple Availability Sets.</p>	<p>Currently, only hosted services that have 1 or less Availability sets can be migrated. To work around this problem, please move the additional Availability sets and Virtual machines in those Availability sets to a different hosted service.</p>
<p>Migration is not supported for Deployment {deployment-name} in HostedService {hosted-service-name} because it has VMs that are not part of the Availability Set even though the HostedService contains one.</p>	<p>The workaround for this scenario is to either move all the virtual machines into a single Availability set or remove all Virtual machines from the Availability set in the hosted service.</p>
<p>Storage account/HostedService/Virtual Network {virtual-network-name} is in the process of being migrated and hence cannot be changed</p>	<p>This error happens when the "Prepare" migration operation has been completed on the resource and an operation that would make a change to the resource is triggered. Because of the lock on the management plane after "Prepare" operation, any changes to the resource are blocked. To unlock the management plane, you can run the "Commit" migration operation to complete migration or the "Abort" migration operation to roll back the "Prepare" operation.</p>
<p>Migration is not allowed for HostedService {hosted-service-name} because it has VM {vm-name} in State: RoleStateUnknown. Migration is allowed only when the VM is in one of the following states - Running, Stopped, Stopped Deallocated.</p>	<p>The VM might be undergoing through a state transition, which usually happens when during an update operation on the HostedService such as a reboot, extension installation etc. It is recommended for the update operation to complete on the HostedService before trying migration.</p>
<p>Deployment {deployment-name} in HostedService {hosted-service-name} contains a VM {vm-name} with Data Disk {data-disk-name} whose physical blob size {size-of-the-vhd-blob-backing-the-data-disk} bytes does not match the VM Data Disk logical size {size-of-the-data-disk-specified-in-the-vm-api} bytes. Migration will proceed without specifying a size for the data disk for the Azure Resource Manager VM.</p>	<p>This error happens if you've resized the VHD blob without updating the size in the VM API model. Detailed mitigation steps are outlined below.</p>
<p>A storage exception occurred while validating data disk {data disk name} with media link {data disk Uri} for VM {VM name} in Cloud Service {Cloud Service name}. Please ensure that the VHD media link is accessible for this virtual machine</p>	<p>This error can happen if the disks of the VM have been deleted or are not accessible anymore. Please make sure the disks for the VM exist.</p>

ERROR STRING	MITIGATION
VM {vm-name} in HostedService {cloud-service-name} contains Disk with MediaLink {vhd-uri} which has blob name {vhd-blob-name} that is not supported in Azure Resource Manager.	This error occurs when the name of the blob has a "/" in it which is not supported in Compute Resource Provider currently.
Migration is not allowed for Deployment {deployment-name} in HostedService {cloud-service-name} as it is not in the regional scope. Please refer to http://aka.ms/regionscope for moving this deployment to regional scope.	In 2014, Azure announced that networking resources will move from a cluster level scope to regional scope. See [http://aka.ms/regionscope] for more details (http://aka.ms/regionscope). This error happens when the deployment being migrated has not had an update operation, which automatically moves it to a regional scope. Best workaround is to either add an endpoint to a VM or a data disk to the VM and then retry migration. See How to set up endpoints on a classic Windows virtual machine in Azure or Attach a data disk to a Windows virtual machine created with the classic deployment model

Detailed mitigations

VM with Data Disk whose physical blob size bytes does not match the VM Data Disk logical size bytes.

This happens when the Data disk logical size can get out of sync with the actual VHD blob size. This can be easily verified using the following commands:

Verifying the issue

```
# Store the VM details in the VM object
$vm = Get-AzureVM -ServiceName $servicename -Name $vmname

# Display the data disk properties
# NOTE the data disk LogicalDiskSizeInGB below which is 11GB. Also note the MediaLink Uri of the VHD blob as
we'll use this in the next step
$vm.VM.DataVirtualHardDisks

HostCaching      : None
DiskLabel        :
DiskName         : coreosvm-coreosvm-0-201611230636240687
Lun              : 0
LogicalDiskSizeInGB : 11
MediaLink        : https://contosostorage.blob.core.windows.net/vhds/coreosvm-dd1.vhd
SourceMediaLink   :
IOType           : Standard
ExtensionData    :

# Now get the properties of the blob backing the data disk above
# NOTE the size of the blob is about 15 GB which is different from LogicalDiskSizeInGB above
blob = Get-AzureStorageblob -Blob "coreosvm-dd1.vhd" -Container vhds

$blob

ICloudBlob       : Microsoft.WindowsAzure.Storage.Blob.CloudPageBlob
BlobType         : PageBlob
Length           : 16106127872
ContentType      : application/octet-stream
LastModified     : 11/23/2016 7:16:22 AM +00:00
SnapshotTime     :
ContinuationToken :
Context          : Microsoft.WindowsAzure.Commands.Common.Storage.AzureStorageContext
Name             : coreosvm-dd1.vhd
```

Mitigating the issue

```
# Convert the blob size in bytes to GB into a variable which we'll use later
$newSize = [int]($blob.Length / 1GB)

# See the calculated size in GB
$newSize

15

# Store the disk name of the data disk as we'll use this to identify the disk to be updated
$diskName = $vm.VM.DataVirtualHardDisks[0].DiskName

# Identify the LUN of the data disk to remove
$lunToRemove = $vm.VM.DataVirtualHardDisks[0].Lun

# Now remove the data disk from the VM so that the disk isn't leased by the VM and it's size can be updated
Remove-AzureDataDisk -LUN $lunToRemove -VM $vm | Update-AzureVm -Name $vmname -ServiceName $servicename



| OperationDescription | OperationId                        | OperationStatus |
|----------------------|------------------------------------|-----------------|
| Update-AzureVM       | 213xx1-b44b-1v6n-23gg-591f2a13cd16 | Succeeded       |



# Verify we have the right disk that's going to be updated
Get-AzureDisk -DiskName $diskName

AffinityGroup      :
AttachedTo        :
IsCorrupted       : False
Label             :
Location          : East US
DiskSizeInGB      : 11
MediaLink         : https://contosostorage.blob.core.windows.net/vhds/coreosvm-dd1.vhd
DiskName          : coreosvm-coreosvm-0-201611230636240687
SourceImageName   :
OS                :
IOType            : Standard
OperationDescription : Get-AzureDisk
OperationId       : 0c56a2b7-a325-123b-7043-74c27d5a61fd
OperationStatus    : Succeeded

# Now update the disk to the new size
Update-AzureDisk -DiskName $diskName -ResizedSizeInGB $newSize -Label $diskName



| OperationDescription | OperationId                          | OperationStatus |
|----------------------|--------------------------------------|-----------------|
| Update-AzureDisk     | cv134b65-1b6n-8908-abuo-ce9e395ac3e7 | Succeeded       |



# Now verify that the "DiskSizeInGB" property of the disk matches the size of the blob
Get-AzureDisk -DiskName $diskName

AffinityGroup      :
AttachedTo        :
IsCorrupted       : False
Label             : coreosvm-coreosvm-0-201611230636240687
Location          : East US
DiskSizeInGB      : 15
MediaLink         : https://contosostorage.blob.core.windows.net/vhds/coreosvm-dd1.vhd
DiskName          : coreosvm-coreosvm-0-201611230636240687
SourceImageName   :
OS                :
IOType            : Standard
OperationDescription : Get-AzureDisk
OperationId       : 1v53bde5-cv56-5621-9078-16b9c8a0bad2
OperationStatus    : Succeeded

# Now we'll add the disk back to the VM as a data disk. First we need to get an updated VM object
```

```
$vm = Get-AzureVM -ServiceName $servicename -Name $vmname

Add-AzureDataDisk -Import -DiskName $diskName -LUN 0 -VM $vm -HostCaching ReadWrite | Update-AzureVm -Name
$vmname -ServiceName $servicename

OperationDescription OperationId OperationStatus
-----
Update-AzureVM      b0ad3d4c-4v68-45vb-xxc1-134fd010d0f8 Succeeded
```

Moving a VM to a different subscription after completing migration

After you complete the migration process, you may want to move the VM to another subscription. However, if you have a secret/certificate on the VM that references a Key Vault resource, the move is currently not supported. The below instructions will allow you to workaround the issue.

PowerShell

```
$vm = Get-AzureRmVM -ResourceGroupName "MyRG" -Name "MyVM"
Remove-AzureRmVMSecret -VM $vm
Update-AzureRmVM -ResourceGroupName "MyRG" -VM $vm
```

Azure CLI 2.0

```
az vm update -g "myrg" -n "myvm" --set osProfile.Secrets=[]
```

Next steps

- [Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager](#)
- [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#)
- [Planning for migration of IaaS resources from classic to Azure Resource Manager](#)
- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager](#)
- [Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager](#)

Community tools to migrate IaaS resources from classic to Azure Resource Manager

7/27/2017 • 1 min to read • [Edit Online](#)

This article catalogs the tools that have been provided by the community to assist with migration of IaaS resources from classic to the Azure Resource Manager deployment model.

NOTE

These tools are not officially supported by Microsoft Support. Therefore they are open sourced on GitHub and we're happy to accept PRs for fixes or additional scenarios. To report an issue, use the GitHub issues feature.

Migrating with these tools will cause downtime for your classic Virtual Machine. If you're looking for platform supported migration, visit

- [Platform supported migration of IaaS resources from Classic to Azure Resource Manager stack](#)
- [Technical Deep Dive on Platform supported migration from Classic to Azure Resource Manager](#)
- [Migrate IaaS resources from Classic to Azure Resource Manager using Azure PowerShell](#)

AsmMetadataParser

This is a collection of helper tools created as part of enterprise migrations from Azure Service Management to Azure Resource Manager. This tool allows you to replicate your infrastructure into another subscription which can be used for testing migration and iron out any issues before running the migration on your Production subscription.

[Link to the tool documentation](#)

migAz

migAz is an additional option to migrate a complete set of classic IaaS resources to Azure Resource Manager IaaS resources. The migration can occur within the same subscription or between different subscriptions and subscription types (ex: CSP subscriptions).

[Link to the tool documentation](#)

Next Steps

- [Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager](#)
- [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#)
- [Planning for migration of IaaS resources from classic to Azure Resource Manager](#)
- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Review most common migration errors](#)
- [Review the most frequently asked questions about migrating IaaS resources from classic to Azure Resource Manager](#)

Frequently asked questions about classic to Azure Resource Manager migration

6/27/2017 • 5 min to read • [Edit Online](#)

Does this migration plan affect any of my existing services or applications that run on Azure virtual machines?

No. The VMs (classic) are fully supported services in general availability. You can continue to use these resources to expand your footprint on Microsoft Azure.

What happens to my VMs if I don't plan on migrating in the near future?

We are not deprecating the existing classic APIs and resource model. We want to make migration easy, considering the advanced features that are available in the Resource Manager deployment model. We highly recommend that you review [some of the advancements](#) that are part of IaaS under Resource Manager.

What does this migration plan mean for my existing tooling?

Updating your tooling to the Resource Manager deployment model is one of the most important changes that you have to account for in your migration plans.

How long will the management-plane downtime be?

It depends on the number of resources that are being migrated. For smaller deployments (a few tens of VMs), the whole migration should take less than an hour. For large-scale deployments (hundreds of VMs), the migration can take a few hours.

Can I roll back after my migrating resources are committed in Resource Manager?

You can abort your migration as long as the resources are in the prepared state. Rollback is not supported after the resources have been successfully migrated through the commit operation.

Can I roll back my migration if the commit operation fails?

You cannot abort migration if the commit operation fails. All migration operations, including the commit operation, are idempotent. So we recommend that you retry the operation after a short time. If you still face an error, create a support ticket or create a forum post with the [ClassicIaaSMigration](#) tag on our [VM forum](#).

Do I have to buy another express route circuit if I have to use IaaS under Resource Manager?

No. We recently enabled [moving ExpressRoute circuits from the classic to the Resource Manager deployment model](#). You don't have to buy a new ExpressRoute circuit if you already have one.

What if I had configured Role-Based Access Control policies for my

classic IaaS resources?

During migration, the resources transform from classic to Resource Manager. So we recommend that you plan the RBAC policy updates that need to happen after migration.

I backed up my classic VMs in a Backup vault. Can I migrate my VMs from classic mode to Resource Manager mode and protect them in a Recovery Services vault?

VM recovery points in a backup vault don't automatically migrate to a Recovery Services vault when you move the VM from classic to Resource Manager mode. Follow these steps to transfer your VM backups:

1. In the Backup vault, go to the **Protected Items** tab and select the VM. Click [Stop Protection](#). Leave *Delete associated backup data* option **unchecked**.
2. Delete the backup/snapshot extension from the VM.
3. Migrate the virtual machine from classic mode to Resource Manager mode. Make sure the storage and network information corresponding to the virtual machine is also migrated to Resource Manager mode.
4. Create a Recovery Services vault and configure backup on the migrated virtual machine using **Backup** action on top of vault dashboard. For detailed information on backing up a VM to a Recovery Services vault, see the article, [Protect Azure VMs with a Recovery Services vault](#).

Can I validate my subscription or resources to see if they're capable of migration?

Yes. In the platform-supported migration option, the first step in preparing for migration is to validate that the resources are capable of migration. In case the validate operation fails, you receive messages for all the reasons the migration cannot be completed.

What happens if I run into a quota error while preparing the IaaS resources for migration?

We recommend that you abort your migration and then log a support request to increase the quotas in the region where you are migrating the VMs. After the quota request is approved, you can start executing the migration steps again.

How do I report an issue?

Post your issues and questions about migration to our [VM forum](#), with the keyword `ClassicIaaSMigration`. We recommend posting all your questions on this forum. If you have a support contract, you're welcome to log a support ticket as well.

What if I don't like the names of the resources that the platform chose during migration?

All the resources that you explicitly provide names for in the classic deployment model are retained during migration. In some cases, new resources are created. For example: a network interface is created for every VM. We currently don't support the ability to control the names of these new resources created during migration. Log your votes for this feature on the [Azure feedback forum](#).

Can I migrate ExpressRoute circuits used across subscriptions with authorization links?

ExpressRoute circuits which use cross-subscription authorization links cannot be migrated automatically without downtime. We have guidance on how these can be migrated using manual steps. See [Migrate ExpressRoute circuits and associated virtual networks from the classic to the Resource Manager deployment model](#) for steps and more information.

I got the message "VM is reporting the overall agent status as Not Ready. Hence, the VM cannot be migrated. Ensure that the VM Agent is reporting overall agent status as Ready" or "VM contains Extension whose Status is not being reported from the VM. Hence, this VM cannot be migrated."

This message is received when the VM does not have outbound connectivity to the internet. The VM agent uses outbound connectivity to reach the Azure storage account for updating the agent status every five minutes.

Next steps

- [Overview of platform-supported migration of IaaS resources from classic to Azure Resource Manager](#)
- [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#)
- [Planning for migration of IaaS resources from classic to Azure Resource Manager](#)
- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Community tools for assisting with migration of IaaS resources from classic to Azure Resource Manager](#)
- [Review most common migration errors](#)

Troubleshoot Remote Desktop connections to an Azure virtual machine

11/3/2017 • 9 min to read • [Edit Online](#)

The Remote Desktop Protocol (RDP) connection to your Windows-based Azure virtual machine (VM) can fail for various reasons, leaving you unable to access your VM. The issue can be with the Remote Desktop service on the VM, the network connection, or the Remote Desktop client on your host computer. This article guides you through some of the most common methods to resolve RDP connection issues.

If you need more help at any point in this article, you can contact the Azure experts on the [MSDN Azure and Stack Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select **Get Support**.

Quick troubleshooting steps

After each troubleshooting step, try reconnecting to the VM:

1. Reset Remote Desktop configuration.
2. Check Network Security Group rules / Cloud Services endpoints.
3. Review VM console logs.
4. Reset the NIC for the VM.
5. Check the VM Resource Health.
6. Reset your VM password.
7. Restart your VM.
8. Redeploy your VM.

Continue reading if you need more detailed steps and explanations. Verify that local network equipment such as routers and firewalls are not blocking outbound TCP port 3389, as noted in [detailed RDP troubleshooting scenarios](#).

TIP

If the **Connect** button for your VM is grayed out in the portal and you are not connected to Azure via an [Express Route](#) or [Site-to-Site VPN](#) connection, you need to create and assign your VM a public IP address before you can use RDP. You can read more about [public IP addresses in Azure](#).

Ways to troubleshoot RDP issues

You can troubleshoot VMs created using the Resource Manager deployment model by using one of the following methods:

- [Azure portal](#) - great if you need to quickly reset the RDP configuration or user credentials and you don't have the Azure tools installed.
- [Azure PowerShell](#) - if you are comfortable with a PowerShell prompt, quickly reset the RDP configuration or user credentials using the Azure PowerShell cmdlets.

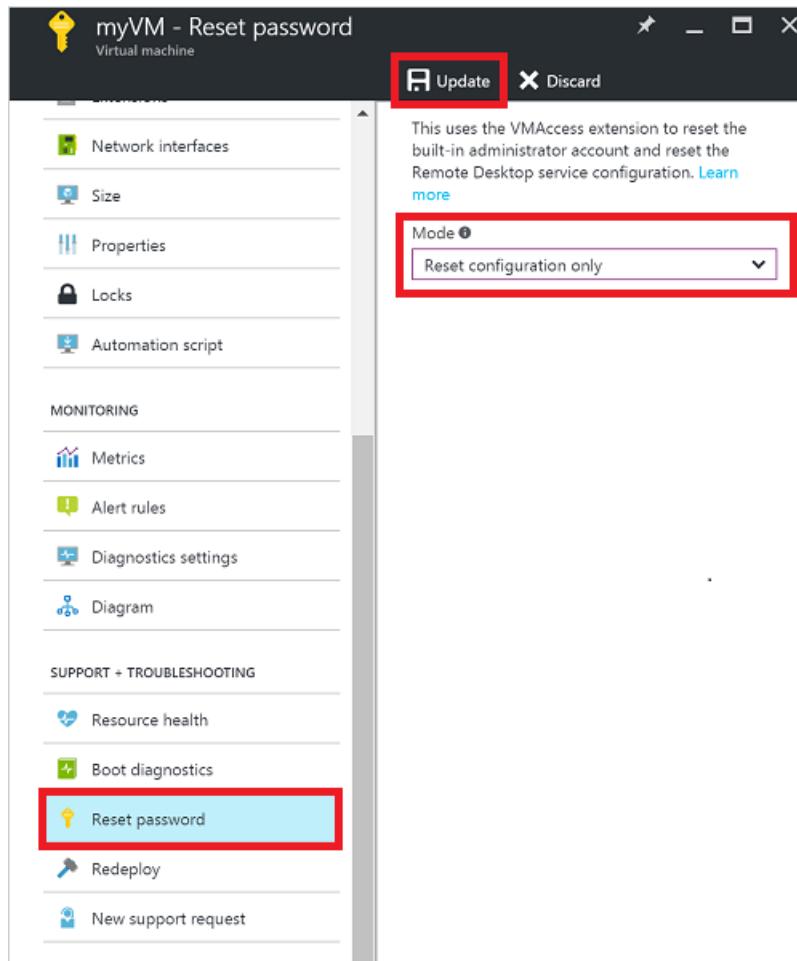
You can also find steps on troubleshooting VMs created using the [Classic deployment model](#).

Troubleshoot using the Azure portal

After each troubleshooting step, try connecting to your VM again. If you still cannot connect, try the next step.

1. **Reset your RDP connection.** This troubleshooting step resets the RDP configuration when Remote Connections are disabled or Windows Firewall rules are blocking RDP, for example.

Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Reset password** button. Set the **Mode** to **Reset configuration only** and then click the **Update** button:



2. **Verify Network Security Group rules.** Use [IP flow verify](#) to confirm if a rule in a Network Security Group is blocking traffic to or from a virtual machine. You can also review effective security group rules to ensure inbound "Allow" NSG rule exists and is prioritized for RDP port(default 3389). For more information, see [Using Effective Security Rules to troubleshoot VM traffic flow](#).

3. **Review VM boot diagnostics.** This troubleshooting step reviews the VM console logs to determine if the VM is reporting an issue. Not all VMs have boot diagnostics enabled, so this troubleshooting step may be optional.

Specific troubleshooting steps are beyond the scope of this article, but may indicate a wider problem that is affecting RDP connectivity. For more information on reviewing the console logs and VM screenshot, see [Boot Diagnostics for VMs](#).

4. **Reset the NIC for the VM.** For more information, see [how to reset NIC for Azure Windows VM](#).

5. **Check the VM Resource Health.** This troubleshooting step verifies there are no known issues with the Azure platform that may impact connectivity to the VM.

Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Resource health** button. A healthy VM reports as being

Available:

myVM - Resource health

Virtual machine

Refresh

Resource health watches your resource and tells you if it's running as expected. [Learn more](#)

Available Last updated: 10/14/2016, 3:07:00 PM

There aren't any known Azure platform problems affecting this virtual machine

[View History](#)

What actions can you take?

1. If you're having problems, use the [Troubleshoot tool](#) to get recommended solutions
2. If you are experiencing problems you believe are caused by Azure, [contact support](#)

SUPPORT + TROUBLESHOOTING

[Resource health](#) (highlighted with a red box)

[Boot diagnostics](#)

6. **Reset user credentials.** This troubleshooting step resets the password on a local administrator account when you are unsure or have forgotten the credentials.

Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Reset password** button. Make sure the **Mode** is set to **Reset password** and then enter your username and a new password. Finally, click the **Update** button:

myVM - Reset password

Virtual machine

Update Discard

This uses the VMAccess extension to reset the built-in administrator account and reset the Remote Desktop service configuration. [Learn more](#)

Mode [Reset password](#) (highlighted with a red box)

* User name [ops](#)

* Password [.....](#) ✓

* Confirm password [.....](#) ✓

SUPPORT + TROUBLESHOOTING

[Resource health](#)

[Boot diagnostics](#)

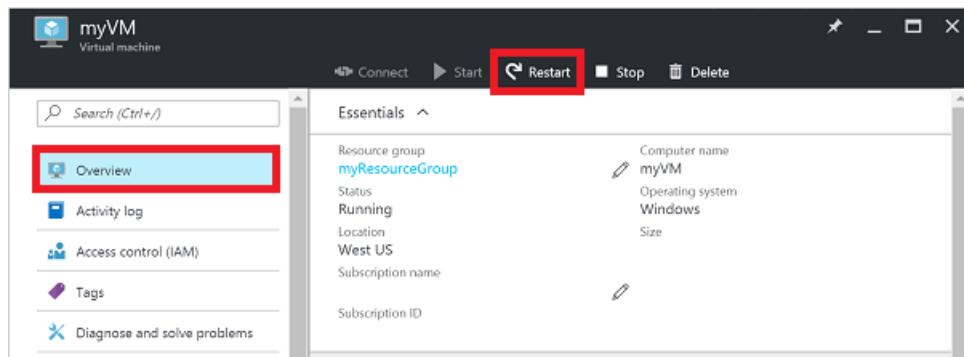
[Reset password](#) (highlighted with a red box)

[Redeploy](#)

[New support request](#)

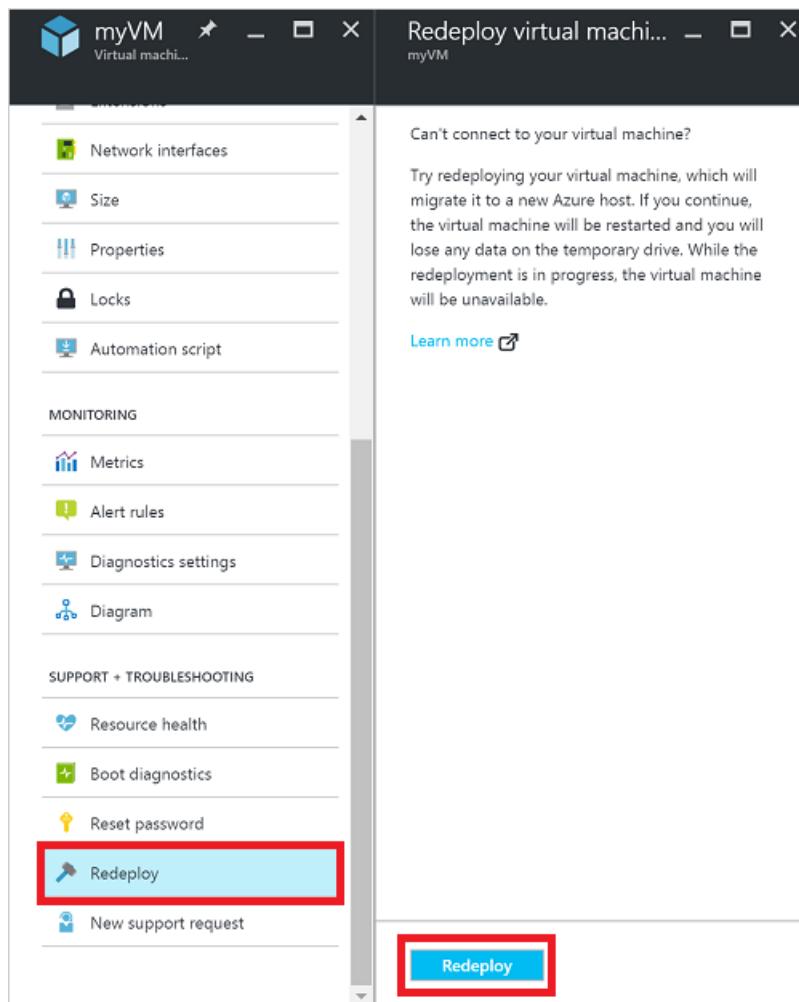
7. **Restart your VM.** This troubleshooting step can correct any underlying issues the VM itself is having.

Select your VM in the Azure portal and click the **Overview** tab. Click the **Restart** button:



8. **Redeploy your VM.** This troubleshooting step redeploys your VM to another host within Azure to correct any underlying platform or networking issues.

Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Redeploy** button, and then click **Redeploy**:



After this operation finishes, ephemeral disk data is lost and dynamic IP addresses that are associated with the VM are updated.

If you are still encountering RDP issues, you can [open a support request](#) or read [more detailed RDP troubleshooting concepts and steps](#).

Troubleshoot using Azure PowerShell

If you haven't already, [install and configure the latest Azure PowerShell](#).

The following examples use variables such as `myResourceGroup`, `myVM`, and `myVMAccessExtension`. Replace these variable names and locations with your own values.

NOTE

You reset the user credentials and the RDP configuration by using the [Set-AzureRmVMAccessExtension](#) PowerShell cmdlet. In the following examples, `myVMAccessExtension` is a name that you specify as part of the process. If you have previously worked with the VMAccessAgent, you can get the name of the existing extension by using `Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVM"` to check the properties of the VM. To view the name, look under the 'Extensions' section of the output.

After each troubleshooting step, try connecting to your VM again. If you still cannot connect, try the next step.

1. **Reset your RDP connection.** This troubleshooting step resets the RDP configuration when Remote Connections are disabled or Windows Firewall rules are blocking RDP, for example.

The following example resets the RDP connection on a VM named `myVM` in the `Westus` location and in the resource group named `myResourceGroup`:

```
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" `  
    -VMName "myVM" -Location Westus -Name "myVMAccessExtension"
```

2. **Verify Network Security Group rules.** This troubleshooting step verifies that you have a rule in your Network Security Group to permit RDP traffic. The default port for RDP is TCP port 3389. A rule to permit RDP traffic may not be created automatically when you create your VM.

First, assign all the configuration data for your Network Security Group to the `$rules` variable. The following example obtains information about the Network Security Group named `myNetworkSecurityGroup` in the resource group named `myResourceGroup`:

```
$rules = Get-AzureRmNetworkSecurityGroup -ResourceGroupName "myResourceGroup" `  
    -Name "myNetworkSecurityGroup"
```

Now, view the rules that are configured for this Network Security Group. Verify that a rule exists to allow TCP port 3389 for inbound connections as follows:

```
$rules.SecurityRules
```

The following example shows a valid security rule that permits RDP traffic. You can see `Protocol`, `DestinationPortRange`, `Access`, and `Direction` are configured correctly:

```
Name          : default-allow-rdp  
Id           :  
/subscriptions/guid/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkSecurityGroups/m  
yNetworkSecurityGroup/securityRules/default-allow-rdp  
Etag         :  
ProvisioningState : Succeeded  
Description   :  
Protocol      : TCP  
SourcePortRange : *  
DestinationPortRange : 3389  
SourceAddressPrefix : *  
DestinationAddressPrefix : *  
Access        : Allow  
Priority      : 1000  
Direction     : Inbound
```

If you do not have a rule that allows RDP traffic, [create a Network Security Group rule](#). Allow TCP port

3. **Reset user credentials.** This troubleshooting step resets the password on the local administrator account that you specify when you are unsure of, or have forgotten, the credentials.

First, specify the username and a new password by assigning credentials to the `$cred` variable as follows:

```
$cred=Get-Credential
```

Now, update the credentials on your VM. The following example updates the credentials on a VM named `myVM` in the `WestUS` location and in the resource group named `myResourceGroup`:

```
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" `  
-VMName "myVM" -Location WestUS -Name "myVMAccessExtension" `  
-UserName $cred.GetNetworkCredential().Username `  
-Password $cred.GetNetworkCredential().Password
```

4. **Restart your VM.** This troubleshooting step can correct any underlying issues the VM itself is having.

The following example restarts the VM named `myVM` in the resource group named `myResourceGroup`:

```
Restart-AzureRmVM -ResourceGroup "myResourceGroup" -Name "myVM"
```

5. **Redeploy your VM.** This troubleshooting step redeploys your VM to another host within Azure to correct any underlying platform or networking issues.

The following example redeploys the VM named `myVM` in the `WestUS` location and in the resource group named `myResourceGroup`:

```
Set-AzureRmVM -Redeploy -ResourceGroupName "myResourceGroup" -Name "myVM"
```

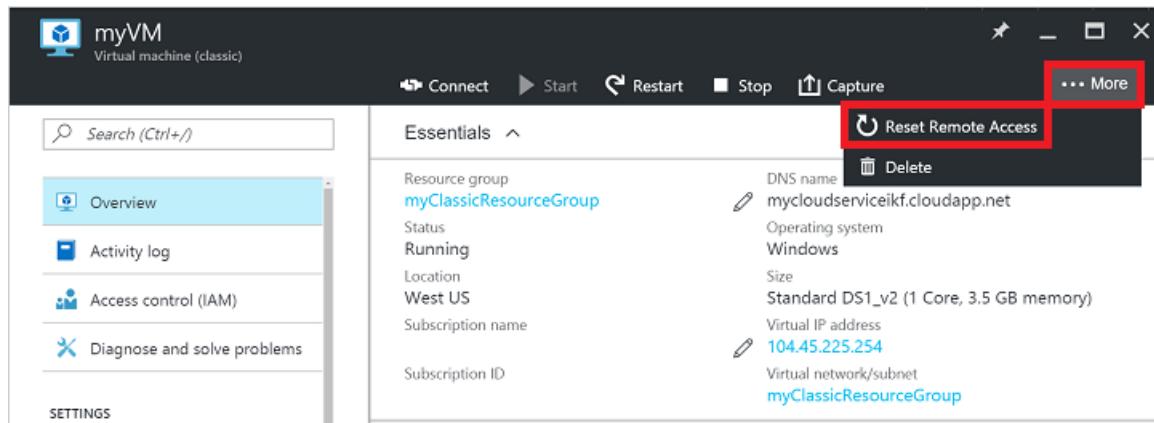
If you are still encountering RDP issues, you can [open a support request](#) or read [more detailed RDP troubleshooting concepts and steps](#).

Troubleshoot VMs created using the Classic deployment model

After each troubleshooting step, try reconnecting to the VM.

1. **Reset your RDP connection.** This troubleshooting step resets the RDP configuration when Remote Connections are disabled or Windows Firewall rules are blocking RDP, for example.

Select your VM in the Azure portal. Click the **...More** button, then click **Reset Remote Access**:



2. **Verify Cloud Services endpoints.** This troubleshooting step verifies that you have endpoints in your Cloud Services to permit RDP traffic. The default port for RDP is TCP port 3389. A rule to permit RDP traffic may not be created automatically when you create your VM.

Select your VM in the Azure portal. Click the **Endpoints** button to view the endpoints currently configured for your VM. Verify that endpoints exist that allow RDP traffic on TCP port 3389.

The following example shows valid endpoints that permit RDP traffic:

NAME	PROTOCOL	PUBLIC PORT	PRIVATE PO...	ACL RULES
LOAD-BALANCED				
No results				
STANDALONE				
RDP TCP	TCP	3389	3389	0
RDP UDP	UDP	3389	3389	0
WinRM	TCP	5986	5986	0

If you do not have an endpoint that allows RDP traffic, [create a Cloud Services endpoint](#). Allow TCP to private port 3389.

3. **Review VM boot diagnostics.** This troubleshooting step reviews the VM console logs to determine if the VM is reporting an issue. Not all VMs have boot diagnostics enabled, so this troubleshooting step may be optional.

Specific troubleshooting steps are beyond the scope of this article, but may indicate a wider problem that is affecting RDP connectivity. For more information on reviewing the console logs and VM screenshot, see [Boot Diagnostics for VMs](#).

4. **Check the VM Resource Health.** This troubleshooting step verifies there are no known issues with the Azure platform that may impact connectivity to the VM.

Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Resource Health** button. A healthy VM reports as being **Available**:

myVM - Resource health
Virtual machine (classic)

Search (Ctrl+ /)

Endpoints

Load balanced sets

Availability set

Extensions

Size

Properties

Locks

MONITORING

Diagnostics

Alert rules

SUPPORT + TROUBLESHOOTING

Resource health

Boot diagnostics

Available

Last updated: 10/17/2016, 11:11:00 AM ⓘ

There aren't any known Azure platform problems affecting this virtual machine

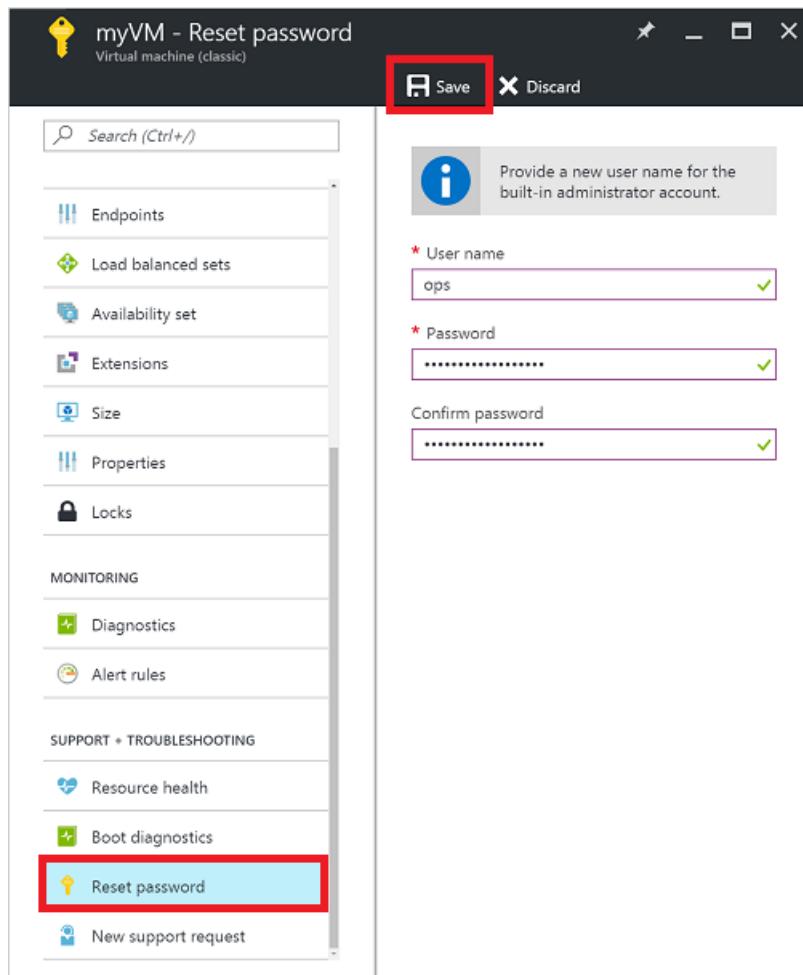
View History

What actions can you take?

1. If you're having problems, use the [Troubleshoot tool](#) to get recommended solutions
2. If you are experiencing problems you believe are caused by Azure, [contact support](#)

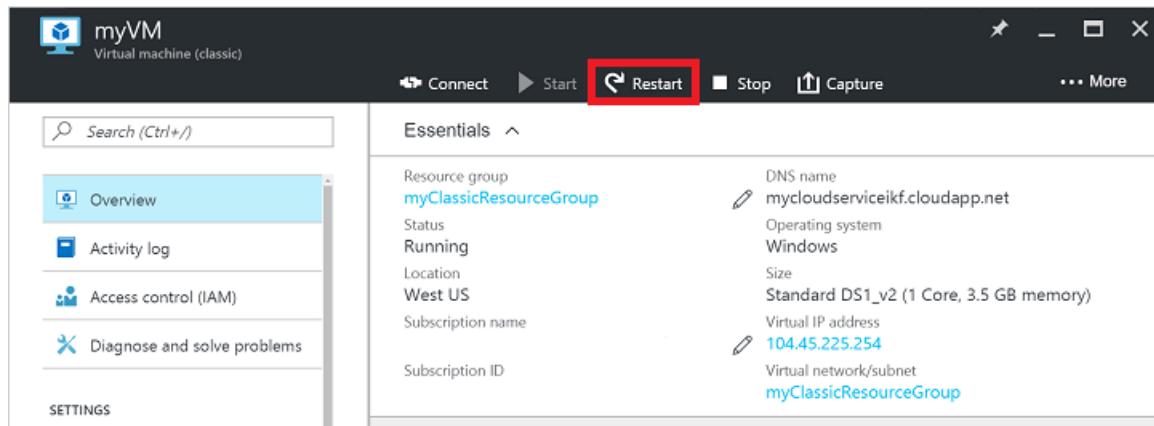
5. **Reset user credentials.** This troubleshooting step resets the password on the local administrator account that you specify when you are unsure or have forgotten the credentials.

Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Reset password** button. Enter your username and a new password. Finally, click the **Save** button:



6. **Restart your VM.** This troubleshooting step can correct any underlying issues the VM itself is having.

Select your VM in the Azure portal and click the **Overview** tab. Click the **Restart** button:



If you are still encountering RDP issues, you can [open a support request](#) or read [more detailed RDP troubleshooting concepts and steps](#).

Troubleshoot specific RDP errors

You may encounter a specific error message when trying to connect to your VM via RDP. The following are the most common error messages:

- [The remote session was disconnected because there are no Remote Desktop License Servers available to provide a license.](#)
- [Remote Desktop can't find the computer "name".](#)
- [An authentication error has occurred. The Local Security Authority cannot be contacted.](#)

- [Windows Security error: Your credentials did not work.](#)
- [This computer can't connect to the remote computer.](#)

Additional resources

If none of these errors occurred and you still can't connect to the VM via Remote Desktop, read the detailed [troubleshooting guide for Remote Desktop](#).

- For troubleshooting steps in accessing applications running on a VM, see [Troubleshoot access to an application running on an Azure VM](#).
- If you are having issues using Secure Shell (SSH) to connect to a Linux VM in Azure, see [Troubleshoot SSH connections to a Linux VM in Azure](#).

Detailed troubleshooting steps for remote desktop connection issues to Windows VMs in Azure

9/27/2017 • 8 min to read • [Edit Online](#)

This article provides detailed troubleshooting steps to diagnose and fix complex Remote Desktop errors for Windows-based Azure virtual machines.

IMPORTANT

To eliminate the more common Remote Desktop errors, make sure to read [the basic troubleshooting article for Remote Desktop](#) before proceeding.

You may encounter a Remote Desktop error message that does not resemble any of the specific error messages covered in [the basic Remote Desktop troubleshooting guide](#). Follow these steps to determine why the Remote Desktop (RDP) client is unable to connect to the RDP service on the Azure VM.

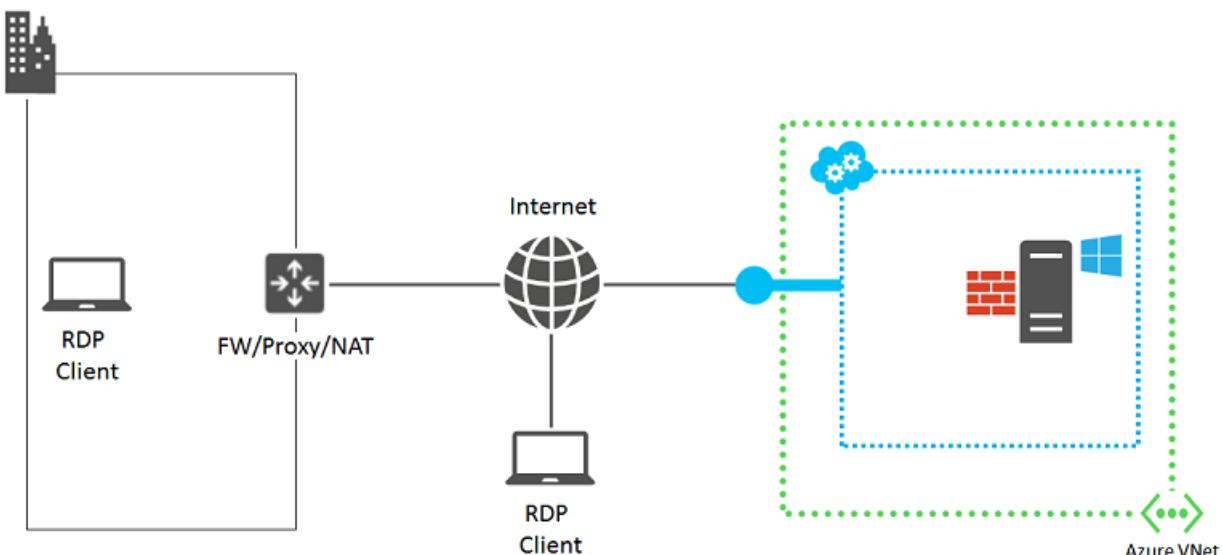
NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager](#) and [classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

If you need more help at any point in this article, you can contact the Azure experts on [the MSDN Azure and the Stack Overflow forums](#). Alternatively, you can also file an Azure support incident. Go to the [Azure Support site](#) and click **Get Support**. For information about using Azure Support, read the [Microsoft Azure Support FAQ](#).

Components of a Remote Desktop connection

The following components are involved in an RDP connection:



Before proceeding, it might help to mentally review what has changed since the last successful Remote Desktop connection to the VM. For example:

- The public IP address of the VM or the cloud service containing the VM (also called the virtual IP address [VIP](#))

has changed. The RDP failure could be because your DNS client cache still has the *old IP address* registered for the DNS name. Flush your DNS client cache and try connecting the VM again. Or try connecting directly with the new VIP.

- You are using a third-party application to manage your Remote Desktop connections instead of using the connection generated by the Azure portal. Verify that the application configuration includes the correct TCP port for the Remote Desktop traffic. You can check this port for a classic virtual machine in the [Azure portal](#), by clicking the VM's Settings > Endpoints.

Preliminary steps

Before proceeding to the detailed troubleshooting,

- Check the status of the virtual machine in the Azure portal for any obvious issues.
- Follow the [quick fix steps for common RDP errors in the basic troubleshooting guide](#).
- For custom images, make sure that your VHD is properly prepared prior to upload it. For more information, see [Prepare a Windows VHD or VHDX to upload to Azure](#).

Try reconnecting to the VM via Remote Desktop after these steps.

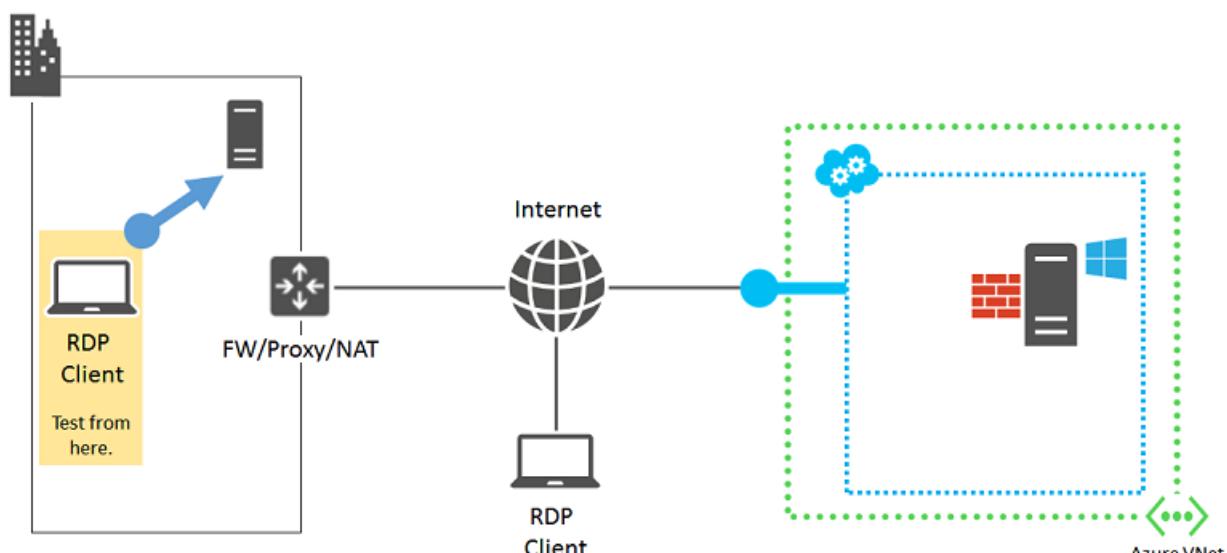
Detailed troubleshooting steps

The Remote Desktop client may not be able to reach the Remote Desktop service on the Azure VM due to issues at the following sources:

- [Remote Desktop client computer](#)
- [Organization intranet edge device](#)
- [Cloud service endpoint and access control list \(ACL\)](#)
- [Network security groups](#)
- [Windows-based Azure VM](#)

Source 1: Remote Desktop client computer

Verify that your computer can make Remote Desktop connections to another on-premises, Windows-based computer.



If you cannot, check for the following settings on your computer:

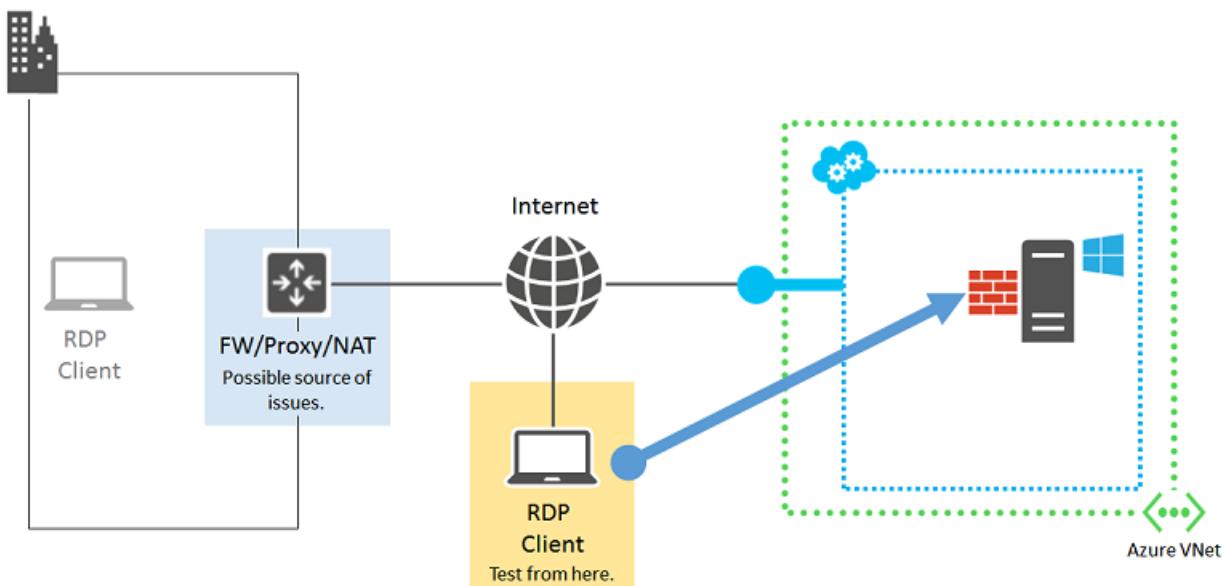
- A local firewall setting that is blocking Remote Desktop traffic.
- Locally installed client proxy software that is preventing Remote Desktop connections.

- Locally installed network monitoring software that is preventing Remote Desktop connections.
- Other types of security software that either monitor traffic or allow/disallow specific types of traffic that is preventing Remote Desktop connections.

In all these cases, temporarily disable the software and try to connect to an on-premises computer via Remote Desktop. If you can find out the actual cause this way, work with your network administrator to correct the software settings to allow Remote Desktop connections.

Source 2: Organization intranet edge device

Verify that a computer directly connected to the Internet can make Remote Desktop connections to your Azure virtual machine.



If you do not have a computer that is directly connected to the Internet, create and test with a new Azure virtual machine in a resource group or cloud service. For more information, see [Create a virtual machine running Windows in Azure](#). You can delete the virtual machine and the resource group or the cloud service, after the test.

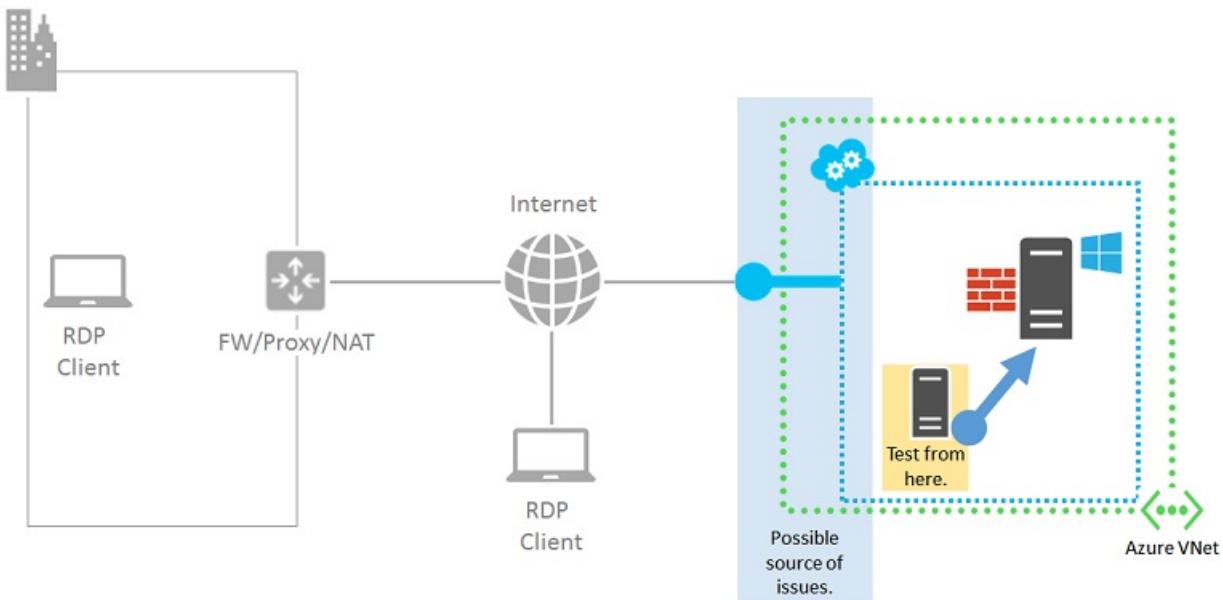
If you can create a Remote Desktop connection with a computer directly attached to the Internet, check your organization intranet edge device for:

- An internal firewall blocking HTTPS connections to the Internet.
- A proxy server preventing Remote Desktop connections.
- Intrusion detection or network monitoring software running on devices in your edge network that is preventing Remote Desktop connections.

Work with your network administrator to correct the settings of your organization intranet edge device to allow HTTPS-based Remote Desktop connections to the Internet.

Source 3: Cloud service endpoint and ACL

For VMs created using the Classic deployment model, verify that another Azure VM that is in the same cloud service or virtual network can make Remote Desktop connections to your Azure VM.



NOTE

For virtual machines created in Resource Manager, skip to [Source 4: Network Security Groups](#).

If you do not have another virtual machine in the same cloud service or virtual network, create one. Follow the steps in [Create a virtual machine running Windows in Azure](#). Delete the test virtual machine after the test is completed.

If you can connect via Remote Desktop to a virtual machine in the same cloud service or virtual network, check for these settings:

- The endpoint configuration for Remote Desktop traffic on the target VM: The private TCP port of the endpoint must match the TCP port on which the VM's Remote Desktop service is listening (default is 3389).
- The ACL for the Remote Desktop traffic endpoint on the target VM: ACLs allow you to specify allowed or denied incoming traffic from the Internet based on its source IP address. Misconfigured ACLs can prevent incoming Remote Desktop traffic to the endpoint. Check your ACLs to ensure that incoming traffic from your public IP addresses of your proxy or other edge server is allowed. For more information, see [What is a Network Access Control List \(ACL\)?](#)

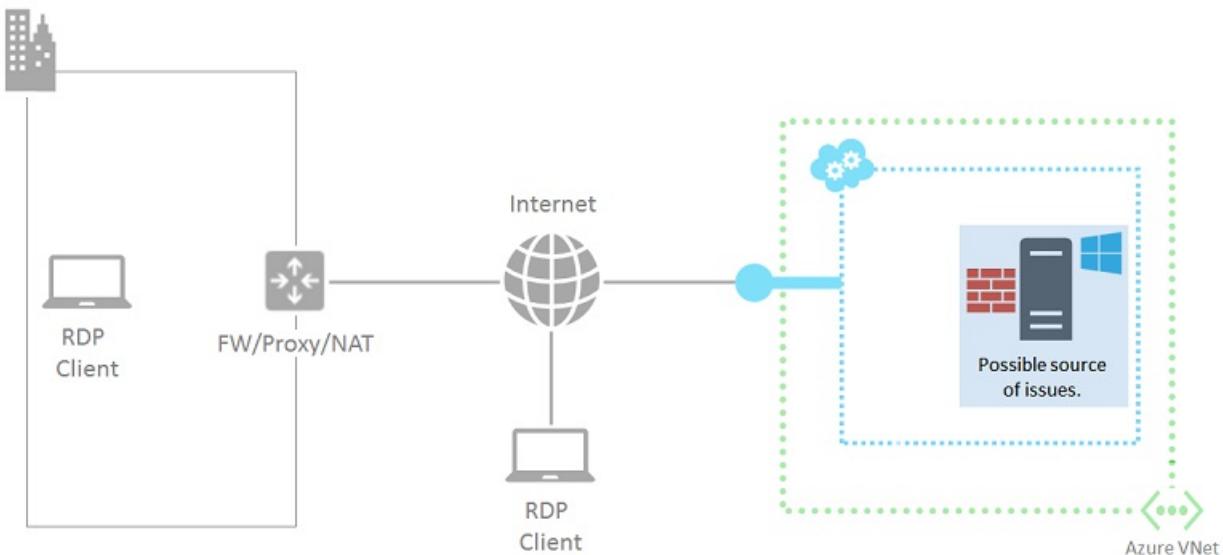
To check if the endpoint is the source of the problem, remove the current endpoint and create a new one, choosing a random port in the range 49152–65535 for the external port number. For more information, see [How to set up endpoints to a virtual machine](#).

Source 4: Network Security Groups

Network Security Groups allow more granular control of allowed inbound and outbound traffic. You can create rules spanning subnets and cloud services in an Azure virtual network.

Use [IP flow verify](#) to confirm if a rule in a Network Security Group is blocking traffic to or from a virtual machine. You can also review effective security group rules to ensure inbound "Allow" NSG rule exists and is prioritized for RDP port(default 3389). For more information, see [Using Effective Security Rules to troubleshoot VM traffic flow](#).

Source 5: Windows-based Azure VM



Follow the instructions in [this article](#). This article resets the Remote Desktop service on the virtual machine:

- Enable the "Remote Desktop" Windows Firewall default rule (TCP port 3389).
- Enable Remote Desktop connections by setting the HKLM\System\CurrentControlSet\Control\Terminal Server\fDenyTSConnections registry value to 0.

Try the connection from your computer again. If you are still not able to connect via Remote Desktop, check for the following possible problems:

- The Remote Desktop service is not running on the target VM.
- The Remote Desktop service is not listening on TCP port 3389.
- Windows Firewall or another local firewall has an outbound rule that is preventing Remote Desktop traffic.
- Intrusion detection or network monitoring software running on the Azure virtual machine is preventing Remote Desktop connections.

For VMs created using the classic deployment model, you can use a remote Azure PowerShell session to the Azure virtual machine. First, you need to install a certificate for the virtual machine's hosting cloud service. Go to [Configure Secure Remote PowerShell Access to Azure Virtual Machines](#) and download the **InstallWinRMCertAzureVM.ps1** script file to your local computer.

Next, install Azure PowerShell if you haven't already. See [How to install and configure Azure PowerShell](#).

Next, open an Azure PowerShell command prompt and change the current folder to the location of the **InstallWinRMCertAzureVM.ps1** script file. To run an Azure PowerShell script, you must set the correct execution policy. Run the **Get-ExecutionPolicy** command to determine your current policy level. For information about setting the appropriate level, see [Set-ExecutionPolicy](#).

Next, fill in your Azure subscription name, the cloud service name, and your virtual machine name (removing the < and > characters), and then run these commands.

```
$subscr=<Name of your Azure subscription>
$serviceName=<Name of the cloud service that contains the target virtual machine>
$vmName=<Name of the target virtual machine>
.\InstallWinRMCertAzureVM.ps1 -SubscriptionName $subscr -ServiceName $serviceName -Name $vmName
```

You can get the correct subscription name from the *SubscriptionName* property of the display of the **Get-AzureSubscription** command. You can get the cloud service name for the virtual machine from the *ServiceName* column in the display of the **Get-AzureVM** command.

Check if you have the new certificate. Open a Certificates snap-in for the current user and look in the **Trusted**

Root Certification Authorities\Certificates folder. You should see a certificate with the DNS name of your cloud service in the Issued To column (example: clouddservice4testing.cloudapp.net).

Next, initiate a remote Azure PowerShell session by using these commands.

```
$uri = Get-AzureWinRMUri -ServiceName $serviceName -Name $vmName  
$creds = Get-Credential  
Enter-PSSession -ConnectionUri $uri -Credential $creds
```

After entering valid administrator credentials, you should see something similar to the following Azure PowerShell prompt:

```
[clouddservice4testing.cloudapp.net]: PS C:\Users\User1\Documents>
```

The first part of this prompt is your cloud service name that contains the target VM, which could be different from "clouddservice4testing.cloudapp.net". You can now issue Azure PowerShell commands for this cloud service to investigate the problems mentioned and correct the configuration.

To manually correct the Remote Desktop Services listening TCP port

At the remote Azure PowerShell session prompt, run this command.

```
Get-ItemProperty -Path "HKLM:\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" -Name  
"PortNumber"
```

The PortNumber property shows the current port number. If needed, change the Remote Desktop port number back to its default value (3389) by using this command.

```
Set-ItemProperty -Path "HKLM:\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" -Name  
"PortNumber" -Value 3389
```

Verify that the port has been changed to 3389 by using this command.

```
Get-ItemProperty -Path "HKLM:\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" -Name  
"PortNumber"
```

Exit the remote Azure PowerShell session by using this command.

```
Exit-PSSession
```

Verify that the Remote Desktop endpoint for the Azure VM is also using TCP port 3398 as its internal port. Restart the Azure VM and try the Remote Desktop connection again.

Additional resources

[How to reset a password or the Remote Desktop service for Windows virtual machines](#)

[How to install and configure Azure PowerShell](#)

[Troubleshoot Secure Shell \(SSH\) connections to a Linux-based Azure virtual machine](#)

[Troubleshoot access to an application running on an Azure virtual machine](#)

Troubleshooting specific RDP error messages to a Windows VM in Azure

11/3/2017 • 4 min to read • [Edit Online](#)

You may receive a specific error message when using Remote Desktop connection to a Windows virtual machine (VM) in Azure. This article details some of the more common error messages encountered, along with troubleshooting steps to resolve them. If you are having issues connecting to your VM using RDP but do not encounter a specific error message, see the [troubleshooting guide for Remote Desktop](#).

For information on specific error messages, see the following:

- [The remote session was disconnected because there are no Remote Desktop License Servers available to provide a license.](#)
- [Remote Desktop can't find the computer "name".](#)
- [An authentication error has occurred. The Local Security Authority cannot be contacted.](#)
- [Windows Security error: Your credentials did not work.](#)
- [This computer can't connect to the remote computer.](#)

The remote session was disconnected because there are no Remote Desktop License Servers available to provide a license.

Cause: The 120-day licensing grace period for the Remote Desktop Server role has expired and you need to install licenses.

As a workaround, save a local copy of the RDP file from the portal and run this command at a PowerShell command prompt to connect. This step disables licensing for just that connection:

```
mstsc <File name>.RDP /admin
```

If you don't actually need more than two simultaneous Remote Desktop connections to the VM, you can use Server Manager to remove the Remote Desktop Server role.

For more information, see the blog post [Azure VM fails with "No Remote Desktop License Servers available"](#).

Remote Desktop can't find the computer "name".

Cause: The Remote Desktop client on your computer can't resolve the name of the computer in the settings of the RDP file.

Possible solutions:

- If you're on an organization's intranet, make sure that your computer has access to the proxy server and can send HTTPS traffic to it.
- If you're using a locally stored RDP file, try using the one that's generated by the portal. This step ensures that you have the correct DNS name for the virtual machine, or the cloud service and the endpoint port of the VM. Here is a sample RDP file generated by the portal:

```
full address:s:tailspin-azdatatier.cloudapp.net:55919
prompt for credentials:i:1
```

The address portion of this RDP file has:

- The fully qualified domain name of the cloud service that contains the VM ("tailspin-azdatatier.cloudapp.net" in this example).
- The external TCP port of the endpoint for Remote Desktop traffic (55919).

An authentication error has occurred. The Local Security Authority cannot be contacted.

Cause: The target VM can't locate the security authority in the user name portion of your credentials.

When your user name is in the form *SecurityAuthority\UserName* (example: CORP\User1), the *SecurityAuthority* portion is either the VM's computer name (for the local security authority) or an Active Directory domain name.

Possible solutions:

- If the account is local to the VM, make sure that the VM name is spelled correctly.
- If the account is on an Active Directory domain, check the spelling of the domain name.
- If it is an Active Directory domain account and the domain name is spelled correctly, verify that a domain controller is available in that domain. It's a common issue in Azure virtual networks that contain domain controllers that a domain controller is unavailable because it hasn't been started. As a workaround, you can use a local administrator account instead of a domain account.

Windows Security error: Your credentials did not work.

Cause: The target VM can't validate your account name and password.

A Windows-based computer can validate the credentials of either a local account or a domain account.

- For local accounts, use the *ComputerName\UserName* syntax (example: SQL1\Admin4798).
- For domain accounts, use the *DomainName\UserName* syntax (example: CONTOSO\peterodman).

If you have promoted your VM to a domain controller in a new Active Directory forest, the local administrator account that you signed in with is converted to an equivalent account with the same password in the new forest and domain. The local account is then deleted.

For example, if you signed in with the local account DC1\DCAdmin, and then promoted the virtual machine as a domain controller in a new forest for the corp.contoso.com domain, the DC1\DCAdmin local account gets deleted and a new domain account (CORP\DCAdmin) is created with the same password.

Make sure that the account name is a name that the virtual machine can verify as a valid account, and that the password is correct.

If you need to change the password of the local administrator account, see [How to reset a password or the Remote Desktop service for Windows virtual machines](#).

This computer can't connect to the remote computer.

Cause: The account that's used to connect does not have Remote Desktop sign-in rights.

Every Windows computer has a Remote Desktop users local group, which contains the accounts and groups that can sign into it remotely. Members of the local administrators group also have access, even though those accounts are not listed in the Remote Desktop users local group. For domain-joined machines, the local administrators group also contains the domain administrators for the domain.

Make sure that the account you're using to connect with has Remote Desktop sign-in rights. As a workaround, use a domain or local administrator account to connect over Remote Desktop. To add the desired account to the Remote

Desktop users local group, use the Microsoft Management Console snap-in (**System Tools > Local Users and Groups > Groups > Remote Desktop Users**).

Next steps

If none of these errors occurred and you have an unknown issue with connecting using RDP, see the [troubleshooting guide for Remote Desktop](#).

- For troubleshooting steps in accessing applications running on a VM, see [Troubleshoot access to an application running on an Azure VM](#).
- If you are having issues using Secure Shell (SSH) to connect to a Linux VM in Azure, see [Troubleshoot SSH connections to a Linux VM in Azure](#).

How to reset the Remote Desktop service or its login password in a Windows VM

12/15/2017 • 2 min to read • [Edit Online](#)

If you can't connect to a Windows virtual machine (VM), you can reset the local administrator password or reset the Remote Desktop service configuration (not supported on Windows Domain Controllers). You can use either the Azure portal or the VM Access extension in Azure PowerShell to reset the password. If you are using PowerShell, make sure that you have the [latest PowerShell module installed and configured](#) and are signed in to your Azure subscription. You can also [perform these steps for VMs created with the Classic deployment model](#).

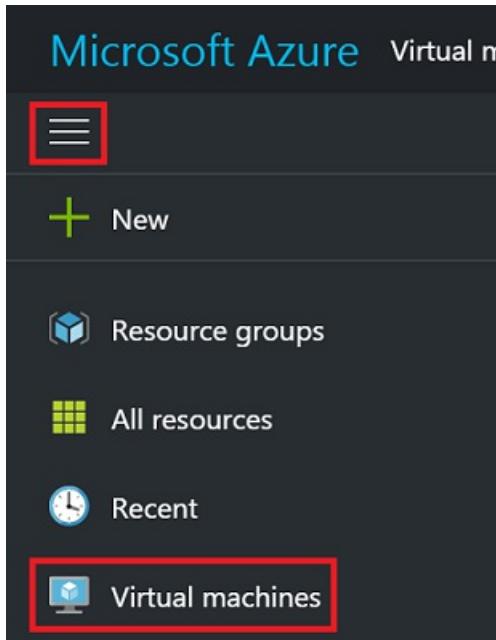
Ways to reset configuration or credentials

You can reset Remote Desktop services and credentials in a few different ways, depending on your needs:

- [Reset using the Azure portal](#)
- [Reset using Azure PowerShell](#)

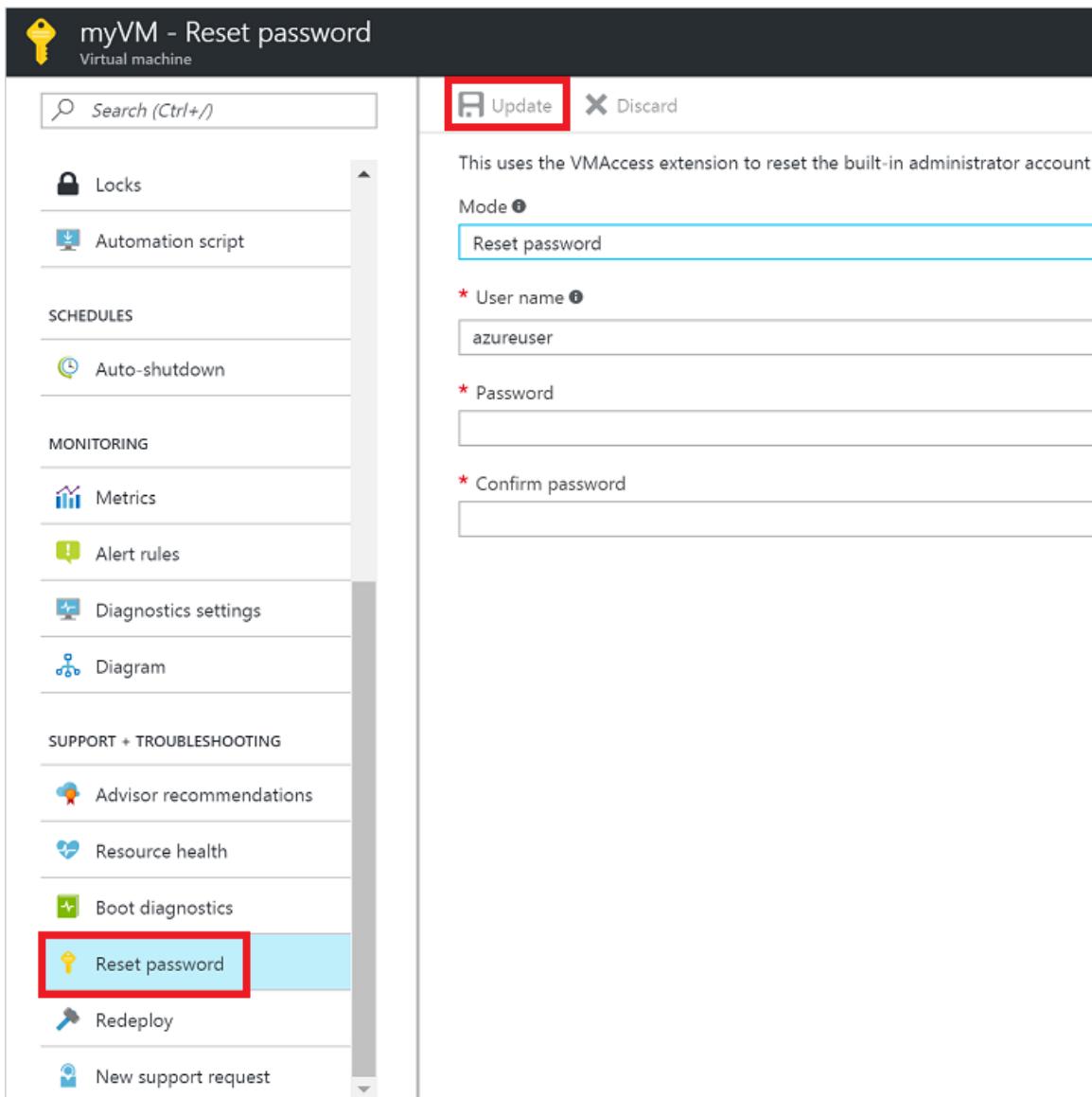
Azure portal

To expand the portal menu, click the three bars in the upper left corner and then click **Virtual machines**:



Reset the local administrator account password

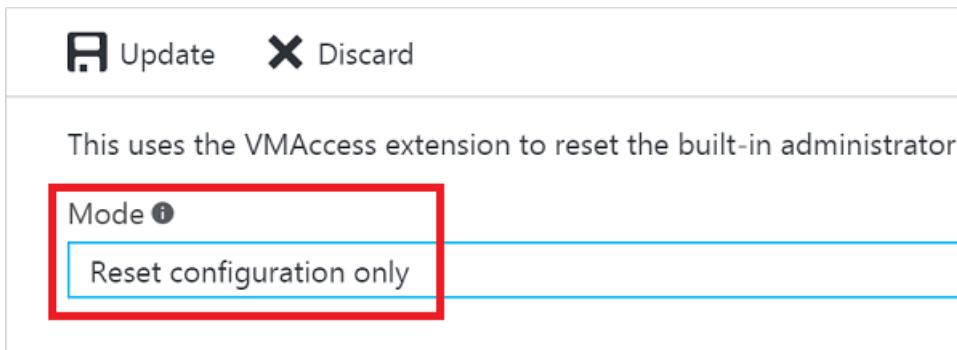
Select your Windows virtual machine then click **Support + Troubleshooting > Reset password**. The password reset blade is displayed:



Enter the username and a new password, then click **Update**. Try connecting to your VM again.

Reset the Remote Desktop service configuration

Select your Windows virtual machine then click **Support + Troubleshooting > Reset password**. The password reset blade is displayed.



Select **Reset configuration only** from the drop-down menu, then click **Update**. Try connecting to your VM again.

VMAccess extension and PowerShell

Make sure that you have the [latest PowerShell module installed and configured](#) and are signed in to your Azure subscription with the `Login-AzureRmAccount` cmdlet.

Reset the local administrator account password

Reset the administrator password or user name with the [Set-AzureRmVMAccessExtension](#) PowerShell cmdlet.

Create your account credentials as follows:

```
$cred=Get-Credential
```

NOTE

If you type a different name than the current local administrator account on your VM, the VMAccess extension will add a local administrator account with that name, and assign your specified password to that account. If the local administrator account on your VM exists, it will reset the password and if the account is disabled, the VMAccess extension enables it.

The following example updates the VM named `myVM` in the resource group named `myResourceGroup` to the credentials specified.

```
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" -VMName "myVM" -Name "myVMAccess" -Location WestUS -UserName $cred.GetNetworkCredential().UserName -Password $cred.GetNetworkCredential().Password -typeHandlerVersion "2.0"
```

Reset the Remote Desktop service configuration

Reset remote access to your VM with the [Set-AzureRmVMAccessExtension](#) PowerShell cmdlet. The following example resets the access extension named `myVMAccess` on the VM named `myVM` in the `myResourceGroup` resource group:

```
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" -VMName "myVM" -Name "myVMAccess" -Location WestUS -typeHandlerVersion "2.0" -ForceRerun
```

TIP

At any point, a VM can have only a single VM access agent. To set the VM access agent properties successfully, the `-ForceRerun` option can be used. When using `-ForceRerun`, make sure to use the same name for the VM access agent as used in any previous commands.

If you still can't connect remotely to your virtual machine, see more steps to try at [Troubleshoot Remote Desktop connections to a Windows-based Azure virtual machine](#).

Next steps

If the Azure VM access extension does not respond and you are unable to reset the password, you can [reset the local Windows password offline](#). This method is a more advanced process and requires you to connect the virtual hard disk of the problematic VM to another VM. Follow the steps documented in this article first, and only attempt the offline password reset method as a last resort.

[Azure VM extensions and features](#)

[Connect to an Azure virtual machine with RDP or SSH](#)

[Troubleshoot Remote Desktop connections to a Windows-based Azure virtual machine](#)

How to reset local Windows password for Azure VM

6/27/2017 • 4 min to read • [Edit Online](#)

You can reset the local Windows password of a VM in Azure using the [Azure portal](#) or [Azure PowerShell](#) provided the Azure guest agent is installed. This method is the primary way to reset a password for an Azure VM. If you encounter issues with the Azure guest agent not responding, or failing to install after uploading a custom image, you can manually reset a Windows password. This article details how to reset a local account password by attaching the source OS virtual disk to another VM.

WARNING

Only use this process as a last resort. Always try to reset a password using the [Azure portal](#) or [Azure PowerShell](#) first.

Overview of the process

The core steps for performing a local password reset for a Windows VM in Azure when there is no access to the Azure guest agent is as follows:

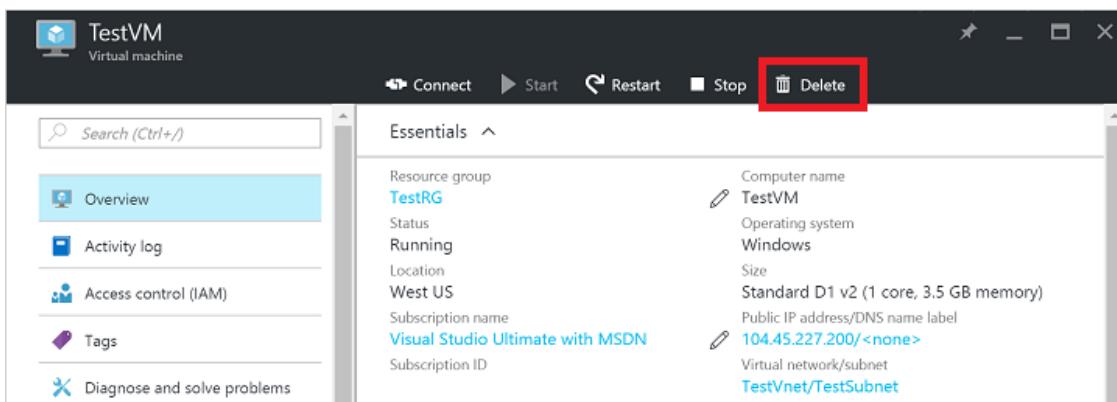
- Delete the source VM. The virtual disks are retained.
- Attach the source VM's OS disk to another VM on the same location within your Azure subscription. This VM is referred to as the troubleshooting VM.
- Using the troubleshooting VM, create some config files on the source VM's OS disk.
- Detach the VM's OS disk from the troubleshooting VM.
- Use a Resource Manager template to create a VM, using the original virtual disk.
- When the new VM boots, the config files you create update the password of the required user.

Detailed steps

Always try to reset a password using the [Azure portal](#) or [Azure PowerShell](#) before trying the following steps. Make sure you have a backup of your VM before you start.

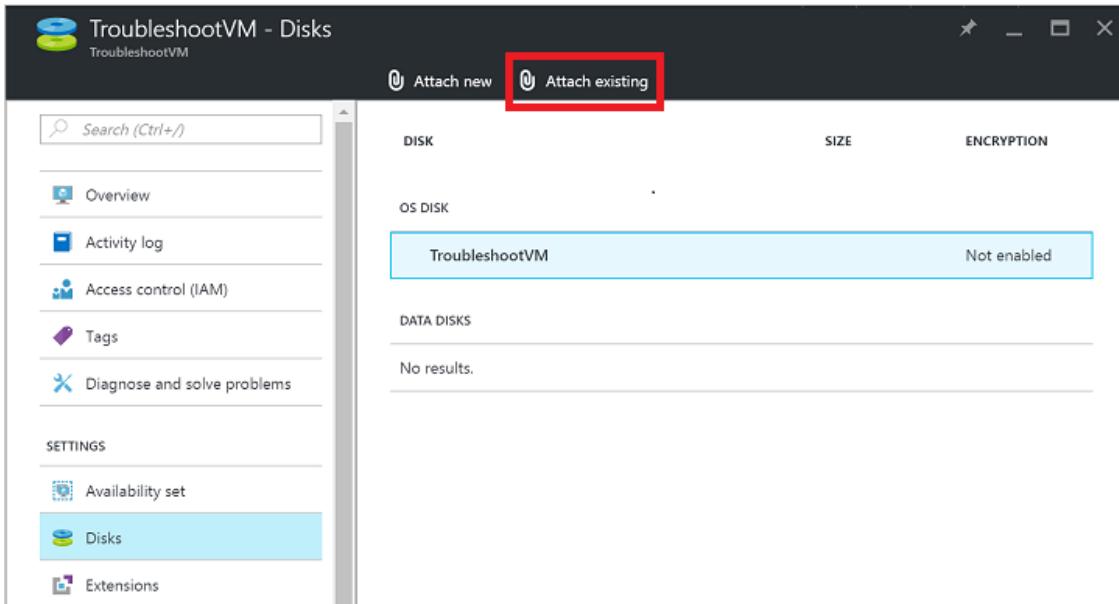
1. Delete the affected VM in Azure portal. Deleting the VM only deletes the metadata, the reference of the VM within Azure. The virtual disks are retained when the VM is deleted:

- Select the VM in the Azure portal, click *Delete*:

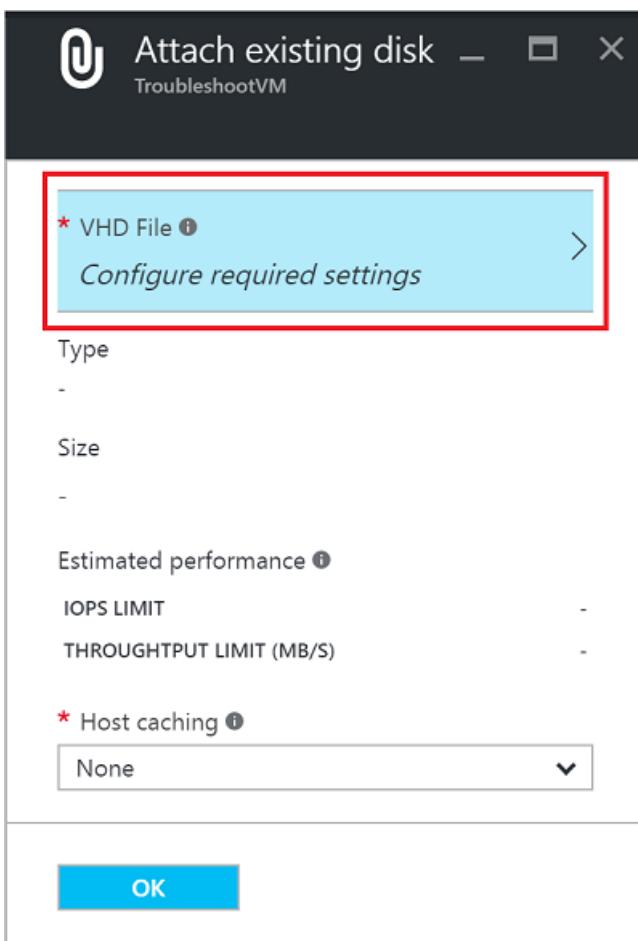


2. Attach the source VM's OS disk to the troubleshooting VM. The troubleshooting VM must be in the same region as the source VM's OS disk (such as `West US`):

- Select the troubleshooting VM in the Azure portal. Click *Disks | Attach existing*:



Select *VHD File* and then select the storage account that contains your source VM:



Select the source container. The source container is typically *vhds*:

A screenshot of the Azure Storage Container list interface. The title bar says 'Containers' and 'testrgvmsstorage'. There are buttons for '+ Container' and 'Refresh'. A search bar at the top says 'Search containers by prefix'. Below is a table with columns 'NAME' and 'LAST MODIFIED'. A single row is shown: 'vhds' with the last modified date 'Fri Sep 02 2016 14:26:58 GMT-0700 (Pacific Da...'. The 'vhds' cell is highlighted with a red border.

NAME	LAST MODIFIED
vhds	Fri Sep 02 2016 14:26:58 GMT-0700 (Pacific Da...

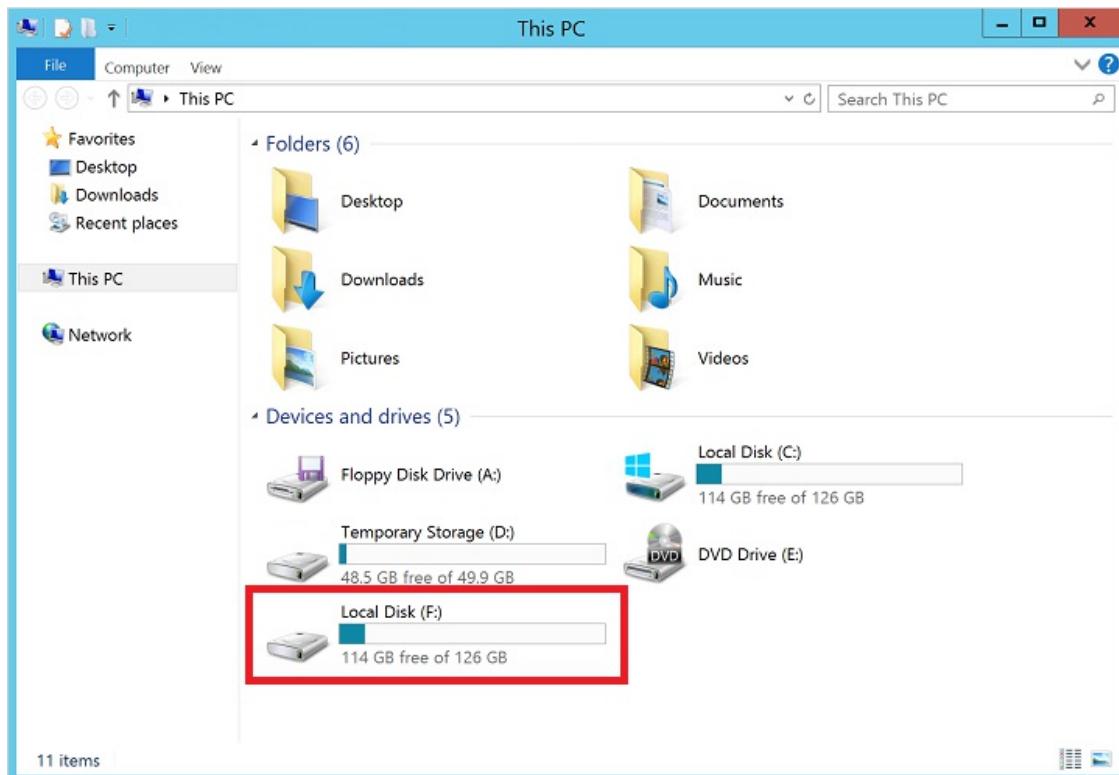
Select the OS vhd to attach. Click *Select* to complete the process:

A screenshot of the Azure Storage Blob list interface. The title bar says 'vhds' and 'https://testrgvmsstorage.blob.core.windows.net/vhds'. There is a 'Refresh' button. A search bar at the top says 'Search blobs by prefix (case-sensitive)'. Below is a table with columns 'NAME' and 'SIZE'. Three items are listed: 'TestVM201682142612.vhd' (136.37 GB), 'TroubleshootVM.f3d1d6c8-9f2a-4e88-9644-4be3243f4bb0.status' (242 B), and 'TroubleshootVM20168214305.vhd' (136.37 GB). The first item, 'TestVM201682142612.vhd', is highlighted with a red border. At the bottom is a blue 'Select' button.

NAME	SIZE
TestVM201682142612.vhd	136.37 GB
TroubleshootVM.f3d1d6c8-9f2a-4e88-9644-4be3243f4bb0.status	242 B
TroubleshootVM20168214305.vhd	136.37 GB

3. Connect to the troubleshooting VM using Remote Desktop and ensure the source VM's OS disk is visible:

- Select the troubleshooting VM in the Azure portal and click *Connect*.
- Open the RDP file that downloads. Enter the username and password of the troubleshooting VM.
- In File Explorer, look for the data disk you attached. If the source VM's VHD is the only data disk attached to the troubleshooting VM, it should be the F: drive:



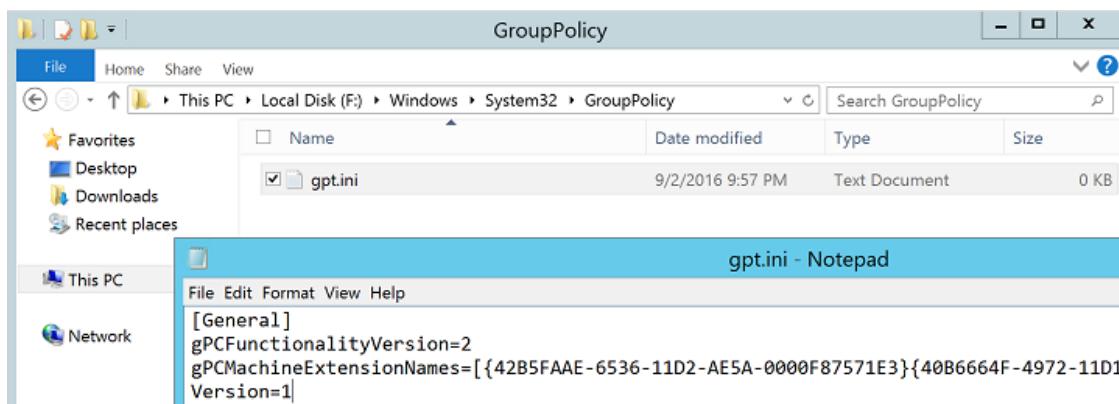
4. Create `gpt.ini` in `\Windows\System32\GroupPolicy` on the source VM's drive (if `gpt.ini` exists, rename to `gpt.ini.bak`):

WARNING

Make sure that you do not accidentally create the following files in C:\Windows, the OS drive for the troubleshooting VM. Create the following files in the OS drive for your source VM that is attached as a data disk.

- Add the following lines into the `gpt.ini` file you created:

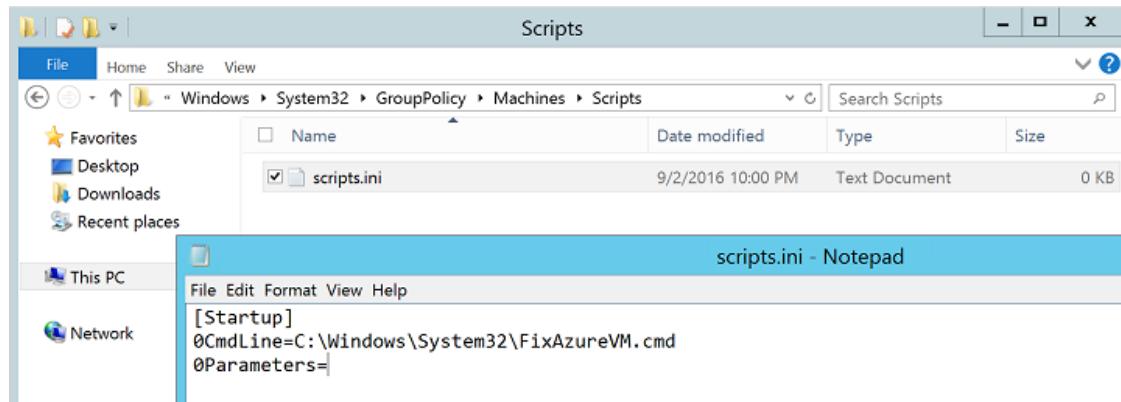
```
[General]
gPCFunctionalityVersion=2
gPCMchineExtensionNames=[{42B5FAAE-6536-11D2-AE5A-0000F87571E3}{40B6664F-4972-11D1-A7CA-
0000F87571E3}]
Version=1
```



5. Create `scripts.ini` in `\Windows\System32\GroupPolicy\Machine\Scripts`. Make sure hidden folders are shown. If needed, create the `Machine` or `Scripts` folders.

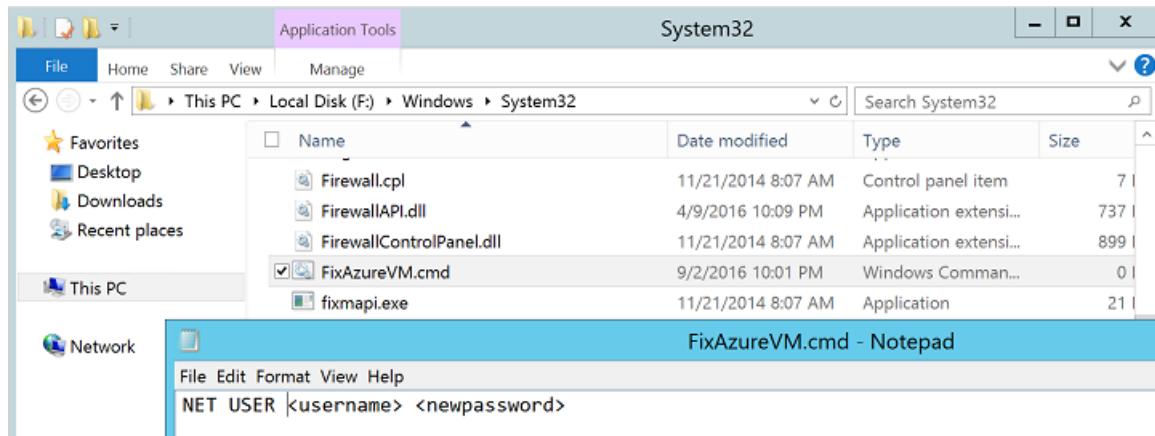
- Add the following lines the `scripts.ini` file you created:

```
[Startup]
@CmdLine=C:\Windows\System32\FixAzureVM.cmd
@Parameters=
```



6. Create `FixAzureVM.cmd` in `\Windows\System32` with the following contents, replacing `<username>` and `<newpassword>` with your own values:

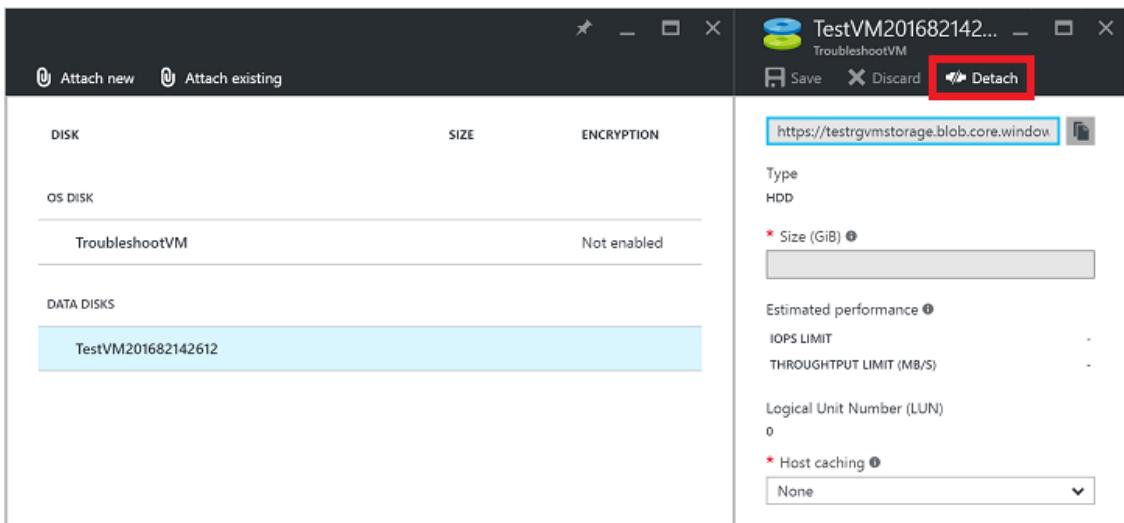
```
net user <username> <newpassword> /add
net localgroup administrators <username> /add
net localgroup "remote desktop users" <username> /add
```



You must meet the configured password complexity requirements for your VM when defining the new password.

7. In Azure portal, detach the disk from the troubleshooting VM:

- Select the troubleshooting VM in the Azure portal, click *Disks*.
- Select the data disk attached in step 2, click *Detach*:



8. Before you create a VM, obtain the URI to your source OS disk:

- Select the storage account in the Azure portal, click *Blobs*.
- Select the container. The source container is typically *vhds*:

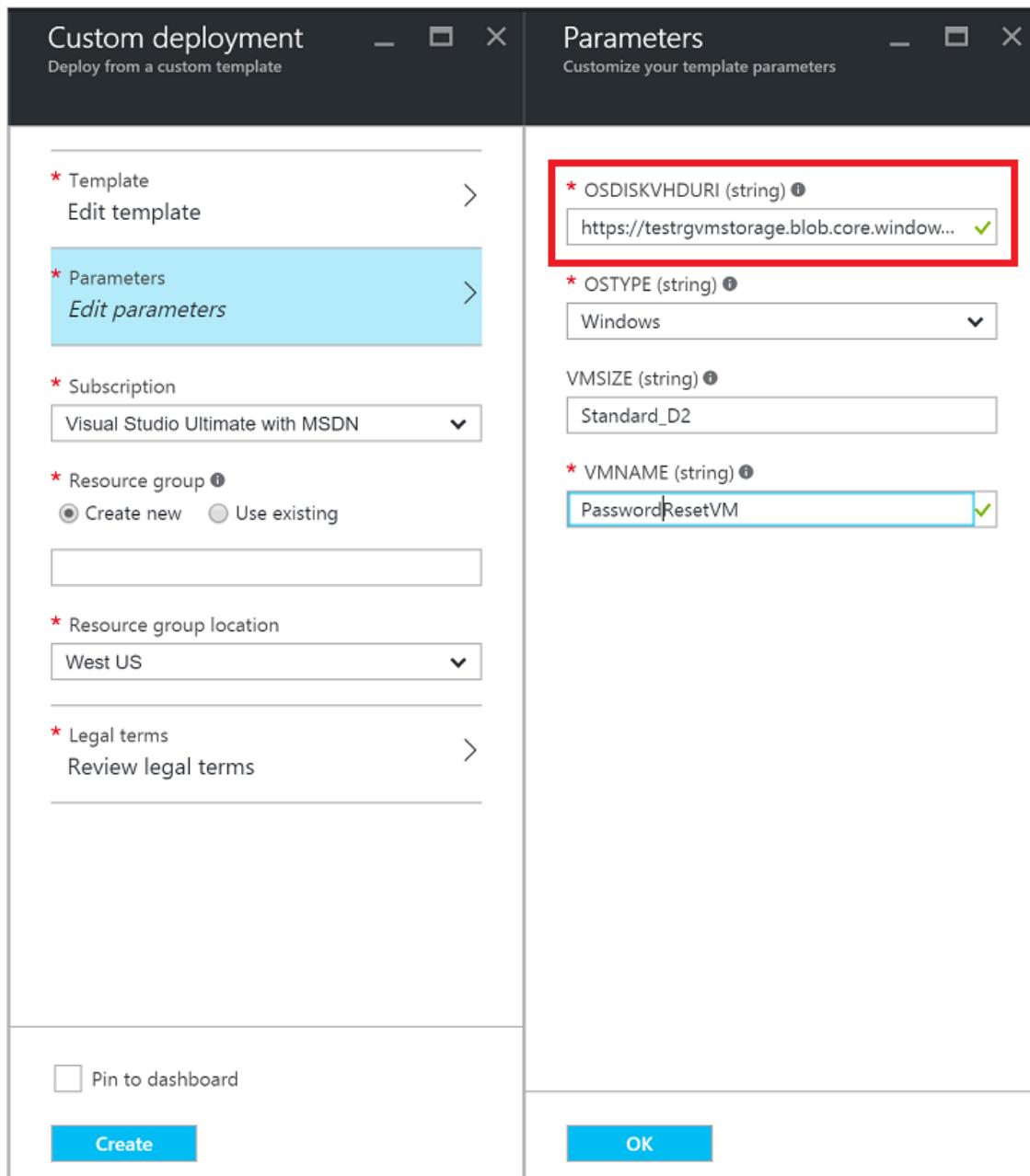
NAME	URL
vhds	https://testrgvmsstorage.blob.core.windows.net/vhds

Select your source VM OS VHD and click the *Copy* button next to the *URL* name:

NAME	URL
TestVM201682142612.vhd	https://testrgvmsstorage.blob.core.windows.net/TestVM201682142612.vhd

9. Create a VM from the source VM's OS disk:

- Use [this Azure Resource Manager template](#) to create a VM from a specialized VHD. Click the *Deploy to Azure* button to open the Azure portal with the templated details populated for you.
- If you want to retain all the previous settings for the VM, select *Edit template* to provide your existing VNet, subnet, network adapter, or public IP.
- In the *OSDISKVHDURI* parameter text box, paste the URI of your source VHD obtain in the preceding step:



- After the new VM is running, connect to the VM using Remote Desktop with the new password you specified in the `FixAzureVM.cmd` script.
- From your remote session to the new VM, remove the following files to clean up the environment:
 - From %windir%\System32
 - remove FixAzureVM.cmd
 - From %windir%\System32\GroupPolicy\Machine\
 - remove scripts.ini
 - From %windir%\System32\GroupPolicy
 - remove gpt.ini (if gpt.ini existed before, and you renamed it to gpt.ini.bak, rename the .bak file back to gpt.ini)

Next steps

If you still cannot connect using Remote Desktop, see the [RDP troubleshooting guide](#). The [detailed RDP troubleshooting guide](#) looks at troubleshooting methods rather than specific steps. You can also [open an Azure support request](#) for hands-on assistance.

Understand a system reboot for Azure VM

11/3/2017 • 7 min to read • [Edit Online](#)

Azure virtual machines (VMs) might sometimes reboot for no apparent reason, without evidence of your having initiated the reboot operation. This article lists the actions and events that can cause VMs to reboot and provides insight into how to avoid unexpected reboot issues or reduce the impact of such issues.

Configure the VMs for high availability

The best way to protect an application that's running on Azure against VM reboots and downtime is to configure the VMs for high availability.

To provide this level of redundancy to your application, we recommend that you group two or more VMs in an availability set. This configuration ensures that during either a planned or unplanned maintenance event, at least one VM is available and meets the 99.95 percent [Azure SLA](#).

For more information about availability sets, see the following articles:

- [Manage the availability of VMs](#)
- [Configure availability of VMs](#)

Resource Health information

Azure Resource Health is a service that exposes the health of individual Azure resources and provides actionable guidance for troubleshooting problems. In a cloud environment where it isn't possible to directly access servers or infrastructure elements, the goal of Resource Health is to reduce the time that you spend on troubleshooting. In particular, the aim is to reduce the time that you spend determining whether the root of the problem lies in the application or in an event inside the Azure platform. For more information, see [Understand and use Resource Health](#).

Actions and events that can cause the VM to reboot

Planned maintenance

Microsoft Azure periodically performs updates across the globe to improve the reliability, performance, and security of the host infrastructure that underlies VMs. Many of these updates, including memory-preserving updates, are performed without any impact on your VMs or cloud services.

However, some updates do require a reboot. In such cases, the VMs are shut down while we patch the infrastructure, and then the VMs are restarted.

To understand what Azure planned maintenance is and how it can affect the availability of your Linux VMs, see the articles listed here. The articles provide background about the Azure planned maintenance process and how to schedule planned maintenance to further reduce the impact.

- [Planned maintenance for VMs in Azure](#)
- [How to schedule planned maintenance on Azure VMs](#)

Memory-preserving updates

For this class of updates in Microsoft Azure, users experience no impact on their running VMs. Many of these updates are to components or services that can be updated without interfering with the running instance. Some are platform infrastructure updates on the host operating system that can be applied without a reboot of the VMs.

These memory-preserving updates are accomplished with technology that enables in-place live migration. When it is being updated, the VM is placed in a *paused* state. This state preserves the memory in RAM while the underlying host operating system receives the necessary updates and patches. The VM is resumed within 30 seconds of being paused. After the VM is resumed, its clock is automatically synchronized.

Because of the short pause period, deploying updates through this mechanism greatly reduces the impact on the VMs. However, not all updates can be deployed in this way.

Multi-instance updates (for VMs in an availability set) are applied one update domain at a time.

NOTE

Linux machines that have old kernel versions are affected by a kernel panic during this update method. To avoid this issue, update to kernel version 3.10.0-327.10.1 or later. For more information, see [An Azure Linux VM on a 3.10-based kernel panics after a host node upgrade](#).

User-initiated reboot or shutdown actions

If you perform a reboot from the Azure portal, Azure PowerShell, command-line interface, or Reset API, you can find the event in the [Azure Activity Log](#).

If you perform the action from the VM's operating system, you can find the event in the system logs.

Other scenarios that usually cause the VM to reboot include multiple configuration-change actions. You'll ordinarily see a warning message indicating that executing a particular action will result in a reboot of the VM. Examples include any VM resize operations, changing the password of the administrative account, and setting a static IP address.

Azure Security Center and Windows Update

Azure Security Center monitors daily Windows and Linux VMs for missing operating-system updates. Security Center retrieves a list of available security and critical updates from Windows Update or Windows Server Update Services (WSUS), depending on which service is configured on a Windows VM. Security Center also checks for the latest updates for Linux systems. If your VM is missing a system update, Security Center recommends that you apply system updates. The application of these system updates is controlled through the Security Center in the Azure portal. After you apply some updates, VM reboots might be required. For more information, see [Apply system updates in Azure Security Center](#).

Like on-premises servers, Azure does not push updates from Windows Update to Windows Azure VMs, because these machines are intended to be managed by their users. You are, however, encouraged to leave the automatic Windows Update setting enabled. Automatic installation of updates from Windows Update can also cause reboots to occur after the updates are applied. For more information, see [Windows Update FAQ](#).

Other situations affecting the availability of your VM

There are other cases in which Azure might actively suspend the use of a VM. You'll receive email notifications before this action is taken, so you'll have a chance to resolve the underlying issues. Examples of issues that affect VM availability include security violations and the expiration of payment methods.

Host server faults

The VM is hosted on a physical server that is running inside an Azure datacenter. The physical server runs an agent called the Host Agent in addition to a few other Azure components. When these Azure software components on the physical server become unresponsive, the monitoring system triggers a reboot of the host server to attempt recovery. The VM is usually available again within five minutes and continues to live on the same host as previously.

Server faults are usually caused by hardware failure, such as the failure of a hard disk or solid-state drive. Azure continuously monitors these occurrences, identifies the underlying bugs, and rolls out updates after the mitigation

has been implemented and tested.

Because some host server faults can be specific to that server, a repeated VM reboot situation might be improved by manually redeploying the VM to another host server. This operation can be triggered by using the **redeploy** option on the details page of the VM, or by stopping and restarting the VM in the Azure portal.

Auto-recovery

If the host server cannot reboot for any reason, the Azure platform initiates an auto-recovery action to take the faulty host server out of rotation for further investigation.

All VMs on that host are automatically relocated to a different, healthy host server. This process is usually complete within 15 minutes. To learn more about the auto-recovery process, see [Auto-recovery of VMs](#).

Unplanned maintenance

On rare occasions, the Azure operations team might need to perform maintenance activities to ensure the overall health of the Azure platform. This behavior might affect VM availability, and it usually results in the same auto-recovery action as described earlier.

Unplanned maintenances include the following:

- Urgent node defragmentation
- Urgent network switch updates

VM crashes

VMs might restart because of issues within the VM itself. The workload or role that's running on the VM might trigger a bug check within the guest operating system. For help determining the reason for the crash, view the system and application logs for Windows VMs, and the serial logs for Linux VMs.

Storage-related forced shutdowns

VMs in Azure rely on virtual disks for operating system and data storage that is hosted on the Azure Storage infrastructure. Whenever the availability or connectivity between the VM and the associated virtual disks is affected for more than 120 seconds, the Azure platform performs a forced shutdown of the VMs to avoid data corruption. The VMs are automatically powered back on after storage connectivity has been restored.

The duration of the shutdown can be as short as five minutes but can be significantly longer. The following is one of the specific cases that is associated with storage-related forced shutdowns:

Exceeding IO limits

VMs might be temporarily shut down when I/O requests are consistently throttled because the volume of I/O operations per second (IOPS) exceeds the I/O limits for the disk. (Standard disk storage is limited to 500 IOPS.) To mitigate this issue, use disk striping or configure the storage space inside the guest VM, depending on the workload. For details, see [Configuring Azure VMs for Optimal Storage Performance](#).

Higher IOPS limits are available via Azure Premium Storage with up to 80,000 IOPS. For more information, see [High-Performance Premium Storage](#).

Other incidents

In rare circumstances, a widespread issue can affect multiple servers in an Azure datacenter. If this issue occurs, the Azure team sends email notifications to the affected subscriptions. You can check the [Azure Service Health dashboard](#) and the Azure portal for the status of ongoing outages and past incidents.

How to reset network interface for Azure Windows VM

11/3/2017 • 2 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

You cannot connect to Microsoft Azure Windows Virtual Machine (VM) after you disable the default Network Interface (NIC) or manually sets a static IP for the NIC. This article shows how to reset the network interface for Azure Windows VM, which will resolve the remote connection issue.

If your Azure issue is not addressed in this article, visit the Azure forums on [MSDN and the Stack Overflow](#). You can post your issue in these forums, or post to [@AzureSupport on Twitter](#). You also can submit an Azure support request. To submit a support request, on the [Azure support](#) page, select **Get support**.

Reset network interface

For Classic VMs

To reset network interface, follow these steps:

1. Go to the [Azure portal](#).
2. Select **Virtual Machines (Classic)**.
3. Select the affected Virtual Machine.
4. Select **IP addresses**.
5. If the **Private IP assignment** is not **Static**, change it to **Static**.
6. Change the **IP address** to another IP address that is available in the Subnet.
7. Select Save.
8. The virtual machine will restart to initialize the new NIC to the system.
9. Try to RDP to your machine. If successful, you can change the Private IP address back to the original if you would like. Otherwise, you can keep it.

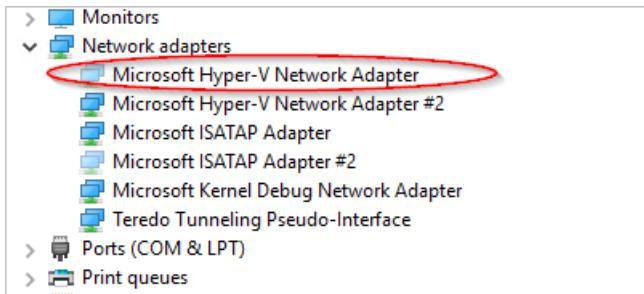
For VMs deployed in Resource group model

1. Go to the [Azure portal](#).
2. Select the affected Virtual Machine.
3. Select **Network Interfaces**.
4. Select the Network Interface associated with your machine
5. Select **IP configurations**.
6. Select the IP.
7. If the **Private IP assignment** is not **Static**, change it to **Static**.
8. Change the **IP address** to another IP address that is available in the Subnet.
9. The virtual machine will restart to initialize the new NIC to the system.
10. Try to RDP to your machine. If successful, you can change the Private IP address back to the original if you would like. Otherwise, you can keep it.

Delete the unavailable NICs

After you can remote desktop to the machine, you must delete the old NICs to avoid the potential problem:

1. Open Device Manager.
2. Select **View > Show hidden devices**.
3. Select **Network Adapters**.
4. Check for the adapters named as "Microsoft Hyper-V Network Adapter".
5. You might see an unavailable adapter that is grayed out. Right-click the adapter and then select Uninstall.



NOTE

Only uninstall the unavailable adapters that have the name "Microsoft Hyper-V Network Adapter". If you uninstall any of the other hidden adapters, it could cause additional issues.

6. Now all unavailable adapter should be cleared out from your system.

How to use boot diagnostics to troubleshoot Windows virtual machines in Azure

9/27/2017 • 1 min to read • [Edit Online](#)

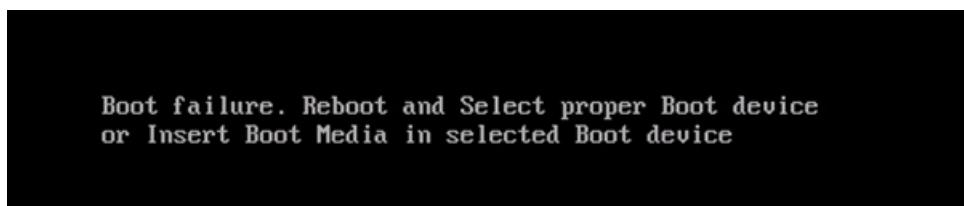
Support for two debugging features is now available in Azure: Console Output and Screenshot support for Azure Virtual Machines Resource Manager deployment model.

When bringing your own image to Azure or even booting one of the platform images, there can be many reasons why a Virtual Machine gets into a non-bootable state. These features enable you to easily diagnose and recover your Virtual Machines from boot failures.

For Linux Virtual Machines, you can easily view the output of your console log from the Portal:

```
[M][3[1;H 0.000000] Initializing cgroup subsys cpuset
[ 0.000000] Initializing cgroup subsys cpu
[ 0.000000] Initializing cgroup subsys cpacct
[ 0.000000] Linux version 3.19.0-25-generic (buildd@lgw01-20)
(gcc version 4.8.2 (Ubuntu 4.8.2-19ubuntu1) ) #26-14.04.1-Ubuntu
SMP Fri Jul 24 21:16:20 UTC 2015 (Ubuntu
3.19.0-25.26-14.04.1-generic 3.19.8-ckt2)
[ 0.000000] Command line:
BOOT_IMAGE=/boot/vmlinuz-3.19.0-25-generic
root=UUID=3fe10351-012c-4a01-9c3a-0e991100f2f5 ro console=tty1
console=ttyS0 earlyprintk=ttyS0 rootdelay=300
[ 0.000000] KERNEL supported cpus:
[ 0.000000] Intel GenuineIntel
[ 0.000000] AMD AuthenticAMD
[ 0.000000] Centaur CentaurHauls
[ 0.000000] BIOS e820: BIOS-provided physical RAM map:
[ 0.000000] BIOS-e820: [mem
0x0000000000000000-0x0000000000000fbfff] usable
[ 0.000000] BIOS-e820: [mem
0x00000000009fc00-0x000000000000ffff] reserved
[ 0.000000] BIOS-e820: [mem
0x0000000000e0000-0x00000000000ffff] reserved
[ 0.000000] BIOS-e820: [mem
0x0000000000100000-0x000000001ffff] usable
[ 0.000000] BIOS-e820: [mem
0x0000000000100000-0x000000001ffff] usable
```

However, for both Windows and Linux Virtual Machines, Azure also enables you to see a screenshot of the VM from the hypervisor:



Both of these features are supported for Azure Virtual Machines in all regions. Note, screenshots, and output can take up to 10 minutes to appear in your storage account.

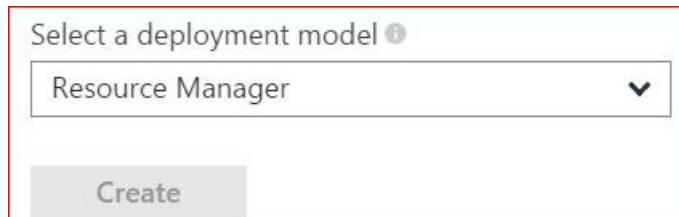
Common boot errors

- [0xC000000E](#)
- [0xC000000F](#)
- [0xC0000011](#)
- [0xC0000034](#)
- [0xC0000098](#)
- [0xC00000BA](#)
- [0xC000014C](#)
- [0xC0000221](#)

- 0xC0000225
- 0xC0000359
- 0xC0000605
- An operating system wasn't found
- Boot failure or INACCESSIBLE_BOOT_DEVICE

Enable diagnostics on a new virtual machine

1. When creating a new Virtual Machine from the Preview Portal, select the **Azure Resource Manager** from the deployment model dropdown:



2. Configure the Monitoring option to select the storage account where you would like to place these diagnostic files.

A screenshot of the 'Create virtual machine' wizard. The left sidebar shows steps 1 through 4: Basics (Done), Size (Done), Settings (Configure optional features), and Summary (Ubuntu Server 14.04 LTS). The 'Settings' tab is active. On the right, under 'Storage', 'Standard' is selected and 'Premium (SSD)' is highlighted. Under 'Network', a virtual network and subnet are configured. Under 'Monitoring', 'Diagnostics' is set to 'Enabled' and a storage account '(new) linuxvm4471' is selected. A red box highlights the 'Monitoring' section.

3. If you are deploying from an Azure Resource Manager template, navigate to your Virtual Machine resource

and append the diagnostics profile section. Remember to use the "2015-06-15" API version header.

```
{  
    "apiVersion": "2015-06-15",  
    "type": "Microsoft.Compute/virtualMachines",  
    ...
```

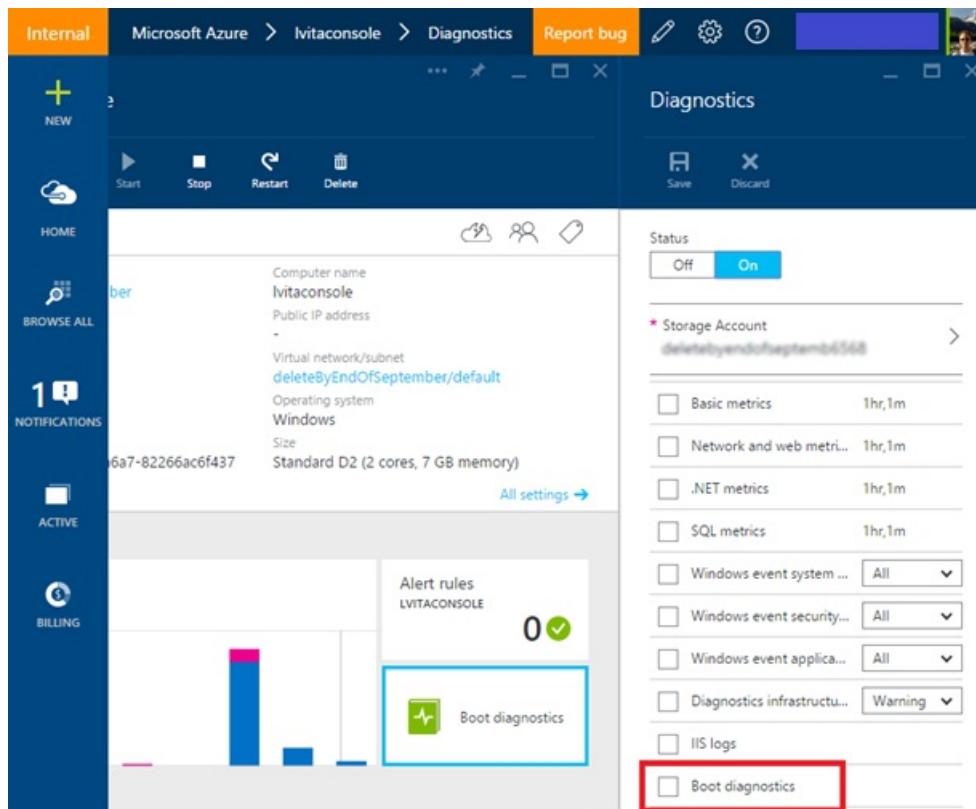
4. The diagnostics profile enables you to select the storage account where you want to put these logs.

```
"diagnosticsProfile": {  
    "bootDiagnostics": {  
        "enabled": true,  
        "storageUri": "[concat('http://', parameters('newStorageAccountName'),  
.blob.core.windows.net')]"  
    }  
}
```

To deploy a sample Virtual Machine with boot diagnostics enabled, check out our repo here.

Update an existing virtual machine

To enable boot diagnostics through the Portal, you can also update an existing Virtual Machine through the Portal. Select the Boot Diagnostics option and Save. Restart the VM to take effect.



Troubleshoot deployment issues when creating a new Windows VM in Azure

11/3/2017 • 4 min to read • [Edit Online](#)

When you try to create a new Azure Virtual Machine (VM), the common errors you encounter are provisioning failures or allocation failures.

- A provisioning failure happens when the OS image fails to load either due to incorrect preparatory steps or because of selecting the wrong settings during the image capture from the portal.
- An allocation failure results when the cluster or region either does not have resources available or cannot support the requested VM size.

If your Azure issue is not addressed in this article, visit the Azure forums on [MSDN and the Stack Overflow](#). You can post your issue in these forums, or post to [@AzureSupport on Twitter](#). You also can submit an Azure support request. To submit a support request, on the [Azure support](#) page, select **Get support**.

Top issues

The following top issues may help resolve your issue. To start troubleshooting, review these steps:

- [The cluster cannot support the requested VM size](#)
- [The cluster does not have free resources](#)

For other VM deployment issues and questions, see [Troubleshoot deploying Windows virtual machine issues in Azure](#).

Collect activity logs

To start troubleshooting, collect the activity logs to identify the error associated with the issue. The following links contain detailed information on the process to follow.

[View deployment operations](#)

[View activity logs to manage Azure resources](#)

Issue: Custom image; provisioning errors

Provisioning errors arise if you upload or capture a generalized VM image as a specialized VM image or vice versa. The former will cause a provisioning timeout error and the latter will cause a provisioning failure. To deploy your custom image without errors, you must ensure that the type of the image does not change during the capture process.

The following table lists the possible combinations of generalized and specialized images, the error type you will encounter and what you need to do to fix the errors.

The following table lists the possible upload and capture combinations of Windows generalized (gen.) and specialized (spec.) OS images. The combinations that will process without any errors are indicated by a Y, and those that will throw errors are indicated by an N. The causes and resolutions for the different errors you will run into are given below the table.

OS	UPLOAD SPEC.	UPLOAD GEN.	CAPTURE SPEC.	CAPTURE GEN.
Windows gen.	N ¹	Y	N ³	Y
Windows spec.	Y	N ²	Y	N ⁴

Y: If the OS is Windows generalized, and it is uploaded and/or captured with the generalized setting, then there won't be any errors. Similarly, if the OS is Windows specialized, and it is uploaded and/or captured with the specialized setting, then there won't be any errors.

Upload Errors:

N¹: If the OS is Windows generalized, and it is uploaded as specialized, you will get a provisioning timeout error with the VM stuck at the OOB screen.

N²: If the OS is Windows specialized, and it is uploaded as generalized, you will get a provisioning failure error with the VM stuck at the OOB screen because the new VM is running with the original computer name, username and password.

Resolution

To resolve both these errors, use [Add-AzureRmVhd to upload the original VHD](#), available on-premises, with the same setting as that for the OS (generalized/specialized). To upload as generalized, remember to run sysprep first.

Capture Errors:

N³: If the OS is Windows generalized, and it is captured as specialized, you will get a provisioning timeout error because the original VM is not usable as it is marked as generalized.

N⁴: If the OS is Windows specialized, and it is captured as generalized, you will get a provisioning failure error because the new VM is running with the original computer name, username, and password. Also, the original VM is not usable because it is marked as specialized.

Resolution

To resolve both these errors, delete the current image from the portal, and [recapture it from the current VHDs](#) with the same setting as that for the OS (generalized/specialized).

Issue: Custom/gallery/marketplace image; allocation failure

This error arises in situations when the new VM request is pinned to a cluster that either cannot support the VM size being requested, or does not have available free space to accommodate the request.

Cause 1: The cluster cannot support the requested VM size.

Resolution 1:

- Retry the request using a smaller VM size.
- If the size of the requested VM cannot be changed:
 - Stop all the VMs in the availability set. Click **Resource groups** > *your resource group* > **Resources** > *your availability set* > **Virtual Machines** > *your virtual machine* > **Stop**.
 - After all the VMs stop, create the new VM in the desired size.
 - Start the new VM first, and then select each of the stopped VMs and click **Start**.

Cause 2: The cluster does not have free resources.

Resolution 2:

- Retry the request at a later time.
- If the new VM can be part of a different availability set
 - Create a new VM in a different availability set (in the same region).
 - Add the new VM to the same virtual network.

Next steps

If you encounter issues when you start a stopped Windows VM or resize an existing Windows VM in Azure, see [Troubleshoot Resource Manager deployment issues with restarting or resizing an existing Windows Virtual Machine in Azure](#).

Troubleshoot deploying Windows virtual machine issues in Azure

11/3/2017 • 4 min to read • [Edit Online](#)

To troubleshoot virtual machine (VM) deployment issues in Azure, review the [top issues](#) for common failures and resolutions.

If you need more help at any point in this article, you can contact the Azure experts on [the MSDN Azure and Stack Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select **Get Support**.

Top issues

The following top issues may help resolve your issue. To start troubleshooting, review these steps:

- [The cluster cannot support the requested VM size](#)
- [The cluster does not have free resources](#)

The cluster cannot support the requested VM size

- Retry the request using a smaller VM size.
- If the size of the requested VM cannot be changed:
 - Stop all the VMs in the availability set. Click **Resource groups** > your resource group > **Resources** > your availability set > **Virtual Machines** > your virtual machine > **Stop**.
 - After all the VMs stop, create the VM in the desired size.
 - Start the new VM first, and then select each of the stopped VMs and click Start.

The cluster does not have free resources

- Retry the request later.
- If the new VM can be part of a different availability set
 - Create a VM in a different availability set (in the same region).
 - Add the new VM to the same virtual network.

How can I use and deploy a windows client image into Azure?

You can use Windows 7, Windows 8, or Windows 10 in Azure for dev/test scenarios if you have an appropriate Visual Studio (formerly MSDN) subscription. This [article](#) outlines the eligibility requirements for running Windows client in Azure and uses of the Azure Gallery images.

How can I deploy a virtual machine using the Hybrid Use Benefit (HUB)?

There are a couple of different ways to deploy Windows virtual machines with the Azure Hybrid Use Benefit.

For an Enterprise Agreement subscription:

- Deploy VMs from specific Marketplace images that are pre-configured with Azure Hybrid Use Benefit.

For Enterprise agreement:

- Upload a custom VM and deploy using a Resource Manager template or Azure PowerShell.

For more information, see the following resources:

- [Azure Hybrid Use Benefit overview](#)
- [Downloadable FAQ](#)
- [Azure Hybrid Use Benefit for Windows Server and Windows Client](#).
- [How can I use the Hybrid Use Benefit in Azure](#)

How do I activate my monthly credit for Visual studio Enterprise (BizSpark)

To activate your monthly credit, see this [article](#).

How to add Enterprise Dev/Test to my Enterprise Agreement (EA) to get access to Window client images?

The ability to create subscriptions based on the Enterprise Dev/Test offer is restricted to Account Owners who have been given permission to do so by an Enterprise Administrator. The Account Owner creates subscriptions via the Azure Account Portal, and then should add active Visual Studio subscribers as co-administrators. So that they can manage and use the resources needed for development and testing. For more information, see [Enterprise Dev/Test](#).

My drivers are missing for my Windows N-Series VM

Drivers for Windows-based VMs are located [here](#).

I can't find a GPU instance within my N-Series VM

To take advantage of the GPU capabilities of Azure N-series VMs running Windows Server 2016 or Windows Server 2012 R2, you must install NVIDIA graphics drivers on each VM after deployment. Driver setup information is available for [Windows VMs](#) and [Linux VMs](#).

Are client images supported for N-Series?

Currently, Azure only supports N-Series on VMs running Windows Server and Linux operating systems.

Is N-Series VMs available in my region?

You can check the availability from the [Products available by region table](#), and pricing [here](#).

What client images can I use and deploy in Azure, and how to I get them?

You can use Windows 7, Windows 8, or Windows 10 in Azure for dev/test scenarios provided you have an appropriate Visual Studio (formerly MSDN) subscription.

- Windows 10 images are available from the Azure Gallery within [eligible dev/test offers](#).
- Visual Studio subscribers within any type of offer can also [adequately prepare and create](#) a 64-bit Windows 7, Windows 8, or Windows 10 image and then [upload to Azure](#). The use remains limited to dev/test by active Visual Studio subscribers.

This [article](#) outlines the eligibility requirements for running Windows client in Azure and use of the Azure Gallery images.

I am not able to see VM Size family that I want when resizing my VM.

When a VM is running, it is deployed to a physical server. The physical servers in Azure regions are grouped in clusters of common physical hardware. Resizing a VM that requires the VM to be moved to different hardware clusters is different depending on which deployment model was used to deploy the VM.

- VMs deployed in Classic deployment model, the cloud service deployment must be removed and redeployed to change the VMs to a size in another size family.
- VMs deployed in Resource Manager deployment model, you must stop all VMs in the availability set before changing the size of any VM in the availability set.

The listed VM size is not supported while deploying in Availability Set.

Choose a size that is supported on the availability set's cluster. It is recommended when creating an availability set to choose the largest VM size you think you need, and have that be your first deployment to the Availability set.

Can I add an existing Classic VM to an availability set?

Yes. You can add an existing classic VM to a new or existing Availability Set. For more information see [Add an existing virtual machine to an availability set](#).

Next steps

If you need more help at any point in this article, you can contact the Azure experts on [the MSDN Azure and Stack Overflow forums](#).

Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select **Get Support**.

Troubleshoot deployment issues with restarting or resizing an existing Windows VM in Azure

11/3/2017 • 2 min to read • [Edit Online](#)

When you try to start a stopped Azure Virtual Machine (VM), or resize an existing Azure VM, the common error you encounter is an allocation failure. This error results when the cluster or region either does not have resources available or cannot support the requested VM size.

If your Azure issue is not addressed in this article, visit the Azure forums on [MSDN and the Stack Overflow](#). You can post your issue in these forums, or post to [@AzureSupport on Twitter](#). You also can submit an Azure support request. To submit a support request, on the [Azure support](#) page, select **Get support**.

Collect activity logs

To start troubleshooting, collect the activity logs to identify the error associated with the issue. The following links contain detailed information on the process:

[View deployment operations](#)

[View activity logs to manage Azure resources](#)

Issue: Error when starting a stopped VM

You try to start a stopped VM but get an allocation failure.

Cause

The request to start the stopped VM has to be attempted at the original cluster that hosts the cloud service. However, the cluster does not have free space available to fulfill the request.

Resolution

- Stop all the VMs in the availability set, and then restart each VM.
 1. Click **Resource groups** > *your resource group* > **Resources** > *your availability set* > **Virtual Machines** > *your virtual machine* > **Stop**.
 2. After all the VMs stop, select each of the stopped VMs and click Start.
- Retry the restart request at a later time.

Issue: Error when resizing an existing VM

You try to resize an existing VM but get an allocation failure.

Cause

The request to resize the VM has to be attempted at the original cluster that hosts the cloud service. However, the cluster does not support the requested VM size.

Resolution

- Retry the request using a smaller VM size.
- If the size of the requested VM cannot be changed:
 1. Stop all the VMs in the availability set.

- o Click **Resource groups** > *your resource group* > **Resources** > *your availability set* > **Virtual Machines** > *your virtual machine* > **Stop**.
2. After all the VMs stop, resize the desired VM to a larger size.
 3. Select the resized VM and click **Start**, and then start each of the stopped VMs.

Next steps

If you encounter issues when you create a new Windows VM in Azure, see [Troubleshoot deployment issues with creating a new Windows virtual machine in Azure](#).

Troubleshoot access to an application running on a Windows virtual machine in Azure

9/27/2017 • 5 min to read • [Edit Online](#)

There are various reasons when you cannot start or connect to an application running on an Azure virtual machine (VM). Reasons include the application not running or listening on the expected ports, the listening port blocked, or networking rules not correctly passing traffic to the application. This article describes a methodical approach to find and correct the problem.

If you are having issues connecting to your VM using RDP or SSH, see one of the following articles first:

- [Troubleshoot Remote Desktop connections to a Windows-based Azure Virtual Machine](#)
- [Troubleshoot Secure Shell \(SSH\) connections to a Linux-based Azure virtual machine.](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

If you need more help at any point in this article, you can contact the Azure experts on [the MSDN Azure and the Stack Overflow forums](#). Alternatively, you can also file an Azure support incident. Go to the [Azure support site](#) and select **Get Support**.

Quick-start troubleshooting steps

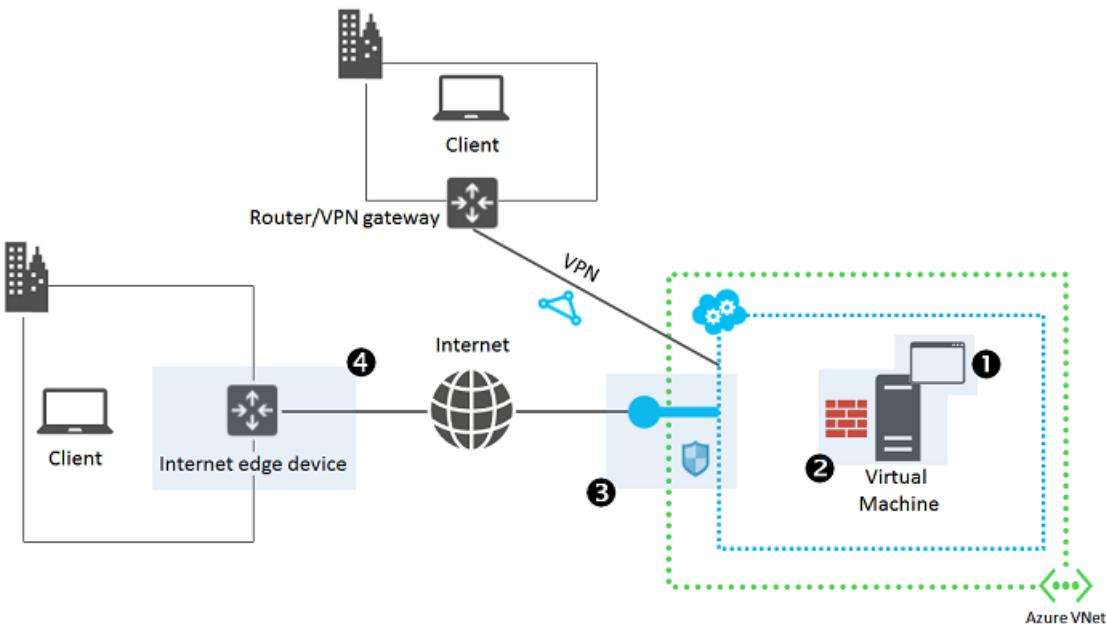
If you have problems connecting to an application, try the following general troubleshooting steps. After each step, try connecting to your application again:

- Restart the virtual machine
- Recreate the endpoint / firewall rules / network security group (NSG) rules
 - [Resource Manager model - Manage Network Security Groups](#)
 - [Classic model - Manage Cloud Services endpoints](#)
- Connect from different location, such as a different Azure virtual network
- Redeploy the virtual machine
 - [Redeploy Windows VM](#)
 - [Redeploy Linux VM](#)
- Recreate the virtual machine

For more information, see [Troubleshooting Endpoint Connectivity \(RDP/SSH/HTTP, etc. failures\)](#).

Detailed troubleshooting overview

There are four main areas to troubleshoot the access of an application that is running on an Azure virtual machine.



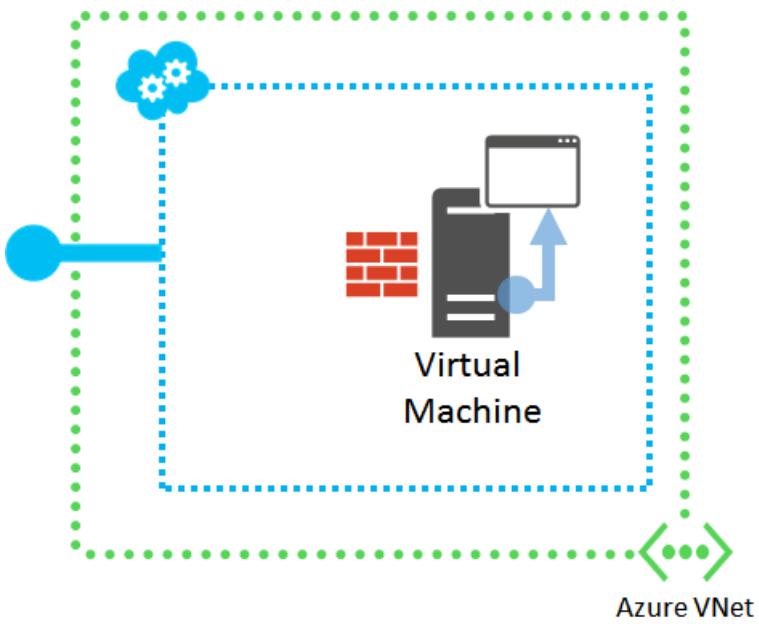
1. The application running on the Azure virtual machine.
 - Is the application itself running correctly?
2. The Azure virtual machine.
 - Is the VM itself running correctly and responding to requests?
3. Azure network endpoints.
 - Cloud service endpoints for virtual machines in the Classic deployment model.
 - Network Security Groups and inbound NAT rules for virtual machines in Resource Manager deployment model.
 - Can traffic flow from users to the VM/application on the expected ports?
4. Your Internet edge device.
 - Are firewall rules in place preventing traffic from flowing correctly?

For client computers that are accessing the application over a site-to-site VPN or ExpressRoute connection, the main areas that can cause problems are the application and the Azure virtual machine.

To determine the source of the problem and its correction, follow these steps.

Step 1: Access application from target VM

Try to access the application with the appropriate client program from the VM on which it is running. Use the local host name, the local IP address, or the loopback address (127.0.0.1).



For example, if the application is a web server, open a browser on the VM and try to access a web page hosted on the VM.

If you can access the application, go to [Step 2](#).

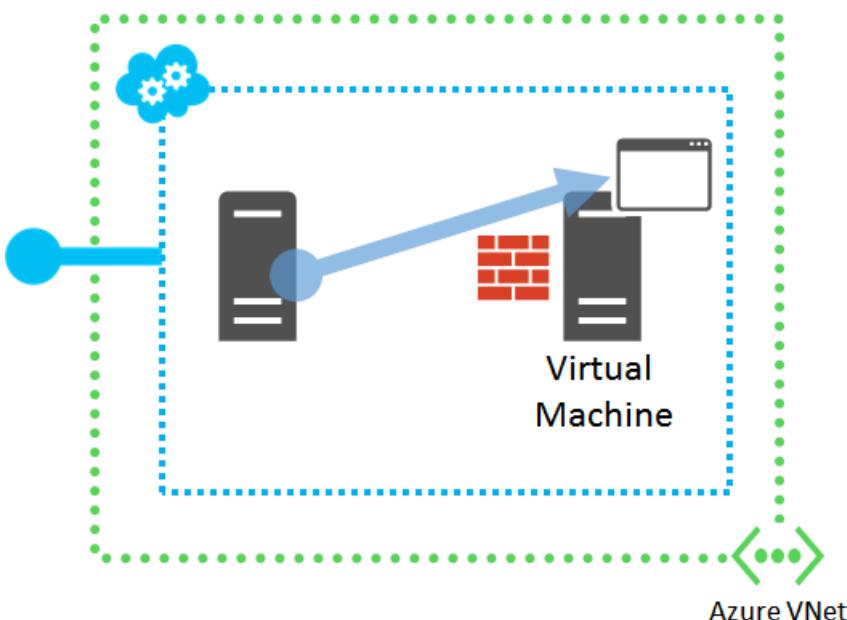
If you cannot access the application, verify the following settings:

- The application is running on the target virtual machine.
- The application is listening on the expected TCP and UDP ports.

On both Windows and Linux-based virtual machines, use the **netstat -a** command to show the active listening ports. Examine the output for the expected ports on which your application should be listening. Restart the application or configure it to use the expected ports as needed and try to access the application locally again.

Step 2: Access application from another VM in the same virtual network

Try to access the application from a different VM but in the same virtual network, using the VM's host name or its Azure-assigned public, private, or provider IP address. For virtual machines created using the classic deployment model, do not use the public IP address of the cloud service.



For example, if the application is a web server, try to access a web page from a browser on a different VM in the same virtual network.

If you can access the application, go to [Step 3](#).

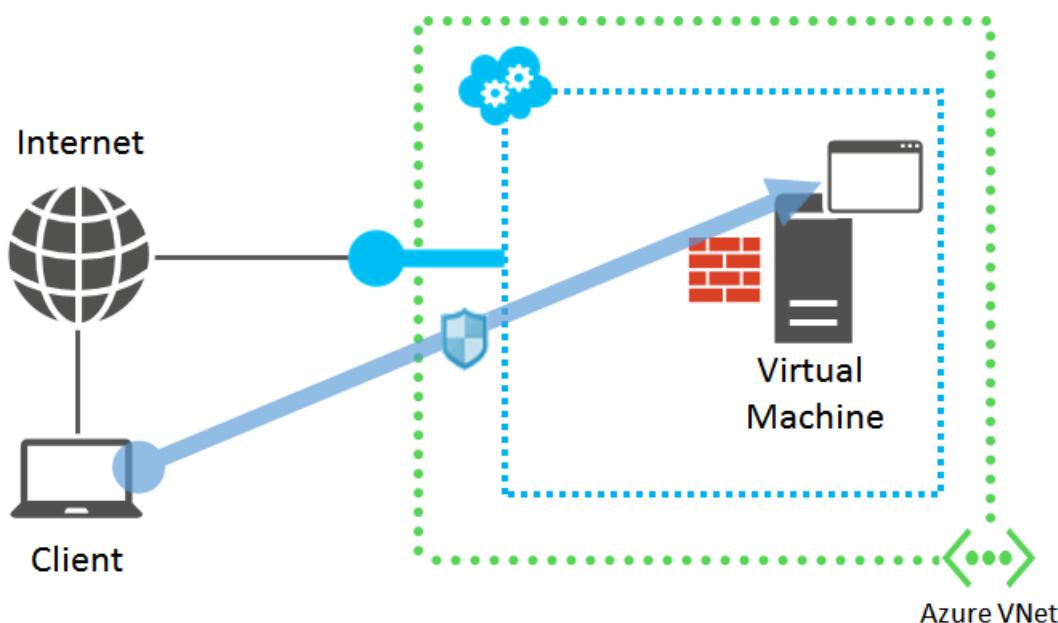
If you cannot access the application, verify the following settings:

- The host firewall on the target VM is allowing the inbound request and outbound response traffic.
- Intrusion detection or network monitoring software running on the target VM is allowing the traffic.
- Cloud Services endpoints or Network Security Groups are allowing the traffic:
 - [Classic model - Manage Cloud Services endpoints](#)
 - [Resource Manager model - Manage Network Security Groups](#)
- A separate component running in your VM in the path between the test VM and your VM, such as a load balancer or firewall, is allowing the traffic.

On a Windows-based virtual machine, use Windows Firewall with Advanced Security to determine whether the firewall rules exclude your application's inbound and outbound traffic.

Step 3: Access application from outside the virtual network

Try to access the application from a computer outside the virtual network as the VM on which the application is running. Use a different network as your original client computer.



For example, if the application is a web server, try to access the web page from a browser running on a computer that is not in the virtual network.

If you cannot access the application, verify the following settings:

- For VMs created using the classic deployment model:
 - Verify that the endpoint configuration for the VM is allowing the incoming traffic, especially the protocol (TCP or UDP) and the public and private port numbers.
 - Verify that access control lists (ACLs) on the endpoint are not preventing incoming traffic from the Internet.
 - For more information, see [How to Set Up Endpoints to a Virtual Machine](#).
- For VMs created using the Resource Manager deployment model:
 - Verify that the inbound NAT rule configuration for the VM is allowing the incoming traffic, especially the

protocol (TCP or UDP) and the public and private port numbers.

- Verify that Network Security Groups are allowing the inbound request and outbound response traffic.
- For more information, see [What is a Network Security Group \(NSG\)?](#)

If the virtual machine or endpoint is a member of a load-balanced set:

- Verify that the probe protocol (TCP or UDP) and port number are correct.
- If the probe protocol and port is different than the load-balanced set protocol and port:
 - Verify that the application is listening on the probe protocol (TCP or UDP) and port number (use **netstat -a** on the target VM).
 - Verify that the host firewall on the target VM is allowing the inbound probe request and outbound probe response traffic.

If you can access the application, ensure that your Internet edge device is allowing:

- The outbound application request traffic from your client computer to the Azure virtual machine.
- The inbound application response traffic from the Azure virtual machine.

Step 4 If you cannot access the application, use IP Verify to check the settings.

For more information, see [Azure network monitoring overview](#).

Additional resources

[Troubleshoot Remote Desktop connections to a Windows-based Azure Virtual Machine](#)

[Troubleshoot Secure Shell \(SSH\) connections to a Linux-based Azure virtual machine](#)

Troubleshoot Azure Windows virtual machine activation problems

11/3/2017 • 4 min to read • [Edit Online](#)

NOTE

Azure has two different deployment models for creating and working with resources: [Resource Manager and classic](#). This article covers using both models, but Microsoft recommends that most new deployments use the Resource Manager model.

If you have trouble when activating Azure Windows virtual machine (VM) that is created from a custom image, you can use the information provided in this document to troubleshoot the issue.

Symptom

When you try to activate an Azure Windows VM, you receive an error message resembles the following sample:

Error: 0xC004F074 The Software LicensingService reported that the computer could not be activated. No Key ManagementService (KMS) could be contacted. Please see the Application Event Log for additional information.

Cause

Generally, Azure VM activation issues occur if the Windows VM is not configured by using the appropriate KMS client setup key, or the Windows VM has a connectivity problem to the Azure KMS service (kms.core.windows.net, port 1668).

Solution

NOTE

If you are using a site-to-site VPN and forced tunneling, see [Use Azure custom routes to enable KMS activation with forced tunneling](#).

If you are using ExpressRoute and you have a default route published, see [Azure VM may fail to activate over ExpressRoute](#).

Step 1 Configure the appropriate KMS client setup key (for Windows Server 2016 and Windows Server 2012 R2)

For the VM that is created from a custom image of Windows Server 2016 or Windows Server 2012 R2, you must configure the appropriate KMS client setup key for the VM.

This step does not apply to Windows 2012 or Windows 2008 R2. It uses the Automation Virtual Machine Activation (AVMA) feature, which is supported only by Windows Server 2016 and Windows Server 2012 R2.

1. Run **slmgr.vbs /dlv** at an elevated command prompt. Check the Description value in the output, and then determine whether it was created from retail (RETAIL channel) or volume (VOLUME_KMSCLIENT) license media:

```
cscript c:\windows\system32\slmgr.vbs /dlv
```

2. If **slmgr.vbs /dlv** shows RETAIL channel, run the following commands to set the [KMS client setup key](#) for

the version of Windows Server being used, and force it to retry activation:

```
cscript c:\windows\system32\slmgr.vbs /ipk <KMS client setup key>  
cscript c:\windows\system32\slmgr.vbs /ato
```

For example, for Windows Server 2016 Datacenter, you would run the following command:

```
cscript c:\windows\system32\slmgr.vbs /ipk CB7KF-BWN84-R7R2Y-793K2-8XDDG
```

Step 2 Verify the connectivity between the VM and Azure KMS service

1. Download and extract the [Psping](#) tool to a local folder in the VM that does not activate.
2. Go to Start, search on Windows PowerShell, right-click Windows PowerShell, and then select Run as administrator.
3. Make sure that the VM is configured to use the correct Azure KMS server. To do this, run the following command:

```
iex "$env:windir\system32\cscript.exe $env:windir\system32\slmgr.vbs /skms  
kms.core.windows.net:1688
```

The command should return: Key Management Service machine name set to kms.core.windows.net:1688 successfully.

4. Verify by using Psping that you have connectivity to the KMS server. Switch to the folder where you extracted the Pstools.zip download, and then run the following:

```
\psping.exe kms.core.windows.net:1688
```

In the second-to-last line of the output, make sure that you see: Sent = 4, Received = 4, Lost = 0 (0% loss).

If Lost is greater than 0 (zero), the VM does not have connectivity to the KMS server. In this situation, if the VM is in a virtual network and has a custom DNS server specified, you must make sure that DNS server is able to resolve kms.core.windows.net. Or, change the DNS server to one that does resolve kms.core.windows.net.

Notice that if you remove all DNS servers from a virtual network, VMs use Azure's internal DNS service. This service can resolve kms.core.windows.net.

Also verify that the guest firewall has not been configured in a manner that would block activation attempts.

1. After you verify successful connectivity to kms.core.windows.net, run the following command at that elevated Windows PowerShell prompt. This command tries activation multiple times.

```
1..12 | % { iex "$env:windir\system32\cscript.exe $env:windir\system32\slmgr.vbs /ato" ; start-sleep 5 }
```

A successful activation returns information that resembles the following:

Activating Windows(R), ServerDatacenter edition (12345678-1234-1234-1234-12345678) ... Product activated successfully.

FAQ

I created the Windows Server 2016 from Azure Marketplace. Do I need to configure KMS key for activating the Windows Server 2016?

No. The image in Azure Marketplace has the appropriate KMS client setup key already configured.

Does Windows activation work the same way regardless if the VM is using Azure Hybrid Use Benefit (HUB) or not?

Yes.

What happens if Windows activation period expires?

When the grace period has expired and Windows is still not activated, Windows Server 2008 R2 and later versions of Windows will show additional notifications about activating. The desktop wallpaper remains black, and Windows Update will install security and critical updates only, but not optional updates. See the Notifications section at the bottom of the [Licensing Conditions](#) page.

Need help? Contact support.

If you still need help, [contact support](#) to get your issue resolved quickly.

Troubleshoot allocation failures when you create, restart, or resize Windows VMs in Azure

11/3/2017 • 12 min to read • [Edit Online](#)

When you create a VM, restart stopped (deallocated) VMs, or resize a VM, Microsoft Azure allocates compute resources to your subscription. You may occasionally receive errors when performing these operations -- even before you reach the Azure subscription limits. This article explains the causes of some of the common allocation failures and suggests possible remediation. The information may also be useful when you plan the deployment of your services. You can also [troubleshoot allocation failures when you create, restart, or resize Linux VMs in Azure](#).

If your Azure issue is not addressed in this article, visit the [Azure forums on MSDN and Stack Overflow](#). You can post your issue on these forums or to @AzureSupport on Twitter. Also, you can file an Azure support request by selecting **Get support** on the [Azure support](#) site.

General troubleshooting steps

Troubleshoot common allocation failures in the classic deployment model

These steps can help resolve many allocation failures in virtual machines:

- Resize the VM to a different VM size.
Click **Browse all > Virtual machines (classic)** > your virtual machine > **Settings > Size**. For detailed steps, see [Resize the virtual machine](#).
- Delete all VMs from the cloud service and re-create VMs.
Click **Browse all > Virtual machines (classic)** > your virtual machine > **Delete**. Then, click **New > Compute > [virtual machine image]**.

Troubleshoot common allocation failures in the Azure Resource Manager deployment model

These steps can help resolve many allocation failures in virtual machines:

- Stop (deallocate) all VMs in the same availability set, then restart each one.
To stop: Click **Resource groups > your resource group > Resources > your availability set > Virtual Machines > your virtual machine > Stop**.

After all VMs stop, select the first VM and click **Start**.

Background information

How allocation works

The servers in Azure datacenters are partitioned into clusters. Normally, an allocation request is attempted in multiple clusters, but it's possible that certain constraints from the allocation request force the Azure platform to attempt the request in only one cluster. In this article, we'll refer to this as "pinned to a cluster." Diagram 1 below illustrates the case of a normal allocation that is attempted in multiple clusters. Diagram 2 illustrates the case of an allocation that's pinned to Cluster 2 because that's where the existing Cloud Service CS_1 or availability set is

Diagram 1
Allocation Attempted in Multiple Clusters

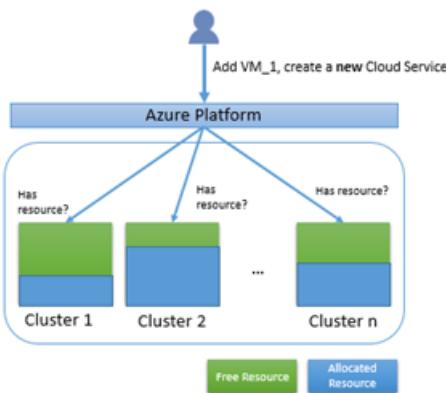
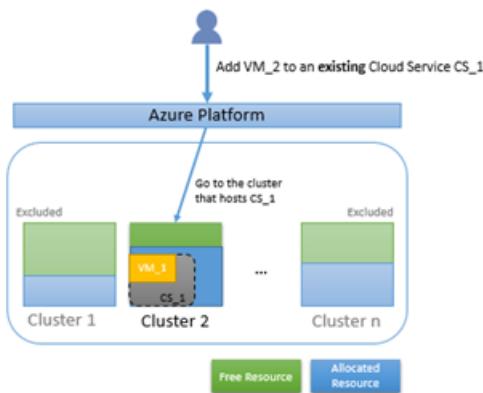


Diagram 2
Allocation Pinned to One Cluster



hosted.

Why allocation failures happen

When an allocation request is pinned to a cluster, there's a higher chance of failing to find free resources since the available resource pool is smaller. Furthermore, if your allocation request is pinned to a cluster but the type of resource you requested is not supported by that cluster, your request will fail even if the cluster has free resources. Diagram 3 below illustrates the case where a pinned allocation fails because the only candidate cluster does not have free resources. Diagram 4 illustrates the case where a pinned allocation fails because the only candidate cluster does not support the requested VM size, even though the cluster has free resources.

Diagram 3
Allocation Failed at Pinned Cluster:
No Free Resource Available

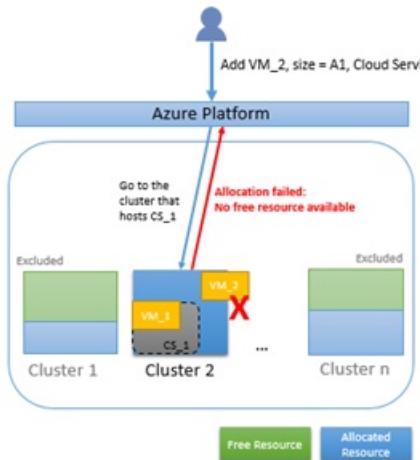
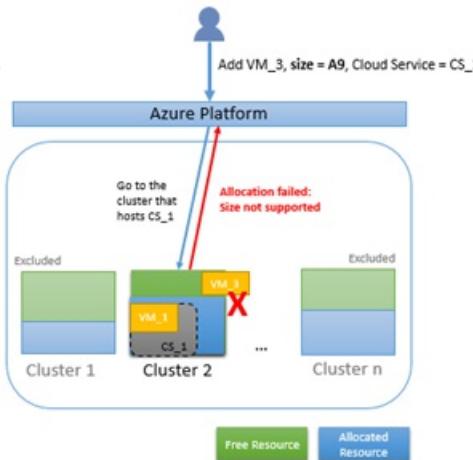


Diagram 4
Allocation Failed at Pinned Cluster:
Size not supported



Detailed troubleshoot steps specific allocation failure scenarios in the classic deployment model

Here are common allocation scenarios that cause an allocation request to be pinned. We'll dive into each scenario later in this article.

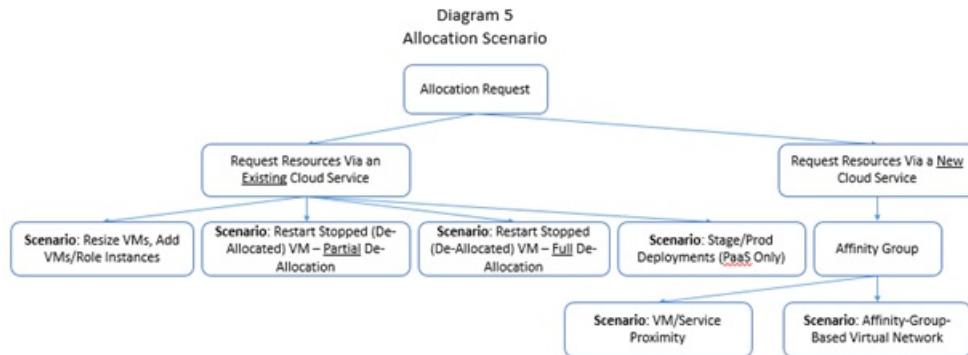
- Resize a VM or add VMs or role instances to an existing cloud service
- Restart partially stopped (deallocated) VMs
- Restart fully stopped (deallocated) VMs
- Staging/production deployments (platform as a service only)
- Affinity group (VM/service proximity)
- Affinity-group-based virtual network

When you receive an allocation error, see if any of the scenarios described apply to your error. Use the allocation error returned by the Azure platform to identify the corresponding scenario. If your request is pinned, remove some of the pinning constraints to open your request to more clusters, thereby increasing the chance of allocation success.

In general, as long as the error does not indicate "the requested VM size is not supported," you can always retry at a later time, as enough resources may have been freed in the cluster to accommodate your request. If the problem is that the requested VM size is not supported, try a different VM size. Otherwise, the only option is to remove the pinning constraint.

Two common failure scenarios are related to affinity groups. In the past, an affinity group was used to provide close proximity to VMs/service instances, or it was used to enable the creation of a virtual network. With the introduction of regional virtual networks, affinity groups are no longer required to create a virtual network. With the reduction of network latency in Azure infrastructure, the recommendation to use affinity groups for VM/service proximity has changed.

Diagram 5 below presents the taxonomy of the (pinned) allocation scenarios.



NOTE

The error listed in each allocation scenario is a short form. Refer to the [Error string lookup](#) for detailed error strings.

Allocation scenario: Resize a VM or add VMs or role instances to an existing cloud service

Error

Upgrade_VMSizeNotSupported or GeneralError

Cause of cluster pinning

A request to resize a VM or add a VM or a role instance to an existing cloud service has to be attempted at the original cluster that hosts the existing cloud service. Creating a new cloud service allows the Azure platform to find another cluster that has free resources or supports the VM size that you requested.

Workaround

If the error is Upgrade_VMSizeNotSupported*, try a different VM size. If using a different VM size is not an option, but if it's acceptable to use a different virtual IP address (VIP), create a new cloud service to host the new VM and add the new cloud service to the regional virtual network where the existing VMs are running. If your existing cloud service does not use a regional virtual network, you can still create a new virtual network for the new cloud service, and then connect your [existing virtual network to the new virtual network](#). See more about [regional virtual networks](#).

If the error is GeneralError*, it's likely that the type of resource (such as a particular VM size) is supported by the cluster, but the cluster does not have free resources at the moment. Similar to the above scenario, add the desired compute resource through creating a new cloud service (note that the new cloud service has to use a different VIP) and use a regional virtual network to connect your cloud services.

Allocation scenario: Restart partially stopped (deallocated) VMs

Error

GeneralError*

Cause of cluster pinning

Partial deallocation means that you stopped (deallocated) one or more, but not all, VMs in a cloud service. When you stop (deallocate) a VM, the associated resources are released. Restarting that stopped (deallocated) VM is therefore a new allocation request. Restarting VMs in a partially deallocated cloud service is equivalent to adding VMs to an existing cloud service. The allocation request has to be attempted at the original cluster that hosts the existing cloud service. Creating a different cloud service allows the Azure platform to find another cluster that has free resource or supports the VM size that you requested.

Workaround

If it's acceptable to use a different VIP, delete the stopped (deallocated) VMs (but keep the associated disks) and add the VMs back through a different cloud service. Use a regional virtual network to connect your cloud services:

- If your existing cloud service uses a regional virtual network, simply add the new cloud service to the same virtual network.
- If your existing cloud service does not use a regional virtual network, create a new virtual network for the new cloud service, and then [connect your existing virtual network to the new virtual network](#). See more about [regional virtual networks](#).

Allocation scenario: Restart fully stopped (deallocated) VMs

Error

GeneralError*

Cause of cluster pinning

Full deallocation means that you stopped (deallocated) all VMs from a cloud service. The allocation requests to restart these VMs have to be attempted at the original cluster that hosts the cloud service. Creating a new cloud service allows the Azure platform to find another cluster that has free resources or supports the VM size that you requested.

Workaround

If it's acceptable to use a different VIP, delete the original stopped (deallocated) VMs (but keep the associated disks) and delete the corresponding cloud service (the associated compute resources were already released when you stopped (deallocated) the VMs). Create a new cloud service to add the VMs back.

Allocation scenario: Staging/production deployments (platform as a service only)

Error

New_General* or New_VMSizeNotSupported*

Cause of cluster pinning

The staging deployment and the production deployment of a cloud service are hosted in the same cluster. When you add the second deployment, the corresponding allocation request will be attempted in the same cluster that hosts the first deployment.

Workaround

Delete the first deployment and the original cloud service and redeploy the cloud service. This action may land the first deployment in a cluster that has enough free resources to fit both deployments or in a cluster that supports the VM sizes that you requested.

Allocation scenario: Affinity group (VM/service proximity)

Error

New_General* or New_VMSizeNotSupported*

Cause of cluster pinning

Any compute resource assigned to an affinity group is tied to one cluster. New compute resource requests in that affinity group are attempted in the same cluster where the existing resources are hosted. This is true whether the new resources are created through a new cloud service or through an existing cloud service.

Workaround

If an affinity group is not necessary, do not use an affinity group, or group your compute resources into multiple affinity groups.

Allocation scenario: Affinity-group-based virtual network

Error

New_General* or New_VMSizeNotSupported*

Cause of cluster pinning

Before regional virtual networks were introduced, you were required to associate a virtual network with an affinity group. As a result, compute resources placed into an affinity group are bound by the same constraints as described in the "Allocation scenario: Affinity group (VM/service proximity)" section above. The compute resources are tied to one cluster.

Workaround

If you do not need an affinity group, create a new regional virtual network for the new resources you're adding, and then [connect your existing virtual network to the new virtual network](#). See more about [regional virtual networks](#).

Alternatively, you can [migrate your affinity-group-based virtual network to a regional virtual network](#), and then add the desired resources again.

Detailed troubleshooting steps specific allocation failure scenarios in the Azure Resource Manager deployment model

Here are common allocation scenarios that cause an allocation request to be pinned. We'll dive into each scenario later in this article.

- Resize a VM or add VMs or role instances to an existing cloud service
- Restart partially stopped (deallocated) VMs
- Restart fully stopped (deallocated) VMs

When you receive an allocation error, see if any of the scenarios described apply to your error. Use the allocation error returned by the Azure platform to identify the corresponding scenario. If your request is pinned to an existing cluster, remove some of the pinning constraints to open your request to more clusters, thereby increasing the chance of allocation success.

In general, as long as the error does not indicate "the requested VM size is not supported," you can always retry at a

later time, as enough resources may have been freed in the cluster to accommodate your request. If the problem is that the requested VM size is not supported, see below for workarounds.

Allocation scenario: Resize a VM or add VMs to an existing availability set

Error

Upgrade_VMSizeNotSupported* or GeneralError*

Cause of cluster pinning

A request to resize a VM or add a VM to an existing availability set has to be attempted at the original cluster that hosts the existing availability set. Creating a new availability set allows the Azure platform to find another cluster that has free resources or supports the VM size that you requested.

Workaround

If the error is Upgrade_VMSizeNotSupported*, try a different VM size. If using a different VM size is not an option, stop all VMs in the availability set. You can then change the size of the virtual machine that will allocate the VM to a cluster that supports the desired VM size.

If the error is GeneralError*, it's likely that the type of resource (such as a particular VM size) is supported by the cluster, but the cluster does not have free resources at the moment. If the VM can be part of a different availability set, create a new VM in a different availability set (in the same region). This new VM can then be added to the same virtual network.

Allocation scenario: Restart partially stopped (deallocated) VMs

Error

GeneralError*

Cause of cluster pinning

Partial deallocation means that you stopped (deallocated) one or more, but not all, VMs in an availability set. When you stop (deallocate) a VM, the associated resources are released. Restarting that stopped (deallocated) VM is therefore a new allocation request. Restarting VMs in a partially deallocated availability set is equivalent to adding VMs to an existing availability set. The allocation request has to be attempted at the original cluster that hosts the existing availability set.

Workaround

Stop all VMs in the availability set before restarting the first one. This will ensure that a new allocation attempt is run and that a new cluster can be selected that has available capacity.

Allocation scenario: Restart fully stopped (deallocated)

Error

GeneralError*

Cause of cluster pinning

Full deallocation means that you stopped (deallocated) all VMs in an availability set. The allocation request to restart these VMs will target all clusters that support the desired size.

Workaround

Select a new VM size to allocate. If this does not work, please try again later.

Error string lookup

New_VMSizeNotSupported*

"The VM size (or combination of VM sizes) required by this deployment cannot be provisioned due to deployment request constraints. If possible, try relaxing constraints such as virtual network bindings, deploying to a hosted service with no other deployment in it and to a different affinity group or with no affinity group, or try deploying to a different region."

New_General*

"Allocation failed; unable to satisfy constraints in request. The requested new service deployment is bound to an affinity group, or it targets a virtual network, or there is an existing deployment under this hosted service. Any of these conditions constrains the new deployment to specific Azure resources. Please retry later or try reducing the VM size or number of role instances. Alternatively, if possible, remove the aforementioned constraints or try deploying to a different region."

Upgrade_VMSizeNotSupported*

"Unable to upgrade the deployment. The requested VM size XXX may not be available in the resources supporting the existing deployment. Please try again later, try with a different VM size or smaller number of role instances, or create a deployment under an empty hosted service with a new affinity group or no affinity group binding."

GeneralError*

"The server encountered an internal error. Please retry the request." Or "Failed to produce an allocation for the service."

Redeploy Windows virtual machine to new Azure node

11/3/2017 • 1 min to read • [Edit Online](#)

If you have been facing difficulties troubleshooting Remote Desktop (RDP) connection or application access to Windows-based Azure virtual machine (VM), redeploying the VM may help. When you redeploy a VM, it moves the VM to a new node within the Azure infrastructure and then powers it back on, retaining all your configuration options and associated resources. This article shows you how to redeploy a VM using Azure PowerShell or the Azure portal.

NOTE

After you redeploy a VM, the temporary disk is lost and dynamic IP addresses associated with virtual network interface are updated.

Using Azure PowerShell

Make sure you have the latest Azure PowerShell 1.x installed on your machine. For more information, see [How to install and configure Azure PowerShell](#).

The following example deploys the VM named `myVM` in the resource group named `myResourceGroup`:

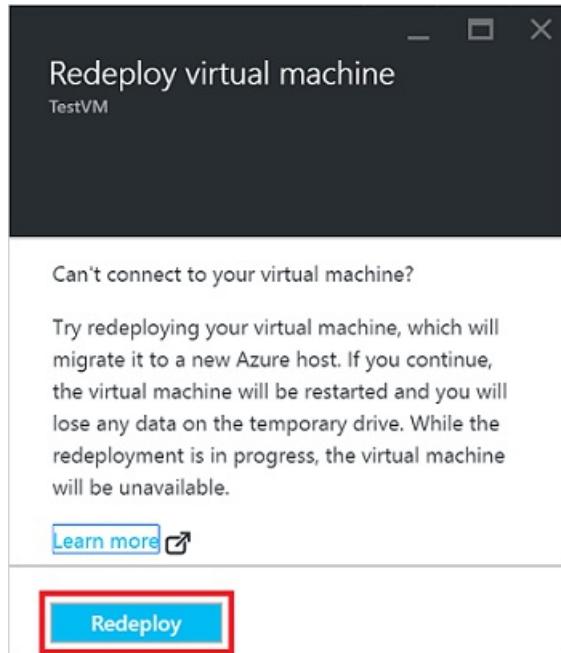
```
Set-AzureRmVM -Redeploy -ResourceGroupName "myResourceGroup" -Name "myVM"
```

Use the Azure portal

1. Select the VM you wish to redeploy, then select the *Redeploy* button in the *Settings* blade. You may need to scroll down to see the **Support and Troubleshooting** section that contains the 'Redeploy' button as in the following example:

The screenshot shows the Azure portal interface for a virtual machine named 'TestVM'. The left sidebar contains navigation links for Disks, Extensions, Network interfaces, Size, Properties, Locks, Automation script, MONITORING (Alert rules, Diagnostics, Diagram), SUPPORT + TROUBLESHOOTING (Resource health, Boot diagnostics, Reset password, Redeploy, New support request). The main content area is divided into 'Essentials' and 'Monitoring' sections. In the Essentials section, details about the resource group (TestRG), status (Running), location (West US), subscription name, computer name (TestVM), operating system (Linux), size (Standard DS1 (1 core, 3.5 GB memory)), public IP address (13.93.235.59/testv-westu-worh9mzo082g...), and virtual network/subnet (testv-westu-worh9mzo082g vnet/testv-we...) are displayed. The Monitoring section shows a chart for CPU percentage with a message 'No available data.' The 'Redeploy' button in the support section is highlighted with a red box.

2. To confirm the operation, select the *Redeploy* button:



3. The **Status** of the VM changes to *Updating* as the VM prepares to redeploy, as shown in the following example:

The screenshot shows the Azure portal interface for a virtual machine named "TestVM" located in the "TestRG" resource group. The status of the VM is currently "Updating". Other details shown include the computer name "TestVM", operating system "Linux", size "Standard D1 v2 (1 core, 3.5 GB memory)", and public IP address "40.78.108.205/<none>". The "Status" field is highlighted with a red box.

4. The **Status** then changes to *Starting* as the VM boots up on a new Azure host, as shown in the following example:

The screenshot shows the Azure portal interface for the same virtual machine "TestVM" in the "TestRG" resource group. The status of the VM has changed to "Starting". Other details shown include the computer name "TestVM", operating system "Linux", size "Standard D1 v2 (1 core, 3.5 GB memory)", and public IP address "40.78.108.205/<none>". The "Status" field is highlighted with a red box.

5. After the VM finishes the boot process, the **Status** then returns to *Running*, indicating the VM has been successfully redeployed:

The screenshot shows the Azure portal interface for a virtual machine named 'TestVM'. At the top, there are action buttons: Settings, Connect, Start, Restart, Stop, and Delete. Below this is a navigation bar with 'Essentials' and three icons: a cloud, a person, and a tag. The main content area displays the following details:

Resource group	Computer name
TestRG	TestVM
Status	Operating system
Running	Linux
Location	Size
West US	Standard D1 v2 (1 core, 3.5 GB memory)
Subscription name	Public IP address/DNS name label
	40.78.108.205/<none>
Subscription ID	Virtual network/subnet
	TestRG/default

A red box highlights the 'Status' field, which shows 'Running'. A blue button at the bottom right says 'All settings →'.

Next steps

If you are having issues connecting to your VM, you can find specific help on [troubleshooting RDP connections](#) or [detailed RDP troubleshooting steps](#). If you cannot access an application running on your VM, you can also read [application troubleshooting issues](#).

Understand common error messages when you manage Windows virtual machines in Azure

11/3/2017 • 15 min to read • [Edit Online](#)

This article describes some of the most common error codes and messages you may encounter when you create or manage Windows virtual machines (VMs) in Azure.

NOTE

You can leave comments on this page for feedback or through [Azure feedback](#) with #azerrormessage tag.

Error Response Format

Azure VMs use the following JSON format for error response:

```
{  
  "status": "status code",  
  "error": {  
    "code": "Top level error code",  
    "message": "Top level error message",  
    "details": [  
      {  
        "code": "Inner level error code",  
        "message": "Inner level error message"  
      }  
    ]  
  }  
}
```

An error response always includes a status code and an error object. Each error object always contains an error code and a message. If the VM is created with a template, the error object also contains a details section that contains an inner level of error codes and message. Normally, the most inner level of error message is the root failure.

Common virtual machine management errors

This section lists the common error messages you may encounter when managing VMs:

ERROR CODE	ERROR MESSAGE
AcquireDiskLeaseFailed	Failed to acquire lease while creating disk '{0}' using blob with URI {1}. Blob is already in use.
AllocationFailed	Allocation failed. Please try reducing the VM size or number of VMs, retry later, or try deploying to a different Availability Set or different Azure location.
AllocationFailed	The VM allocation failed due to an internal error. Please retry later or try deploying to a different location.

ERROR CODE	ERROR MESSAGE
ArtifactNotFound	The VM extension with publisher '{0}' and type '{1}' could not be found in location '{2}'.
ArtifactNotFound	Extension with publisher '{0}', type '{1}', and type handler version '{2}' could not be found in the extension repository.
ArtifactVersionNotFound	No version found in the artifact repository that satisfies the requested version '{0}'.
ArtifactVersionNotFound	No version found in the artifact repository that satisfies the requested version '{0}' for VM extension with publisher '{1}' and type '{2}'.
AttachDiskWhileBeingDetached	Cannot attach data disk '{0}' to VM '{1}' because the disk is currently being detached. Please wait until the disk is completely detached and then try again.
BadRequest	'Aligned' Availability Sets are not yet supported in this region.
BadRequest	Addition of a VM with managed disks to non-managed Availability Set or addition of a VM with blob based disks to managed Availability Set is not supported. Please create an Availability Set with 'managed' property set in order to add a VM with managed disks to it.
BadRequest	Managed Disks are not supported in this region.
BadRequest	Multiple VMExtensions per handler not supported for OS type '{0}'. VMExtension '{1}' with handler '{2}' already added or specified in input.
BadRequest	Operation '{0}' is not supported on Resource '{1}' with managed disks.
CertificateImproperlyFormatted	The secret's JSON representation retrieved from {0} has a data field which is not a properly formatted PFX file, or the password provided does not decode the PFX file correctly.
CertificateImproperlyFormatted	The data retrieved from {0} is not deserializable into JSON.
Conflict	Disk resizing is allowed only when creating a VM or when the VM is deallocated.
ConflictingUserInput	Disk '{0}' cannot be attached as the disk is already owned by VM '{1}'.
ConflictingUserInput	Source and destination resource groups are the same.
ConflictingUserInput	Source and destination storage accounts for disk {0} are different.
ContainerAlreadyOnLease	There is already a lease on the storage container holding the blob with URI {0}.

ERROR CODE	ERROR MESSAGE
CrossSubscriptionMoveWithKeyVaultResources	The Move resources request contains KeyVault resources which are referenced by one or more {0}s in the request. This is not supported currently in Cross subscription Move. Please check the error details for the KeyVault resource Ids.
DiagnosticsOperationInternalError	An internal error occurred while processing diagnostics profile of VM {0}.
DiskBlobAlreadyInUseByAnotherDisk	Blob {0} is already in use by another disk belonging to VM '{1}'. You can examine the blob metadata for the disk reference information.
DiskBlobNotFound	Unable to find VHD blob with URI {0} for disk '{1}'.
DiskBlobNotFound	Unable to find VHD blob with URI {0}.
DiskEncryptionKeySecretMissingTags	{0} secret doesn't have the {1} tags. Please update the secret version, add the required tags and retry.
DiskEncryptionKeySecretUnwrapFailed	Unwrap of secret {0} value using key {1} failed.
DiskImageNotReady	Disk image {0} is in {1} state. Please retry when image is ready.
DiskPreparationError	One or more errors occurred while preparing VM disks. See disk instance view for details.
DiskProcessingError	Disk processing halted as the VM has other disks in failed disks.
ImageBlobNotFound	Unable to find VHD blob with URI {0} for disk '{1}'.
ImageBlobNotFound	Unable to find VHD blob with URI {0}.
IncorrectDiskBlobType	Disk blobs can only be of type page blob. Blob {0} for disk '{1}' is of type block blob.
IncorrectDiskBlobType	Disk blobs can only be of type page blob. Blob {0} is of type '{1}'.
IncorrectImageBlobType	Disk blobs can only be of type page blob. Blob {0} for disk '{1}' is of type block blob.
IncorrectImageBlobType	Disk blobs can only be of type page blob. Blob {0} is of type '{1}'.
InternalOperationError	Could not resolve storage account {0}. Please ensure it was created through the Storage Resource Provider in the same location as the compute resource.
InternalOperationError	{0} goal seeking tasks failed.
InternalOperationError	Error occurred in validating the network profile of VM '{0}'.

ERROR CODE	ERROR MESSAGE
InvalidAccountType	The AccountType {0} is invalid.
InvalidParameter	The value of parameter {0} is invalid.
InvalidParameter	The Admin password specified is not allowed.
InvalidParameter	"The supplied password must be between {0}-{1} characters long and must satisfy at least {2} of password complexity requirements from the following: 1. Contains an uppercase character 2. Contains a lowercase character 3. Contains a numeric digit 4. Contains a special character.
InvalidParameter	The Admin Username specified is not allowed.
InvalidParameter	Cannot attach an existing OS disk if the VM is created from a platform or user image.
InvalidParameter	Container name {0} is invalid. Container names must be 3-63 characters in length and may contain only lower-case alphanumeric characters and hyphen. Hyphen must be preceded and followed by an alphanumeric character.
InvalidParameter	Container name {0} in URL {1} is invalid. Container names must be 3-63 characters in length and may contain only lower-case alphanumeric characters and hyphen. Hyphen must be preceded and followed by an alphanumeric character.
InvalidParameter	The blob name in URL {0} contains a slash. This is presently not supported for disks.
InvalidParameter	The URI {0} does not look to be correct blob URI.
InvalidParameter	A disk named '{0}' already uses the same LUN: {1}.
InvalidParameter	A disk named '{0}' already exists.
InvalidParameter	Cannot specify user image overrides for a disk already defined in the specified image reference.
InvalidParameter	A disk named '{0}' already uses the same VHD URL {1}.
InvalidParameter	The specified fault domain count {0} must fall in the range {1} to {2}.
InvalidParameter	The license type {0} is invalid. Valid license types are: Windows_Client or Windows_Server, case sensitive.
InvalidParameter	Linux host name cannot exceed {0} characters in length or contain the following characters: {1}.

ERROR CODE	ERROR MESSAGE
InvalidParameter	Destination path for Ssh public keys is currently limited to its default value {0} due to a known issue in Linux provisioning agent.
InvalidParameter	A disk at LUN {0} already exists.
InvalidParameter	Subscription {0} of the request must match the subscription {1} contained in the managed disk id.
InvalidParameter	Custom data in OSProfile must be in Base64 encoding and with a maximum length of {0} characters.
InvalidParameter	Blob name in URL {0} must end with '{1}' extension.
InvalidParameter	{0}' is not a valid captured VHD blob name prefix. A valid prefix matches regex '{1}'.
InvalidParameter	Certificates cannot be added to your VM if the VM agent is not provisioned.
InvalidParameter	A disk at LUN {0} already exists.
InvalidParameter	Unable to create the VM because the requested size {0} is not available in the cluster where the availability set is currently allocated. The available sizes are: {1}. Read more on VM resizing strategy at https://aka.ms/azure-resizevm .
InvalidParameter	The requested VM size {0} is not available in the current region. The sizes available in the current region are: {1}. Find out more on the available VM sizes in each region at https://aka.ms/azure-regions .
InvalidParameter	The requested VM size {0} is not available in the current region. Find out more on the available VM sizes in each region at https://aka.ms/azure-regions .
InvalidParameter	Windows admin user name cannot be more than {0} characters long, end with a period(.), or contain the following characters: {1}.
InvalidParameter	Windows computer name cannot be more than {0} characters long, be entirely numeric, or contain the following characters: {1}.
MissingMoveDependentResources	The move resources request does not contain all the dependent resources. Please check error details for missing resource ids.
MoveResourcesHaveInvalidState	The Move Resources request contains VMs which are associated with invalid storage accounts. Please check details for these resource ids and referenced storage account names.

ERROR CODE	ERROR MESSAGE
MoveResourcesHavePendingOperations	The move resources request contains resources for which an operation is pending. Please check details for these resource ids. Retry your operation once the pending operations complete.
MoveResourcesNotFound	The move resources request contains resources that cannot be found. Please check details for these resource ids.
NetworkingInternalOperationError	Unknown network allocation error.
NetworkingInternalOperationError	Unknown network allocation error
NetworkingInternalOperationError	An internal error occurred in processing network profile of the VM.
NotFound	The Availability Set {0} cannot be found.
NotFound	Source Virtual Machine '{0}' specified in the request does not exist in this Azure location.
NotFound	Tenant with id {0} not found.
NotFound	The Image {0} cannot be found.
NotSupported	The license type is {0}, but the image blob {1} is not from on-premises.
OperationNotAllowed	Availability Set {0} cannot be deleted. Before deleting an Availability Set please ensure that it does not contain any VM.
OperationNotAllowed	Changing availability set SKU from 'Aligned' to 'Classic' is not allowed.
OperationNotAllowed	Cannot modify extensions in the VM when the VM is not running.
OperationNotAllowed	The Capture action is only supported on a Virtual Machine with blob based disks. Please use the 'Image' resource APIs to create an Image from a managed Virtual Machine.
OperationNotAllowed	The resource {0} cannot be created from Image {1} until Image has been successfully created.
OperationNotAllowed	Updates to encryptionSettings is not allowed when VM is allocated, Please retry after VM is deallocated
OperationNotAllowed	Addition of a managed disk to a VM with blob based disks is not supported.
OperationNotAllowed	The maximum number of data disks allowed to be attached to a VM of this size is {0}.

ERROR CODE	ERROR MESSAGE
OperationNotAllowed	Addition of a blob based disk to VM with managed disks is not supported.
OperationNotAllowed	Operation '{0}' is not allowed on Image '{1}' since the Image is marked for deletion. You can only retry the Delete operation (or wait for an ongoing one to complete).
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since the VM is generalized.
OperationNotAllowed	Operation '{0}' is not allowed as Restore point collection '{1}' is marked for deletion.
OperationNotAllowed	Operation '{0}' is not allowed on VM extension '{1}' since it is marked for deletion. You can only retry the Delete operation (or wait for an ongoing one to complete).
OperationNotAllowed	Operation '{0}' is not allowed since the Virtual Machines '{1}' are being provisioned using the Image '{2}'.
OperationNotAllowed	Operation '{0}' is not allowed since the Virtual Machine ScaleSet '{1}' is currently using the Image '{2}'.
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since the VM is marked for deletion. You can only retry the Delete operation (or wait for an ongoing one to complete).
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since the VM is either deallocated or marked to be deallocated.
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since the VM is running. Please power off explicitly in case you shut down the VM from inside the guest operating system.
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since the VM is not deallocated.
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since VM has extension '{2}' in failed state.
OperationNotAllowed	Operation '{0}' is not allowed on VM '{1}' since another operation is in progress.
OperationNotAllowed	The operation '{0}' requires the Virtual Machine '{1}' to be Generalized.
OperationNotAllowed	The operation requires the VM to be running (or set to run).
OperationNotAllowed	Disk with size {0}GB, which is smaller than the size {1}GB of corresponding disk in Image, is not allowed.
OperationNotAllowed	VM Scale Set extensions of handler '{0}' can be added only at the time of VM Scale Set creation.

ERROR CODE	ERROR MESSAGE
OperationNotAllowed	VM Scale Set extensions of handler '{0}' can be deleted only at the time of VM Scale Set deletion.
OperationNotAllowed	VM '{0}' is already using managed disks.
OperationNotAllowed	VM '{0}' belongs to 'Classic' availability set '{1}'. Please update the availability set to use 'Aligned' SKU and then retry the Conversion.
OperationNotAllowed	VM created from Image cannot have blob based disks. All disks have to be managed disks.
OperationNotAllowed	Capture operation cannot be completed because the VM is not generalized.
OperationNotAllowed	Management operations on VM '{0}' are disallowed because VM disks are being converted to managed disks.
OperationNotAllowed	An ongoing operation is changing power state of Virtual Machine {0} to {1}. Please perform operation {2} after some time.
OperationNotAllowed	Unable to add or update the VM. The requested VM size {0} may not be available in the existing allocation unit. Read more on VM resizing strategy at https://aka.ms/azure-resizevm .
OperationNotAllowed	Unable to resize the VM because the requested size {0} is not available in the cluster where the availability set is currently allocated. The available sizes are: {1}. Read more on VM resizing strategy at https://aka.ms/azure-resizevm .
OperationNotAllowed	Unable to resize the VM because the requested size {0} is not available in the cluster where the VM is currently allocated. To resize your VM to {1} please deallocate (this is Stop operation in the Azure portal) and try the resize operation again. Read more on VM resizing strategy at https://aka.ms/azure-resizevm .
OSProvisioningClientError	OS Provisioning failed for VM '{0}' because the guest OS is currently being provisioned.
OSProvisioningClientError	OS provisioning for VM '{0}' failed. Error details: {1} Make sure the image has been properly prepared (generalized). <ul style="list-style-type: none"> • Instructions for Windows: https://azure.microsoft.com/documentation/articles/virtual-machines-windows-upload-image/
OSProvisioningClientError	SSH host key generation failed. Error details: {0}. To resolve this issue verify if Linux agent is set up properly. <ul style="list-style-type: none"> • You can check the instructions at : https://azure.microsoft.com/documentation/articles/virtual-machines-linux-agent-user-guide/

Error code	Error message
OSProvisioningClientError	Username specified for the VM is invalid for this Linux distribution. Error details: {0}.
OSProvisioningInternalError	OS Provisioning failed for VM '{0}' due to an internal error.
OSProvisioningTimedOut	OS Provisioning for VM '{0}' did not finish in the allotted time. The VM may still finish provisioning successfully. Please check provisioning state later.
OSProvisioningTimedOut	<p>OS Provisioning for VM '{0}' did not finish in the allotted time. The VM may still finish provisioning successfully. Please check provisioning state later. Also, make sure the image has been properly prepared (generalized).</p> <ul style="list-style-type: none"> • Instructions for Windows: https://azure.microsoft.com/documentation/articles/virtual-machines-windows-upload-image/ • Instructions for Linux: https://azure.microsoft.com/documentation/articles/virtual-machines-linux-capture-image/
OSProvisioningTimedOut	<p>OS Provisioning for VM '{0}' did not finish in the allotted time. However, the VM guest agent was detected running. This suggests the guest OS has not been properly prepared to be used as a VM image (with CreateOption=FromImage). To resolve this issue, either use the VHD as is with CreateOption=Attach or prepare it properly for use as an image:</p> <ul style="list-style-type: none"> • Instructions for Windows: https://azure.microsoft.com/documentation/articles/virtual-machines-windows-upload-image/ • Instructions for Linux: https://azure.microsoft.com/documentation/articles/virtual-machines-linux-capture-image/
OverConstrainedAllocationRequest	The required VM size is not currently available in the selected location.
ResourceUpdateBlockedOnPlatformUpdate	Resource cannot be updated at this time due to ongoing platform update. Please try again later.
StorageAccountLimitation	Storage account '{0}' does not support page blobs which are required to create disks.
StorageAccountLimitation	Storage account '{0}' has exceeded its allocated quota.
StorageAccountLocationMismatch	Could not resolve storage account {0}. Please ensure it was created through the Storage Resource Provider in the same location as the compute resource.
StorageAccountNotFound	Storage account {0} not found. Ensure storage account is not deleted and belongs to the same Azure location as the VM.
StorageAccountNotRecognized	Please use a storage account managed by Storage Resource Provider. Use of {0} is not supported.

ERROR CODE	ERROR MESSAGE
StorageAccountOperationInternalError	Internal error occurred while accessing storage account {0}.
StorageAccountSubscriptionMismatch	Storage account {0} doesn't belong to subscription {1}.
StorageAccountTooBusy	Storage account '{0}' is too busy currently. Consider using another account.
StorageAccountTypeNotSupported	Disk {0} uses {1} which is a Blob storage account. Please retry with General purpose storage account.
StorageAccountTypeNotSupported	Storage account {0} is of {1} type. Boot Diagnostics supports {2} storage account types.
SubscriptionNotAuthorizedForImage	The subscription is not authorized.
TargetDiskBlobAlreadyExists	Blob {0} already exists. Please provide a different blob URI to create a new blank data disk '{1}'.
TargetDiskBlobAlreadyExists	Capture operation cannot continue because target image blob {0} already exists and the flag to overwrite VHD blobs is not set. Either delete the blob or set the flag to overwrite VHD blobs and retry.
TargetDiskBlobAlreadyExists	Capture operation cannot continue because target image blob {0} has an active lease on it.
TargetDiskBlobAlreadyExists	Blob {0} already exists. Please provide a different blob URI as target for disk '{1}'.
TooManyVMRedeploymentRequests	Too many redeployment requests have been received for VM '{0}' or the VMs in the same availabilityset with this VM. Please retry later.
VHDSizeInvalid	The specified disk size value of {0} for disk '{1}' with blob {2} is invalid. Disk size must be between {3} and {4}.
VMAgentStatusCommunicationError	VM '{0}' has not reported status for VM agent or extensions. Please verify the VM has a running VM agent, and can establish outbound connections to Azure storage.
VMArtifactRepositoryInternalError	An error occurred while communicating with the artifact repository to retrieve VM artifact details.
VMArtifactRepositoryInternalError	An internal error occurred while retrieving the VM artifact data from the artifact repository.
VMExtensionHandlerNonTransientError	Handler '{0}' has reported failure for VM Extension '{1}' with terminal error code '{2}' and error message: '{3}'
VMExtensionManagementInternalError	Internal error occurred while processing VM extension '{0}'.
VMExtensionManagementInternalError	Multiple errors occurred while preparing the VM extensions. See VM extension instance view for details.

ERROR CODE	ERROR MESSAGE
VMExtensionProvisioningError	VM has reported a failure when processing extension '{0}'. Error message: "{1}".
VMExtensionProvisioningError	Multiple VM extensions failed to be provisioned on the VM. Please see the VM extension instance view for details.
VMExtensionProvisioningTimeout	Provisioning of VM extension '{0}' has timed out. Extension installation may be taking too long, or extension status could not be obtained.
VMMarketplaceInvalidInput	Creating a virtual machine from a non Marketplace image does not need Plan information, please remove the Plan information in the request. OS disk name is {0}.
VMMarketplaceInvalidInput	The purchase information does not match. Unable to deploy from the Marketplace image. OS disk name is {0}.
VMMarketplaceInvalidInput	Creating a virtual machine from Marketplace image requires Plan information in the request. OS disk name is {0}.
VMNotFound	The VM '{0}' cannot be found.
VMRedeploymentFailed	VM '{0}' redeployment failed due to an internal error. Please retry later.
VMRedeploymentTimedOut	Redeployment of VM '{0}' didn't finish in the allotted time. It might finish successfully in sometime. Else, you can retry the request.
VMStartTimedOut	VM '{0}' did not start in the allotted time. The VM may still start successfully. Please check the power state later.

Next steps

If you need more help, you can contact the Azure experts on [the MSDN Azure and Stack Overflow forums](#).

Alternatively, you can file an Azure support incident. Go to the [Azure support site](#) and select **Get Support**.

Troubleshoot a Windows VM by attaching the OS disk to a recovery VM using Azure PowerShell

11/3/2017 • 6 min to read • [Edit Online](#)

If your Windows virtual machine (VM) in Azure encounters a boot or disk error, you may need to perform troubleshooting steps on the virtual hard disk itself. A common example would be a failed application update that prevents the VM from being able to boot successfully. This article details how to use Azure PowerShell to connect your virtual hard disk to another Windows VM to fix any errors, then re-create your original VM.

Recovery process overview

The troubleshooting process is as follows:

1. Delete the VM encountering issues, keeping the virtual hard disks.
2. Attach and mount the virtual hard disk to another Windows VM for troubleshooting purposes.
3. Connect to the troubleshooting VM. Edit files or run any tools to fix issues on the original virtual hard disk.
4. Unmount and detach the virtual hard disk from the troubleshooting VM.
5. Create a VM using the original virtual hard disk.

Make sure that you have [the latest Azure PowerShell](#) installed and logged in to your subscription:

```
Login-AzureRMAccount
```

In the following examples, replace parameter names with your own values. Example parameter names include `myResourceGroup`, `mystorageaccount`, and `myVM`.

Determine boot issues

You can view a screenshot of your VM in Azure to help troubleshoot boot issues. This screenshot can help identify why a VM fails to boot. The following example gets the screenshot from the Windows VM named `myVM` in the resource group named `myResourceGroup`:

```
Get-AzureRmVMBootDiagnosticsData -ResourceGroupName myResourceGroup `  
-Name myVM -Windows -LocalPath C:\Users\ops\
```

Review the screenshot to determine why the VM is failing to boot. Note any specific error messages or error codes provided.

View existing virtual hard disk details

Before you can attach your virtual hard disk to another VM, you need to identify the name of the virtual hard disk (VHD).

The following example gets information for the VM named `myVM` in the resource group named `myResourceGroup`:

```
Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVM"
```

Look for `Vhd Uri` within the `StorageProfile` section from the output of the preceding command. The following

truncated example output shows the `Vhd Uri` towards the end of the code block:

```
RequestId : 8a134642-2f01-4e08-bb12-d89b5b81a0a0
StatusCode : OK
ResourceGroupName : myResourceGroup
Id :
/subscriptions/guid/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM
Name : myVM
Type : Microsoft.Compute/virtualMachines
...
StorageProfile :
  ImageReference :
    Publisher : MicrosoftWindowsServer
    Offer : WindowsServer
    Sku : 2016-Datacenter
    Version : latest
  OsDisk :
    OsType : Windows
    Name : myVM
    Vhd :
      Uri : https://mystorageaccount.blob.core.windows.net/vhds/myVM.vhd
    Caching : ReadWrite
    CreateOption : FromImage
```

Delete existing VM

Virtual hard disks and VMs are two distinct resources in Azure. A virtual hard disk is where the operating system itself, applications, and configurations are stored. The VM itself is just metadata that defines the size or location, and references resources such as a virtual hard disk or virtual network interface card (NIC). Each virtual hard disk has a lease assigned when attached to a VM. Although data disks can be attached and detached even while the VM is running, the OS disk cannot be detached unless the VM resource is deleted. The lease continues to associate the OS disk with a VM even when that VM is in a stopped and deallocated state.

The first step to recover your VM is to delete the VM resource itself. Deleting the VM leaves the virtual hard disks in your storage account. After the VM is deleted, you attach the virtual hard disk to another VM to troubleshoot and resolve the errors.

The following example deletes the VM named `myVM` from the resource group named `myResourceGroup` :

```
Remove-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVM"
```

Wait until the VM has finished deleting before you attach the virtual hard disk to another VM. The lease on the virtual hard disk that associates it with the VM needs to be released before you can attach the virtual hard disk to another VM.

Attach existing virtual hard disk to another VM

For the next few steps, you use another VM for troubleshooting purposes. You attach the existing virtual hard disk to this troubleshooting VM to browse and edit the disk's content. This process allows you to correct any configuration errors or review additional application or system log files, for example. Choose or create another VM to use for troubleshooting purposes.

When you attach the existing virtual hard disk, specify the URL to the disk obtained in the preceding `Get-AzureRmVM` command. The following example attaches an existing virtual hard disk to the troubleshooting VM named `myVMRecovery` in the resource group named `myResourceGroup` :

```
$myVM = Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVMRecovery"
Add-AzureRmVMDisk -VM $myVM -CreateOption "Attach" -Name "DataDisk" -DiskSizeInGB $null ` 
    -VhdUri "https://mystorageaccount.blob.core.windows.net/vhds/myVM.vhd"
Update-AzureRmVM -ResourceGroup "myResourceGroup" -VM $myVM
```

NOTE

Adding a disk requires you to specify the size of the disk. As we attach an existing disk, the `-DiskSizeInGB` is specified as `$null`. This value ensures the data disk is correctly attached, and without the need to determine the true size of data disk.

Mount the attached data disk

1. RDP to your troubleshooting VM using the appropriate credentials. The following example downloads the RDP connection file for the VM named `myVMRecovery` in the resource group named `myResourceGroup`, and downloads it to `C:\Users\ops\Documents`

```
Get-AzureRMRemoteDesktopFile -ResourceGroupName "myResourceGroup" -Name "myVMRecovery" ` 
    -LocalPath "C:\Users\ops\Documents\myVMRecovery.rdp"
```

2. The data disk is automatically detected and attached. View the list of attached volumes to determine the drive letter as follows:

```
Get-Disk
```

The following example output shows the virtual hard disk connected a disk **2**. (You can also use `Get-Volume` to view the drive letter):

Number	Friendly Name	Serial Number	HealthStatus	OperationalStatus	Total Size	Partition Style
0	Virtual HD		Healthy	Online	127 GB	MBR
1	Virtual HD		Healthy	Online	50 GB	MBR
2	Msft Virtu...		Healthy	Online	127 GB	MBR

Fix issues on original virtual hard disk

With the existing virtual hard disk mounted, you can now perform any maintenance and troubleshooting steps as needed. Once you have addressed the issues, continue with the following steps.

Unmount and detach original virtual hard disk

Once your errors are resolved, you unmount and detach the existing virtual hard disk from your troubleshooting VM. You cannot use your virtual hard disk with any other VM until the lease attaching the virtual hard disk to the troubleshooting VM is released.

1. From within your RDP session, unmount the data disk on your recovery VM. You need the disk number from the previous `Get-Disk` cmdlet. Then, use `Set-Disk` to set the disk as offline:

```
Set-Disk -Number 2 -IsOffline $True
```

Confirm the disk is now set as offline using `Get-Disk` again. The following example output shows the disk is

now set as offline:

Number	Friendly Name	Serial Number	HealthStatus	OperationalStatus	Total Size	Partition Style
0	Virtual HD		Healthy	Online	127 GB MBR	
1	Virtual HD		Healthy	Online	50 GB MBR	
2	Msft Virtu...		Healthy	Offline	127 GB MBR	

2. Exit your RDP session. From your Azure PowerShell session, remove the virtual hard disk from the troubleshooting VM.

```
$myVM = Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVMRecovery"  
Remove-AzureRmVMDataDisk -VM $myVM -Name "DataDisk"  
Update-AzureRmVM -ResourceGroup "myResourceGroup" -VM $myVM
```

Create VM from original hard disk

To create a VM from your original virtual hard disk, use [this Azure Resource Manager template](#). The actual JSON template is at the following link:

- <https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/201-vm-specialized-vhd-existing-vnet/azuredeploy.json>

The template deploys a VM into an existing virtual network, using the VHD URL from the earlier command. The following example deploys the template to the resource group named `myResourceGroup`:

```
New-AzureRmResourceGroupDeployment -Name myDeployment -ResourceGroupName myResourceGroup  
-TemplateUri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/201-vm-specialized-vhd-existing-vnet/azuredeploy.json
```

Answer the prompts for the template such as VM name, OS type, and VM size. The `osDiskVhdUri` is the same as previously used when attaching the existing virtual hard disk to the troubleshooting VM.

Re-enable boot diagnostics

When you create your VM from the existing virtual hard disk, boot diagnostics may not automatically be enabled. The following example enables the diagnostic extension on the VM named `myVMDeployed` in the resource group named `myResourceGroup`:

```
$myVM = Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVMDeployed"  
Set-AzureRmVMBootDiagnostics -ResourceGroupName myResourceGroup -VM $myVM -enable  
Update-AzureRmVM -ResourceGroup "myResourceGroup" -VM $myVM
```

Next steps

If you are having issues connecting to your VM, see [Troubleshoot RDP connections to an Azure VM](#). For issues with accessing applications running on your VM, see [Troubleshoot application connectivity issues on a Windows VM](#).

For more information about using Resource Manager, see [Azure Resource Manager overview](#).

Troubleshoot a Windows VM by attaching the OS disk to a recovery VM using the Azure portal

11/3/2017 • 5 min to read • [Edit Online](#)

If your Windows virtual machine (VM) in Azure encounters a boot or disk error, you may need to perform troubleshooting steps on the virtual hard disk itself. A common example would be a failed application update that prevents the VM from being able to boot successfully. This article details how to use the Azure portal to connect your virtual hard disk to another Windows VM to fix any errors, then re-create your original VM.

Recovery process overview

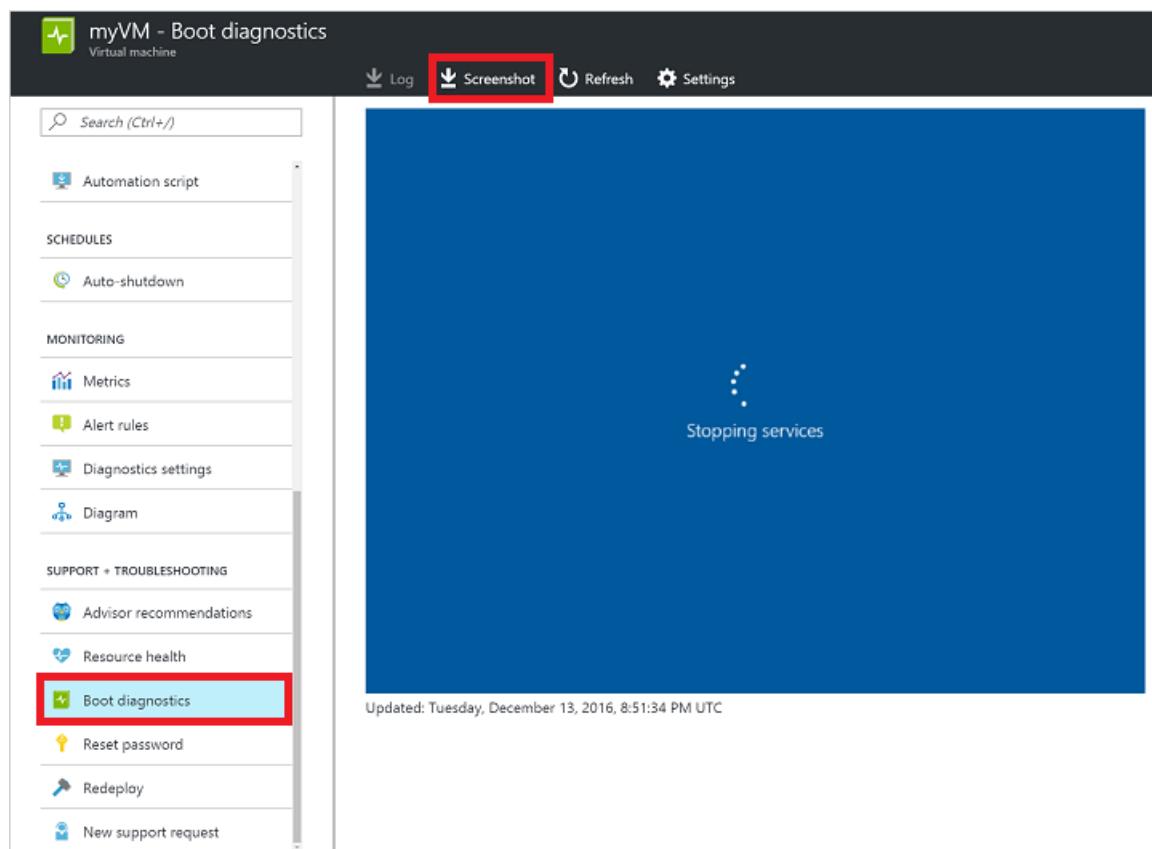
The troubleshooting process is as follows:

1. Delete the VM encountering issues, keeping the virtual hard disks.
2. Attach and mount the virtual hard disk to another Windows VM for troubleshooting purposes.
3. Connect to the troubleshooting VM. Edit files or run any tools to fix issues on the original virtual hard disk.
4. Unmount and detach the virtual hard disk from the troubleshooting VM.
5. Create a VM using the original virtual hard disk.

Determine boot issues

To determine why your VM is not able to boot correctly, examine the boot diagnostics VM screenshot. A common example would be a failed application update, or an underlying virtual hard disk being deleted or moved.

Select your VM in the portal and then scroll down to the **Support + Troubleshooting** section. Click **Boot diagnostics** to view the screenshot. Note any specific error messages or error codes to help determine why the VM is encountering an issue. The following example shows a VM waiting on stopping services:



You can also click **Screenshot** to download a capture of the VM screenshot.

View existing virtual hard disk details

Before you can attach your virtual hard disk to another VM, you need to identify the name of the virtual hard disk (VHD).

Select your resource group from the portal, then select your storage account. Click **Blobs**, as in the following example:

The screenshot shows the 'Essentials' section of the storage account settings. It includes details like Resource group (myresourcegroup), Status (Available), Location (West US), and Subscription information. Below this is a 'Services' section with icons for Blobs, Files, Tables, and Queues. The 'Blobs' icon is highlighted with a red box.

Typically you have a container named **vhds** that stores your virtual hard disks. Select the container to view a list of virtual hard disks. Note the name of your VHD (the prefix is usually the name of your VM):

The screenshot shows the 'vhds' container list. It displays two blobs: 'myDisk.vhd' and 'myVM2016101103813.vhd'. Both blobs are highlighted with a red box. The left pane shows the storage account details and a list of containers, with 'vhds' also highlighted with a red box.

Select your existing virtual hard disk from the list and copy the URL for use in the following steps:

The screenshot shows two windows side-by-side. The left window is titled 'vhds' and shows a list of virtual hard disk files. The right window is titled 'Blob properties' and shows details for a specific blob named 'myVM2016101103813.vhd'. The 'URL' field is highlighted with a red box.

NAME	MODIFIED	BLOB TYPE	SIZE
myDisk.vhd	11/1/2016, 11:34:38 AM	Page blob	1.1 TB
myVM2016101103813.vhd	11/1/2016, 12:04:50 PM	Page blob	31.46 GB
myVMRecovery.a49961a9-41fe-... ...	11/1/2016, 2:31:25 PM	Block blob	246 8
myVMRecovery201610111441.vhd	11/1/2016, 2:27:29 PM	Page blob	31.46 GB

Blob properties

NAME
myVM2016101103813.vhd

URL
<https://mystorageaccountikf.blob.core.windows.net/>

LAST MODIFIED
11/1/2016, 12:04:50 PM

TYPE
Page blob

SIZE
31.46 GB

Delete existing VM

Virtual hard disks and VMs are two distinct resources in Azure. A virtual hard disk is where the operating system itself, applications, and configurations are stored. The VM itself is just metadata that defines the size or location, and references resources such as a virtual hard disk or virtual network interface card (NIC). Each virtual hard disk has a lease assigned when attached to a VM. Although data disks can be attached and detached even while the VM is running, the OS disk cannot be detached unless the VM resource is deleted. The lease continues to associate the OS disk with a VM even when that VM is in a stopped and deallocated state.

The first step to recover your VM is to delete the VM resource itself. Deleting the VM leaves the virtual hard disks in your storage account. After the VM is deleted, you attach the virtual hard disk to another VM to troubleshoot and resolve the errors.

Select your VM in the portal, then click **Delete**:

The screenshot shows the Azure portal's VM details page for 'myVM'. The 'Delete' button is highlighted with a red box. The page displays various VM settings and resource group information.

myVM
Virtual machine

Essentials

Resource group: **myResourceGroup**
Status: **Running**
Location: **West US**
Subscription name:
Subscription ID:

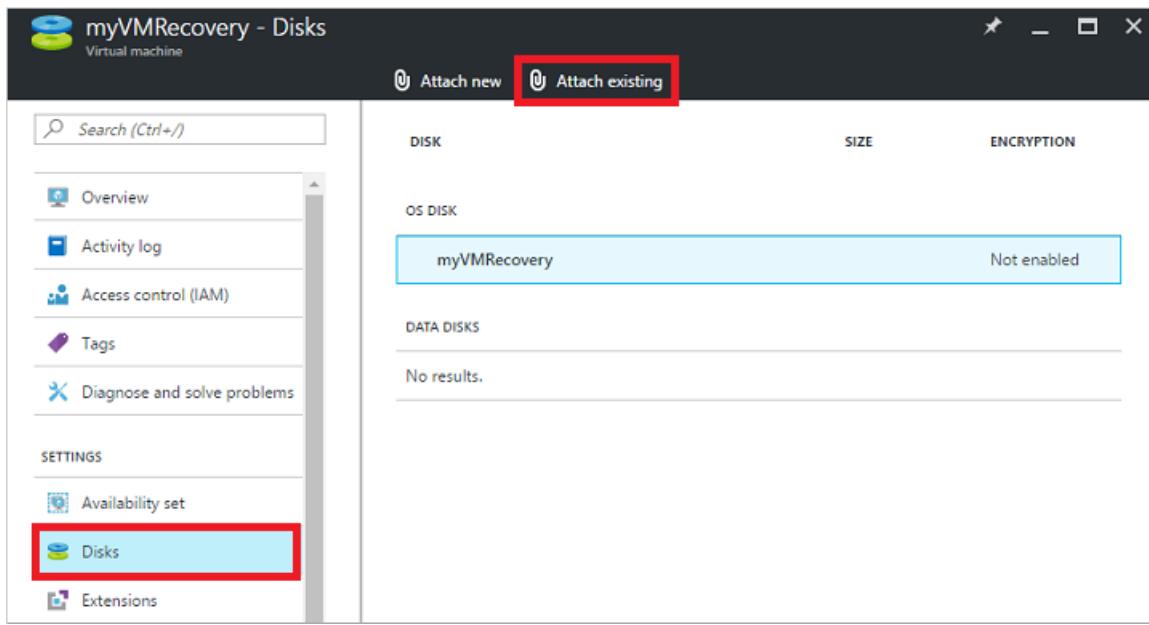
Computer name: **myVM**
Operating system: **Windows**
Size: **Standard D1 v2 (1 core, 3.5 GB memory)**
Public IP address/DNS name label: **40.78.106.206/<none>**
Virtual network/subnet: **myVnet/mySubnet**

Wait until the VM has finished deleting before you attach the virtual hard disk to another VM. The lease on the virtual hard disk that associates it with the VM needs to be released before you can attach the virtual hard disk to another VM.

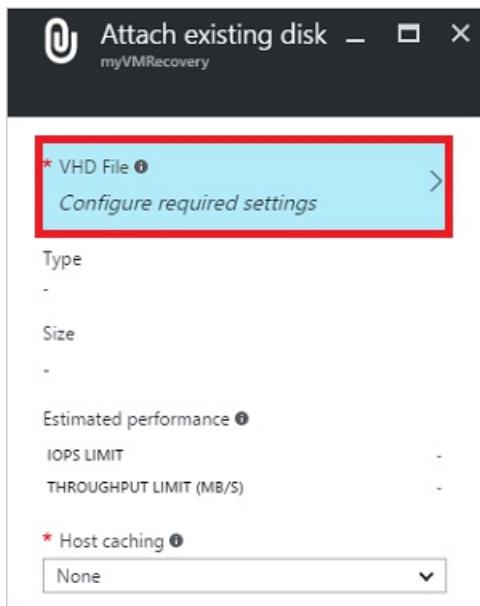
Attach existing virtual hard disk to another VM

For the next few steps, you use another VM for troubleshooting purposes. You attach the existing virtual hard disk to this troubleshooting VM to be able to browse and edit the disk's content. This process allows you to correct any configuration errors or review additional application or system log files, for example. Choose or create another VM to use for troubleshooting purposes.

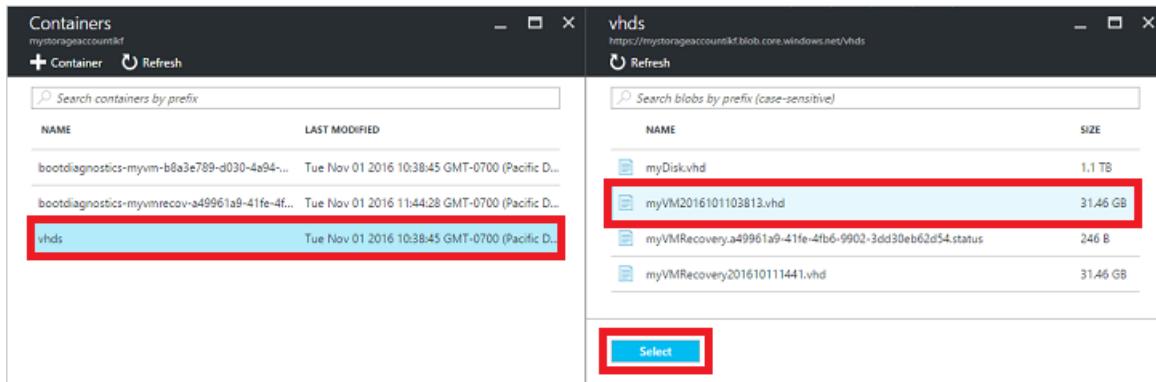
1. Select your resource group from the portal, then select your troubleshooting VM. Select **Disks** and then click **Attach existing**:



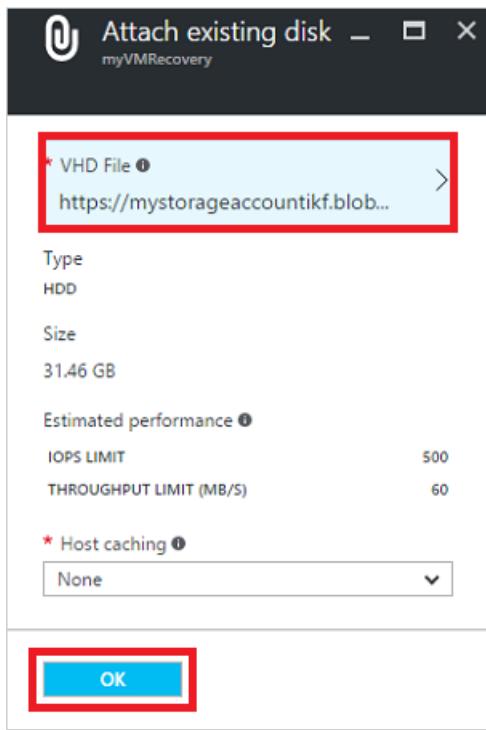
2. To select your existing virtual hard disk, click **VHD File**:



3. Select your storage account and container, then click your existing VHD. Click the **Select** button to confirm your choice:



4. With your VHD now selected, click **OK** to attach the existing virtual hard disk:



- After a few seconds, the **Disks** pane for your VM lists your existing virtual hard disk connected as a data disk:

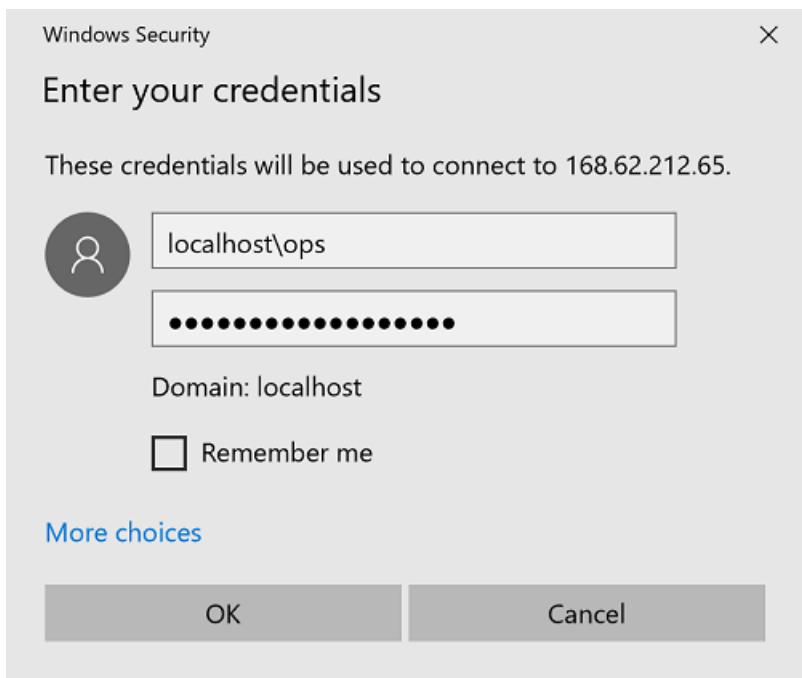
The screenshot shows the 'Disks' pane for the 'myVMRecovery' virtual machine. At the top, it says 'myVMRecovery - Disks' and 'Virtual machine'. There are buttons for 'Attach new' and 'Attach existing'. On the left is a sidebar with links: 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', and 'Diagnose and solve problems'. The main area shows a table of disks:

DISK	SIZE	ENCRYPTION
OS DISK		
myVMRecovery		Not enabled
DATA DISKS		
myVM2016101103813		

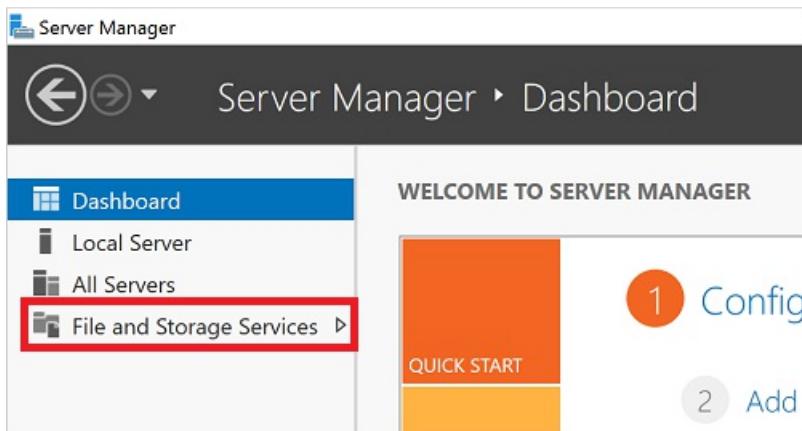
A red box highlights the 'myVM2016101103813' entry in the DATA DISKS section.

Mount the attached data disk

- Open a Remote Desktop connection to your VM. Select your VM in the portal and click **Connect**. Download and open the RDP connection file. Enter your credentials to log in to your VM as follows:



2. Open **Server Manager**, then select **File and Storage Services**.



3. The data disk is automatically detected and attached. To see a list of the connected disks, select **Disks**. You can select your data disk to view volume information, including the drive letter. The following example shows the data disk attached and using **F:**

The screenshot shows the Windows Server Manager interface under the 'File and Storage Services' section. In the left navigation pane, 'Disks' is selected and highlighted with a red box. The main content area displays two tables: 'DISKS' and 'VOLUMES'.
DISKS:
All disks | 3 total
Filter
Number Virtual Disk Status Capacity Unallocated Partition Read Only Clustered
myVMRecovery (3)
0 Online 127 GB 2.00 MB MBR
1 Online 50.0 GB 0.00 B MBR
2 Online 127 GB 2.00 MB MBR
Last refreshed on 12/13/2016 7:16:34 PM
VOLUMES:
Related Volumes | 1 total
Tasks
Filter
▲ Volume Status Provisioning Capacity Free Space Deduplication Rate Dedu...
myVMRecovery (1)
F: Fixed 127 GB 115 GB
Go to Volumes Overview >

Fix issues on original virtual hard disk

With the existing virtual hard disk mounted, you can now perform any maintenance and troubleshooting steps as needed. Once you have addressed the issues, continue with the following steps.

Unmount and detach original virtual hard disk

Once your errors are resolved, detach the existing virtual hard disk from your troubleshooting VM. You cannot use your virtual hard disk with any other VM until the lease attaching the virtual hard disk to the troubleshooting VM is released.

1. From the RDP session to your VM, open **Server Manager**, then select **File and Storage Services**:

The screenshot shows the Windows Server Manager interface. In the left navigation pane, 'File and Storage Services' is selected and highlighted with a red box. The main content area displays a 'WELCOME TO SERVER MANAGER' screen with two large buttons: 'Config' (with a circled '1') and 'Add' (with a circled '2').
Dashboard
Local Server
All Servers
File and Storage Services >

2. Select **Disks** and then select your data disk. Right-click on your data disk and select **Take Offline**:

Number	Virtual Disk	Status	Capacity	Unallocated	Partition	Read Only
0		Online	127 GB	2.00 MB	MBR	
1		Online	50.0 GB	0.00 B	MBR	
2		Online	127 GB	2.00 MB	MBR	

3. Now detach the virtual hard disk from the VM. Select your VM in the Azure portal and click **Disks**. Select your existing virtual hard disk and then click **Detach**:

myDisk
myVMRecovery

https://mystorageaccountikf.blob.core.windows.net/

Type: HDD

* Size (GiB):

Estimated performance:

- IOPS LIMIT:
- THROUGHPUT LIMIT (MB/S):

Logical Unit Number (LUN): 0

* Host caching:

Wait until the VM has successfully detached the data disk before continuing.

Create VM from original hard disk

To create a VM from your original virtual hard disk, use [this Azure Resource Manager template](#). The template deploys a VM into an existing virtual network, using the VHD URL from the earlier command. Click the **Deploy to Azure** button as follows:

marcvanek committed with kvaes Location fix ...

Latest commit 0f30cf on Apr 17

README.md Add "Visualize" buttons to all template README.md files 10 months ago

azuredeploy.json Location fix 7 months ago

azuredeploy.parameters.json Location fix 7 months ago

metadata.json Update metadata.json a year ago

README.md

Create a specialized virtual machine in an existing virtual network

Deploy to Azure **Visualize**

The template is loaded into the Azure portal for deployment. Enter the names for your new VM and existing Azure resources, and paste the URL to your existing virtual hard disk. To begin the deployment, click **Purchase**:

Create a VM from a custom VHD and connect it to an existing VNET
Azure quickstart template

TEMPLATE

201-vm-specialized-vhd-existing-vnet 4 resources **Edit** **Learn more**

BASICS

- * Subscription: Visual Studio Ultimate with MSDN
- * Resource group: Create new: myResourceGroup
- * Location: West US

SETTINGS

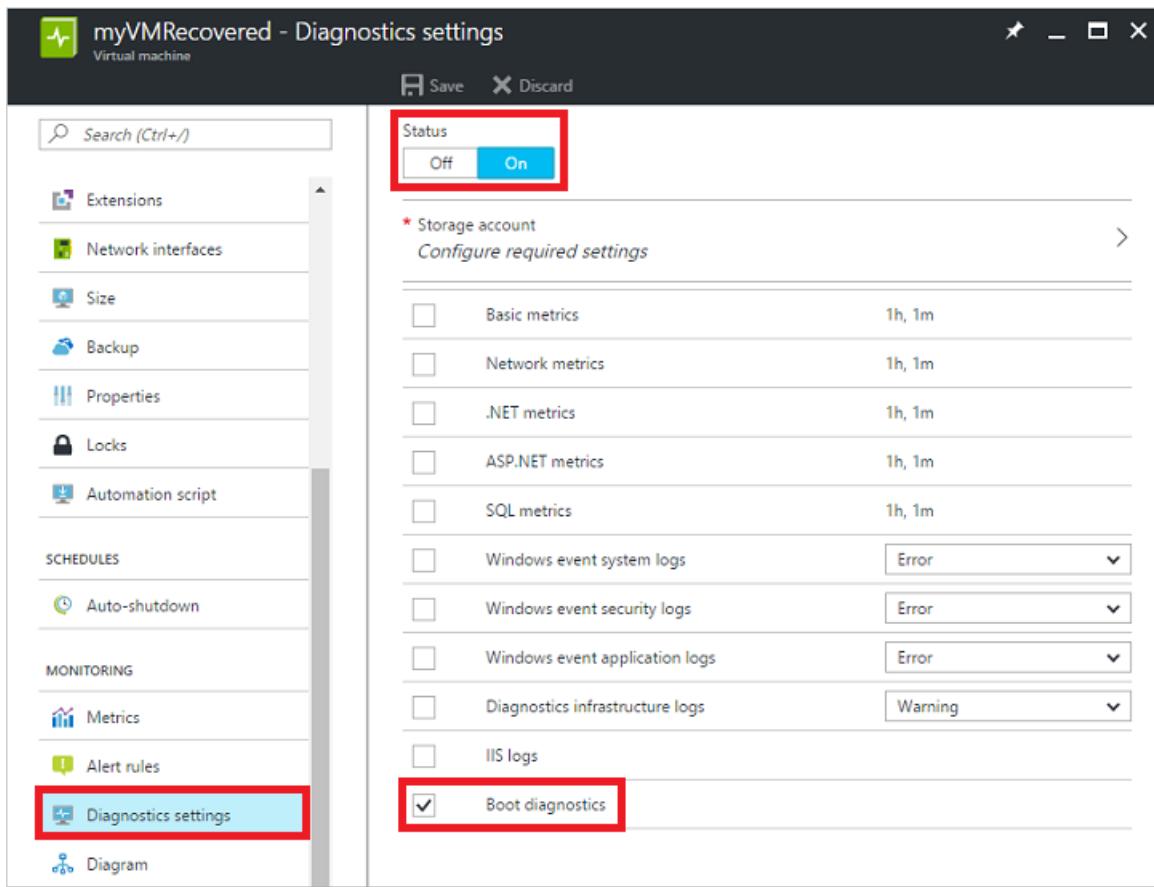
- * Vm Name: myVMRecovered
- * Os Type: Windows
- * Os Disk Vhd Uri: <https://mystorageaccountikf.blob.core.windows.net/vhds/myVM20161213105201...>
- * Vm Size: Standard_DS1_v2
- * Existing Virtual Network Name: myVnet

Pin to dashboard

Purchase

Re-enable boot diagnostics

When you create your VM from the existing virtual hard disk, boot diagnostics may not automatically be enabled. To check the status of boot diagnostics and turn on if needed, select your VM in the portal. Under **Monitoring**, click **Diagnostics settings**. Ensure the status is **On**, and the check mark next to **Boot diagnostics** is selected. If you make any changes, click **Save**:



Next steps

If you are having issues connecting to your VM, see [Troubleshoot RDP connections to an Azure VM](#). For issues with accessing applications running on your VM, see [Troubleshoot application connectivity issues on a Windows VM](#).

For more information about using Resource Manager, see [Azure Resource Manager overview](#).

How to use Perflnights

12/14/2017 • 11 min to read • [Edit Online](#)

Perflnights is an automated script that collects useful diagnostic information. It also runs I/O stress loads, and provides an analysis report to help troubleshoot Windows virtual machine performance problems in Azure. This can be run on virtual machines as a standalone script, or directly from the portal by installing [Azure Performance Diagnostics VM Extension](#).

If you are experiencing performance problems with virtual machines, before contacting support, run this script.

Supported troubleshooting scenarios

Perflnights can collect and analyze several kinds of information. The following sections cover common scenarios.

Collect basic configuration

This scenario collects the disk configuration and other important information, including:

- Event logs
- Network status for all incoming and outgoing connections
- Network and firewall configuration settings
- Task list for all applications that are currently running on the system
- Information file created by msinfo32 for the virtual machine
- Microsoft SQL Server database configuration settings (if the VM is identified as a server that is running SQL Server)
- Storage reliability counters
- Important Windows hotfixes
- Installed filter drivers

This is a passive collection of information that shouldn't affect the system.

NOTE

This scenario is automatically included in each of the following scenarios.

Benchmarking

This scenario runs the [Diskspd](#) benchmark test (IOPS and MBPS) for all drives that are attached to the VM.

NOTE

This scenario can affect the system, and shouldn't be run on a live production system. If necessary, run this scenario in a dedicated maintenance window to avoid any problems. An increased workload that is caused by a trace or benchmark test can adversely affect the performance of your VM.

Slow VM analysis

This scenario runs a [performance counter](#) trace by using the counters that are specified in the Generalcounters.txt

file. If the VM is identified as a server that is running SQL Server, it runs a performance counter trace. It does so by using the counters that are found in the Sqlcounters.txt file, and it also includes performance diagnostics data.

Slow VM analysis and benchmarking

This scenario runs a [performance counter](#) trace that is followed by a [Diskspd](#) benchmark test.

NOTE

This scenario can affect the system, and shouldn't be run on a live production system. If necessary, run this scenario in a dedicated maintenance window to avoid any problems. An increased workload that is caused by a trace or benchmark test can adversely affect the performance of your VM.

Azure Files analysis

This scenario runs a special performance counter capture together with a network trace. The capture includes all the Server Message Block (SMB) client shares counters. The following are some key SMB client share performance counters that are part of the capture:

TYPE	SMB CLIENT SHARES COUNTER
IOPS	Data Requests/sec
	Read Requests/sec
	Write Requests/sec
Latency	Avg. sec/Data Request
	Avg. sec/Read
	Avg. sec/Write
IO Size	Avg. Bytes/Data Request
	Avg. Bytes/Read
	Avg. Bytes/Write
Throughput	Data Bytes/sec
	Read Bytes/sec
	Write Bytes/sec
Queue Length	Avg. Read Queue Length
	Avg. Write Queue Length
	Avg. Data Queue Length

Custom slow VM analysis

When you run a custom slow VM analysis, you select traces to run in parallel. You can run them all (Performance Counter, Xperf, Network, and StorPort) if you want. After tracing is completed, the tool runs the Diskspd benchmark, if it is selected.

NOTE

This scenario can affect the system, and shouldn't be run on a live production system. If necessary, run this scenario in a dedicated maintenance window to avoid any problems. An increased workload that is caused by a trace or benchmark test can adversely affect the performance of your VM.

What kind of information is collected by the script?

Information about Windows VM, disks or storage pools configuration, performance counters, logs, and various traces are collected. It depends on the performance scenario you are using. The following table provides details:

DATA COLLECTED			PERFORMANCE SCENARIOS			
	Collect basic configuration	Benchmarking	Slow VM analysis	Slow VM analysis and benchmarking	Azure Files analysis	Custom slow VM analysis
Information from event logs	Yes	Yes	Yes	Yes	Yes	Yes
System information	Yes	Yes	Yes	Yes	Yes	Yes
Volume map	Yes	Yes	Yes	Yes	Yes	Yes
Disk map	Yes	Yes	Yes	Yes	Yes	Yes
Running tasks	Yes	Yes	Yes	Yes	Yes	Yes
Storage reliability counters	Yes	Yes	Yes	Yes	Yes	Yes
Storage information	Yes	Yes	Yes	Yes	Yes	Yes
Fsutil output	Yes	Yes	Yes	Yes	Yes	Yes
Filter driver info	Yes	Yes	Yes	Yes	Yes	Yes
Netstat output	Yes	Yes	Yes	Yes	Yes	Yes
Network configuration	Yes	Yes	Yes	Yes	Yes	Yes
Firewall configuration	Yes	Yes	Yes	Yes	Yes	Yes
SQL Server configuration	Yes	Yes	Yes	Yes	Yes	Yes

DATA COLLECTED			PERFORMANCE SCENARIOS		
Performance diagnostics traces *	Yes	Yes	Yes	Yes	Yes
Performance counter trace **					Yes
SMB counter trace **				Yes	
SQL Server counter trace **					Yes
Xperf trace					Yes
StorPort trace					Yes
Network trace				Yes	Yes
Diskspd benchmark trace ***		Yes		Yes	

Performance diagnostics trace (*)

Runs a rule-based engine in the background to collect data and diagnose ongoing performance issues. The following rules are currently supported:

- HighCpuUsage rule: Detects high CPU usage periods, and shows the top CPU usage consumers during those periods.
- HighDiskUsage rule: Detects high disk usage periods on physical disks, and shows the top disk usage consumers during those periods.
- HighResolutionDiskMetric rule: Shows IOPS, throughput, and I/O latency metrics per 50 milliseconds for each physical disk. It helps to quickly identify disk throttling periods.
- HighMemoryUsage rule: Detects high memory usage periods, and shows the top memory usage consumers during those periods.

NOTE

Currently, Windows versions that include the .NET Framework 3.5 or later versions are supported.

Performance counter trace (**)

Collects the following performance counters:

- \Process, \Processor, \Memory, \Thread, \PhysicalDisk, and \LogicalDisk
- \Cache\Dirty Pages, \Cache\Lazy Write Flushes/sec, \Server\Pool Nonpaged Failures, and \Server\Pool Paged Failures
- Selected counters under \Network Interface, \IPv4\Datagrams, \IPv6\Datagrams, \TCPv4\Segments, \TCPv6\Segments, \Network Adapter, \WFPv4\Packets, \WFPv6\Packets, \UDPV4\Datagrams,

\UDPV6\Datagrams, \TCPv4\Connection, \TCPv6\Connection, \Network QoS Policy\packets, \Per Processor Network Interface Card Activity, and \Microsoft Winsock BSP

For SQL Server instances

- \SQL Server:Buffer Manager, \SQLServer:Resource Pool Stats, and \SQLServer:SQL Statistics\
- \SQLServer:Locks, \SQLServer:General, Statistics
- \SQLServer:Access Methods

For Azure Files

\SMB Client Shares

Diskspd benchmark trace (***)

Diskspd I/O workload tests (OS Disk [write] and pool drives [read/write])

Run the PerflInsights script on your VM

What do I have to know before I run the script?

Script requirements

- This script must be run on the VM that has the performance issue.
- The following operating systems are supported: Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2, and Windows Server 2016; Windows 8.1 and Windows 10.

Possible problems when you run the script on production VMs

- For any benchmarking scenarios or the "Custom slow VM analysis" scenario that is configured to use Xperf or Diskspd, the script might adversely affect the performance of the VM. You should not run these scenarios in a production environment.
- For any benchmarking scenarios or the "Custom slow VM analysis" scenario that is configured to use Diskspd, ensure that no other background activity interferes with the I/O workload.
- By default, the script uses the temporary storage drive to collect data. If tracing stays enabled for a longer time, the amount of data that is collected might be relevant. This can reduce the availability of space on the temporary disk, and can therefore affect any application that relies on this drive.

How do I run PerflInsights?

You can run PerflInsights on a virtual machine by installing [Azure Performance Diagnostics VM Extension](#). You can also run it as a standalone script.

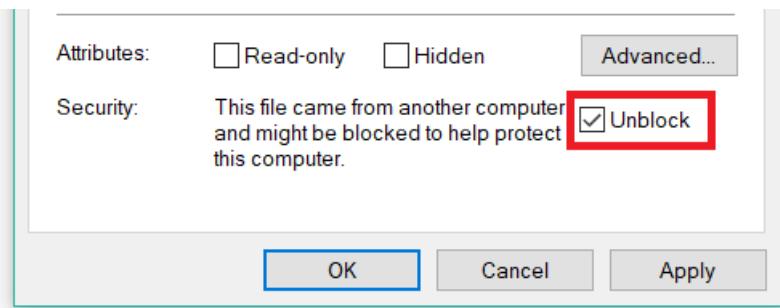
Install and run PerflInsights from the Azure portal

For more information about this option, see [Install Azure Performance Diagnostics VM Extension](#).

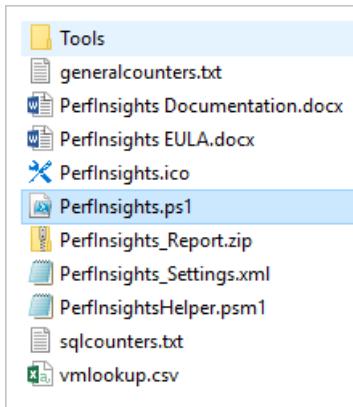
Run PerflInsights script in standalone mode

To run the PerflInsights script, follow these steps:

1. Download [PerflInsights.zip](#).
2. Unblock the PerflInsights.zip file. To do this, right-click the PerflInsights.zip file, and select **Properties**. In the **General** tab, select **Unblock**, and then select **OK**. This ensures that the script runs without any additional security prompts.



3. Expand the compressed Perflnights.zip file into your temporary drive (by default, this is usually the D drive).
The compressed file should contain the following files and folders:

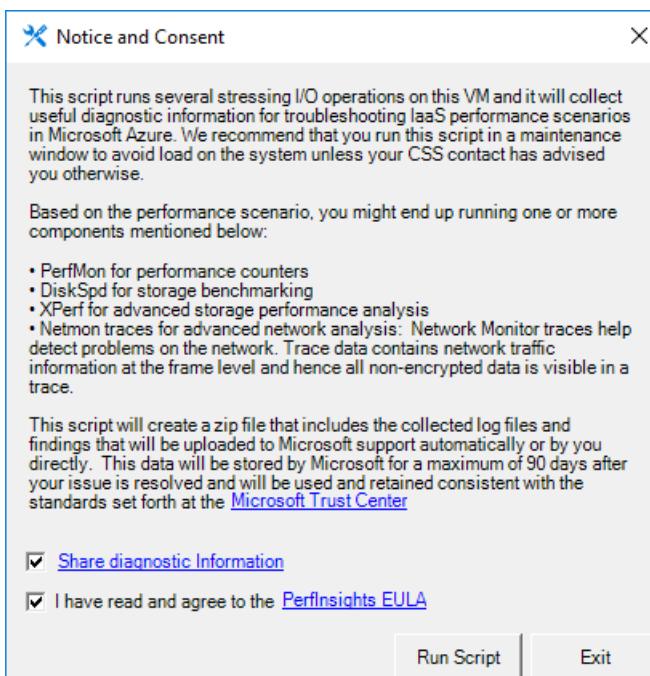


4. Open Windows PowerShell as an administrator, and then run the Perflnights.ps1 script.

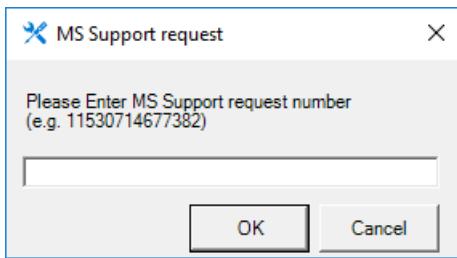
```
cd <the path of Perflnights folder >
Powershell.exe -ExecutionPolicy UnRestricted -NoProfile -File .\\Perflnights.ps1
```

You might have to enter "y" to confirm that you want to change the execution policy.

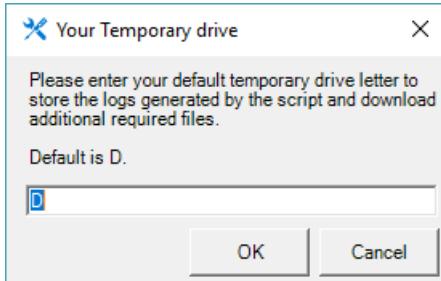
In the **Notice and Consent** dialog box, you have the option to share diagnostic information with Microsoft Support. You must also consent to the license agreement to continue. Make your selections, and then select **Run Script**.



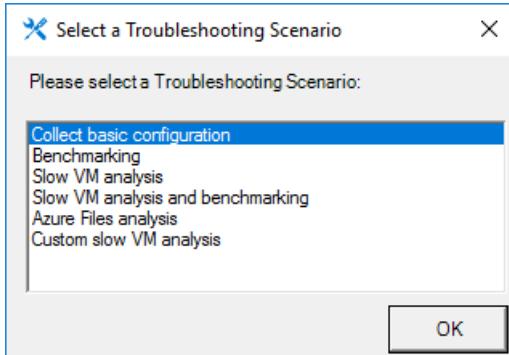
5. Submit the case number, if it is available, when you run the script. Then select **OK**.



6. Select your temporary storage drive. The script can auto-detect the drive letter of the drive. If any problems occur in this stage, you might be prompted to select the drive (the default drive is D). Generated logs are stored here, in the log_collection folder. After you enter or accept the drive letter, select **OK**.



7. Select a troubleshooting scenario from the provided list.



You can also run PerfInsights without UI. The following command runs the "Slow VM analysis" troubleshooting scenario without UI. It prompts you to consent to the same disclaimer and EULA that are mentioned in step 4.

```
powershell.exe -ExecutionPolicy UnRestricted -NoProfile -Command ".\\PerfInsights.ps1 -NoGui -Scenario vmslow -TracingDuration 30"
```

If you want PerfInsights to run in silent mode, use the **-AcceptDisclaimerAndShareDiagnostics** parameter. For example, use the following command:

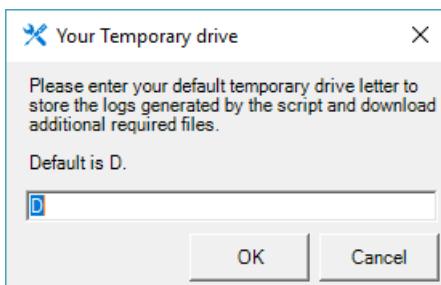
```
powershell.exe -ExecutionPolicy UnRestricted -NoProfile -Command ".\\PerfInsights.ps1 -NoGui -Scenario vmslow -TracingDuration 30 -AcceptDisclaimerAndShareDiagnostics"
```

How do I troubleshoot issues while running the script?

If the script terminates abnormally, you can return to a consistent state by running the script together with the cleanup switch, as follows:

```
powershell.exe -ExecutionPolicy UnRestricted -NoProfile -Command ".\\PerfInsights.ps1 -Cleanup"
```

If any problems occur during the automatic detection of the temporary drive, you might be prompted to select the drive (the default drive is D).



The script uninstalls the utility tools and removes temporary folders.

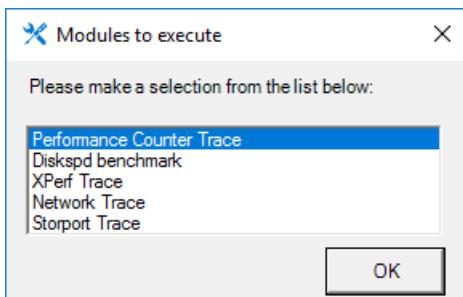
Troubleshoot other script issues

If any problems occur when you run the script, press **Ctrl+C** to interrupt the script. If you continue to experience script failure even after several attempts, run the script in debug mode by using the "**-Debug**" parameter option on startup.

After the failure occurs, copy the full output of the PowerShell console, and send it to the Microsoft Support agent who is assisting you to help troubleshoot the problem.

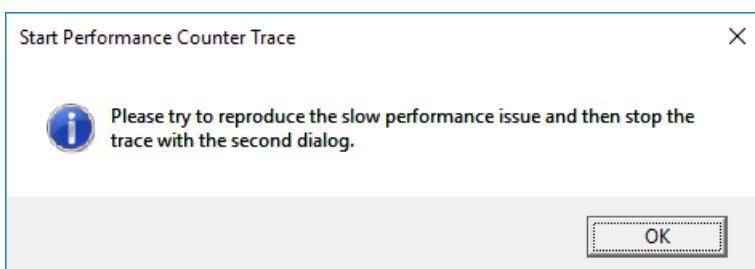
How do I run the script in "Custom slow VM analysis" mode?

By selecting the **Custom slow VM analysis**, you can enable several traces in parallel (to select multiple traces, use the Shift key):

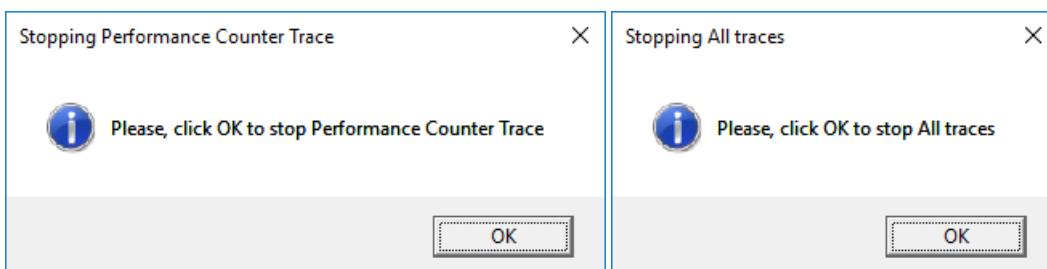


When you select the Performance Counter Trace, Xperf Trace, Network Trace, or Storport Trace scenarios, follow the instructions in the subsequent dialog boxes. Try to reproduce the slow performance issue after you start the traces.

You start a trace through the following dialog box:



To stop the traces, you have to confirm the command in a second dialog box.



When the traces or operations are completed, a new file appears in D:\log_collection (or the temporary drive). The name of the file is **CollectedData_yyyy-MM-dd_hh_mm_ss.zip**. You can send this file to the support agent for analysis.

Review the diagnostics report

Within the **CollectedData_yyyy-MM-dd_hh_mm_ss.zip** file, you can find an HTML report that details the findings of Perflnights. To review the report, expand the **CollectedData_yyyy-MM-dd_hh_mm_ss.zip** file, and then open the **Perflnights Report.html** file.

Select the **Findings** tab.

The screenshot shows the 'Perflnights Report' interface with the 'Findings' tab selected. At the top, there are tabs for Overview, Storage, Network, SQL, Diagnostic, Findings (which is highlighted), and Run Information. Below these tabs, the main content area is titled 'Findings' and contains a table of findings categorized by rule name and severity. The table has columns for Category, Rule Name, Critical, Important, and Informational counts. Below the table, there are more tabs for Overview, Storage, Network, SQL, Diagnostic, Disk Map, Volume Map, Storage Pool, Storage Reliability Counters, Filter Drivers, and Fsutil. The 'Disk Map' tab is selected. The main content area now displays a detailed list of findings with columns for Impact Level, Finding, Impacted Resources, Recommendation, References, and Feedback. Each finding is preceded by a plus sign icon.

Category	Rule Name	Critical	Important	Informational
Storage	DiskConfigurationRule	5	2	0
SQL	SqlSettingsRule	2	2	0
Diagnostic	KnowledgeBaseCheckRule	1	0	0
Network	NetworkConfigurationRule	1	0	0
Diagnostic	HighCpuRule	0	1	0
Diagnostic	HighDiskRule	0	1	0

Impact Level	Finding	Impacted Resources	Recommendation	References	Feedback
+ Critical	Number of columns defined for virtual disks do not match to count of physical disks in their pools.	2	For optimal performance, property NumberOfColumn should match the number of disks in the pool. Recreate the virtual disk using PS cmdlet: New-VirtualDisk with option NumberOfColumns.	<ul style="list-style-type: none">Recommended Storage Configuration for Azure VMs	
+ Critical	Virtual disks have incorrect ResiliencySettingName value for optimal performance.	2	Expected ResiliencySettingName value is: Simple. Redundancy is already implemented in Azure Storage.	<ul style="list-style-type: none">Recommended Storage Configuration for Azure VMs	
+ Critical	The allocation unit size of volumes with SQL files can be optimized.	1	Since SQL Server is running on those volumes, it is preferable to have allocation unit size of 64KB.	<ul style="list-style-type: none">SQL Server Best Practices	
+ Critical	Pools have more than one virtual disk.	1	Azure VMs should be created with a single virtual disk per pool for optimal performance.	<ul style="list-style-type: none">Recommended Storage Configuration for Azure VMs	
+ Critical	Virtual disks have unexpected interleave value.	1	Expected interleave values for virtual disks are: 65536 (for OLPT IO load), 262144 (for DataWarehouse IO Load).	<ul style="list-style-type: none">Configuring Azure Virtual Machines for Optimal Storage Performance	
+ Important	Physical disks have multiple Volumes/Partitions.	3	Please consider configuring only one volume on each physical disk.	<ul style="list-style-type: none">Recommended Storage Configuration for Azure VMs	
+ Important	NumberOfColumn property of virtual disks can be optimized.	3	Storage Spaces Virtual Disks configured with Stripping requires at least 2 disks to improve performance.	<ul style="list-style-type: none">Recommended Storage Configuration for Azure VMs	

NOTE

Findings categorized as critical are known issues that might cause performance issues. Findings categorized as important represent non-optimal configurations that do not necessarily cause performance issues. Findings categorized as informational are informative statements only.

Review the recommendations and links for all critical and important findings. Learn about how they can affect performance, and also about best practices for performance-optimized configurations.

Storage tab

The **Findings** section displays various findings and recommendations related to storage.

The **Disk Map** and **Volume Map** sections describe how logical volumes and physical disks are related to each other.

In the physical disk perspective (Disk Map), the table shows all logical volumes that are running on the disk. In the following example, **PhysicalDrive2** runs two logical volumes created on multiple partitions (J and H):

PhysicalDiskName	IsOSDisk	IsTempDisk	SizeGB	SCSIBus	SCSILun	SCSIPort	SCSITargetId	Letter	Label
PhysicalDrive0	✓		127	0	0	0	0	C:	
PhysicalDrive1	✓		40	0	0	0	1	D:	Temporary Storage
PhysicalDrive2			1023	0	0	3	0	J:	Log3
								H:	Data2

In the volume perspective (Volume Map), the tables show all the physical disks under each logical volume. Notice that for RAID/Dynamic disks, you might run a logical volume on multiple physical disks. In the following example, C:\mount is a mount point configured as *SpannedDisk* on physical disks 2 and 3:

VolumeID	Label	AUS	DriveType	SizeGB	FreeGB	FreePerc	IsOSDisk	IsTempDisk	IsDataDisk	IsSystemDisk	IsDynamic	IsPooled	PoolName	VirtualDiskName	Resiliency	Interleave	NumOfCol	NumOfDisks	PhysicalDiskName	LUN
J:	Log3	4	LocalDisk	1003.466	1003.346	99.99		✓		✓					Simple		1		PhysicalDrive2	0
C:\mount	Data3	4	LocalDisk	39.062	38.970	99.76		✓		✓					Spanned		2		PhysicalDrive3	1
																			PhysicalDrive4	2

SQL tab

If the target VM hosts any SQL Server instances, you see an additional tab in the report, named **SQL**:

The screenshot shows a navigation bar with tabs: Overview, Storage, Network, SQL (highlighted with a red box), and Diagnostic. Below the tabs are three instance-specific tabs: Findings, MSSQLSERVER, and INSTANCE2.

This section contains a **Findings** tab, and additional tabs for each of the SQL Server instances hosted on the VM.

The **Findings** tab contains a list of all the SQL related performance issues found, along with the recommendations.

In the following example, **PhysicalDrive0** (running the C drive) is displayed. This is because both the **modeldev** and **modellog** files are located on the C drive, and they are of different types (such as data file and transaction log, respectively).

PhysicalDisks	LUN	VolumeID	IsOSDisk	InstanceName	DataBaseName	DataBaseId	FileName	FilePath	FileType	IsSystemDB
PhysicalDrive0	0	C:	!	MSSQLSERVER	model	3	modeldev	C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA\model.mdf	Data	✓
				MSSQLSERVER	model	3	modellog	C:\Program Files\Microsoft SQL Server\MSSQL10_50.MSSQLSERVER\MSSQL\DATA\modellog.ldf	Log	✓

The tabs for specific instances of SQL Server contain a general section that displays basic information about the selected instance. The tabs also contain additional sections for advanced information, including settings, configurations, and user options.

Diagnostic tab

The **Diagnostic** tab contains information about top CPU, disk, and memory consumers on the computer for the duration of the running of PerflInsights. You can also find information about critical patches that the system might be missing, the task list, and important system events.

References to the external tools used

Diskspd

Diskspd is a storage load generator and performance test tool from Microsoft. For more information, see [Diskspd](#).

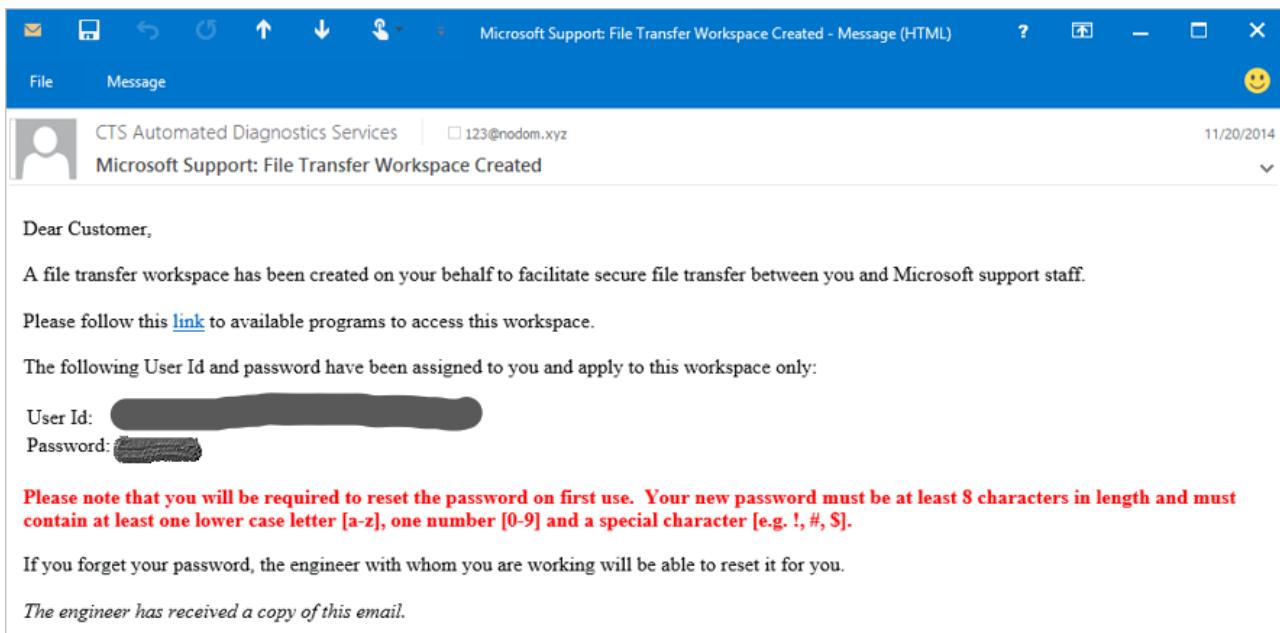
Xperf

Xperf is a command-line tool to capture traces from the Windows Performance Toolkit. For more information, see [Windows Performance Toolkit – Xperf](#).

Next steps

You can upload diagnostics logs and reports to Microsoft Support for further review. Support might request that you transmit the output that is generated by Perflnights to assist with the troubleshooting process.

The following screenshot shows a message similar to what you might receive:



Follow the instructions in the message to access the file transfer workspace. For additional security, you have to change your password on first use.

After you sign in, you will find a dialog box to upload the **CollectedData_yyyy-MM-dd_hh_mm_ss.zip** file that was collected by Perflnights.

Azure Performance Diagnostics VM Extension for Windows

12/14/2017 • 6 min to read • [Edit Online](#)

Azure Performance Diagnostics VM Extension helps collect performance diagnostic data from Windows VMs. The extension performs analysis, and provides a report of findings and recommendations to identify and resolve performance issues on the virtual machine. This extension installs a troubleshooting tool called [PerfInsights](#).

Prerequisites

This extension can be installed on Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2, and Windows Server 2016. It can also be installed on Windows 8.1 and Windows 10.

Extension schema

The following JSON shows the schema for Azure Performance Diagnostics VM Extension. This extension requires the name and key for a storage account to store the diagnostics output and report. These values are sensitive and should be stored inside a protected setting configuration. Azure VM extension protected setting data is encrypted, and it is only decrypted on the target virtual machine. Note that **storageAccountName** and **storageAccountKey** are case-sensitive. Other required parameters are listed in the following section.

```
{
  "name": "[concat(parameters('vmName'), '/AzurePerformanceDiagnostics')]",
  "type": "Microsoft.Compute/virtualMachines/extensions",
  "location": "[parameters('location')]",
  "apiVersion": "2015-06-15",
  "properties": {
    "publisher": "Microsoft.Azure.Performance.Diagnostics",
    "type": "AzurePerformanceDiagnostics",
    "typeHandlerVersion": "1.0",
    "autoUpgradeMinorVersion": true,
    "settings": {
      "performanceScenario": "[parameters('performanceScenario')]",
      "traceDurationInSeconds": "[parameters('traceDurationInSeconds')]",
      "perfCounterTrace": "[parameters('perfCounterTrace')]",
      "networkTrace": "[parameters('networkTrace')]",
      "xperfTrace": "[parameters('xperfTrace')]",
      "storPortTrace": "[parameters('storPortTrace')]",
      "srNumber": "[parameters('srNumber')]",
      "requestTimeUtc": "[parameters('requestTimeUtc')]"
    },
    "protectedSettings": {
      "storageAccountName": "[parameters('storageAccountName')]",
      "storageAccountKey": "[parameters('storageAccountKey')]"
    }
  }
}
```

Property values

NAME	VALUE / EXAMPLE	DESCRIPTION
apiVersion	2015-06-15	The version of the API.

NAME	VALUE / EXAMPLE	DESCRIPTION
publisher	Microsoft.Azure.Performance.DiagnosticS	The publisher namespace for the extension.
type	AzurePerformanceDiagnostics	The type of the VM extension.
typeHandlerVersion	1.0	The version of the extension handler.
performanceScenario	basic	The performance scenario for which to capture data. Valid values are: basic , vmslow , azurefiles , and custom .
traceDurationInSeconds	300	The duration of the traces, if any of the trace options are selected.
perfCounterTrace	p	Option to enable Performance Counter Trace. Valid values are p or empty value. If you do not want to capture this trace, leave the value as empty.
networkTrace	n	Option to enable Network Trace. Valid values are n or empty value. If you do not want to capture this trace, leave the value as empty.
xperfTrace	x	Option to enable XPerf Trace. Valid values are x or empty value. If you do not want to capture this trace, leave the value as empty.
storPortTrace	s	Option to enable StorPort Trace. Valid values are s or empty value. If you do not want to capture this trace, leave the value as empty.
srNumber	123452016365929	The support ticket number, if available. Leave the value as empty if you don't have it.
storageAccountName	mystorageaccount	The name of the storage account to store the diagnostics logs and results.
storageAccountKey	lDuVvxuZB28NNP...hAiRF3voADxLBcc==	The key for the storage account.

Install the extension

Follow these steps to install the extension on Windows virtual machines:

1. Sign in to the [Azure portal](#).
2. Select the virtual machine where you want to install this extension.

Microsoft Azure Virtual machines

New Dashboard Resource groups All resources Recent App Services SQL databases Virtual machines (classic) Virtual machines

Virtual machines Microsoft

Add Assign Tags Columns Refresh Start Restart Stop

i Virtual machines and Virtual machines (classic) can now be managed together in the co

Subscriptions: 2 of 12 selected

Filter by name... 2 subscriptions

NAME TYPE

NAME	TYPE
azperf [REDACTED]	Virtual machine
azperf	Virtual machine
azperf	Virtual machine

3. Select the **Extensions** blade, and select **Add**.

Search (Ctrl+ /)

+ Add

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

SETTINGS

Networking Disks Size Extensions

Search to filter items...

NAME

NAME
IaaSAntimalware
MicrosoftMonitoringAgent

4. Select **Azure Performance Diagnostics**, review the terms and conditions, and select **Create**.

New resource

-  Agent for Cloud Workload Protection (Windows)
Symantec Corp.
-  Deep Security Agent
Trend Micro
-  Control-M Agent
BMC Software, Inc.
-  Windows Chef Extension (1.2.3)
Chef Software Inc.
-  Dynatrace OneAgent
Dynatrace
-  ESET File Security
ESET
-  HPE Security Fortify Application Defender
HPE Security Fortify
-  Azure Performance Diagnostics
Microsoft Corp.
-  Microsoft Antimalware
Microsoft Corp.
-  Network Watcher Agent for Windows
Microsoft Corp.
-  Team Services Agent
Microsoft Corp.

Azure Performance Diagnostics

Microsoft Corp.

This VM extension runs PerfInsights script that collects useful diagnostics information for troubleshooting IaaS performance issues on this Azure VM. We recommend that you work with your CSS contact to select the right performance scenario to help troubleshoot the performance issue you are having.

Based on the performance scenario, you might end up running one or more components mentioned below:

- Performance diagnostic traces for CPU, Disk and memory operations
- PerfMon for performance counters
- XPerf for advanced storage performance analysis
- Netmon traces for advanced network analysis: Network Monitor traces help detect problems on the network. Trace data contains network traffic information at the frame level and hence all non-encrypted data is visible in a trace.

This script will create a zip file that includes the collected log files and findings that will be uploaded to Microsoft support automatically or by you directly. Additionally, this data is uploaded to the specified storage account, shared using SAS (Shared Access Signatures) and available for download for 30 days. This data will be stored by Microsoft for a maximum of 90 days after your issue is resolved and will be used and retained consistent with the standards set forth at the [Microsoft Trust Center](#).

Legal Terms: By clicking the create button I acknowledge that I have read and agree to the [PerfInsights EULA.docx](#) available inside [PerfInsights download](#) file and to [share diagnostics information](#).

Please provide your comment or feedback at [Azure Feedback](#)

[!\[\]\(c49a139f16440927c3877fe276967c08_img.jpg\)](#)
[!\[\]\(e99aeda0d78d1d099d308155ac42501a_img.jpg\)](#)
[!\[\]\(1a72c5a3a63d8a9ec6ee05383d846c09_img.jpg\)](#)
[!\[\]\(a4cfb13fd896da327b7870c08dfef216_img.jpg\)](#)
[!\[\]\(a66fb4deac51f469809ea7759516a8ea_img.jpg\)](#)
[!\[\]\(57b4bf06e668a6fba8196fa19bf607d5_img.jpg\)](#)

PUBLISHER	Microsoft Corp.
USEFUL LINKS	Azure Performance Diagnostics Extension PerfInsights

[Create](#)

5. Provide the parameter values for the installation, and select **OK** to install the extension. For more information about supported scenarios, see [How to use PerfInsights](#).

Install extension

* Storage Account Name

* Storage Account Key

Performance Scenario

Trace duration

Performance counter trace On Off

Network trace On Off

XPerf trace On Off

Storport trace On Off

Service Request Number

OK

- When the installation is successful, you see a message indicating this status.

NAME	TYPE	VERSION	STATUS
AzurePerformanceDiagnostics	Microsoft.Azure.Performance.Diagnostics.AzurePerformanceDiag...	1.*	Provisioning succeeded
IaaSAntimalware	Microsoft.Azure.Security.IaaSAntimalware	1.*	Provisioning succeeded
MicrosoftMonitoringAgent	Microsoft.EnterpriseCloud.Monitoring.MicrosoftMonitoringAgent	1.*	Provisioning succeeded

NOTE

The extension runs when the provisioning has succeeded. It takes two minutes or less to complete for the basic scenario. For other scenarios, it runs through the duration specified during the installation.

Remove the extension

To remove the extension from a virtual machine, follow these steps:

1. Sign in to the [Azure portal](#), select the virtual machine from which you want to remove this extension, and then select the **Extensions** blade.
2. Select the (...) for the Performance Diagnostics Extension entry from the list, and select **Uninstall**.

The screenshot shows the Azure portal's Extensions blade. On the left, there's a sidebar with links like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, SETTINGS (Networking, Disks, Size), and Extensions. The Extensions link is highlighted with a yellow box. The main area shows a table with columns: NAME, TYPE, VERSION, STATUS, and three dots (...). There are three entries: AzurePerformanceDiagnostics (Microsoft.Azure.Performance.Diagnostics.AzurePerformanceDiag...), IaaSAntimalware (Microsoft.Azure.Security.IaaSAntimalware), and MicrosoftMonitoringAgent (Microsoft.EnterpriseCloud.Monitoring.MicrosoftMonitoringAgent). The first entry has a status of 'Provisioning succeeded' and a yellow box highlights the 'Uninstall' button in the STATUS column.

NOTE

You can also select the extension entry, and select the **Uninstall** option.

Template deployment

Azure virtual machine extensions can be deployed with Azure Resource Manager templates. The JSON schema detailed in the previous section can be used in an Azure Resource Manager template. This runs the Azure Performance Diagnostics VM extension during an Azure Resource Manager template deployment. Here is a sample template:

```
{
  "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "vmName": {
      "type": "string",
      "defaultValue": "yourVMName"
    },
    "location": {
      "type": "string",
      "defaultValue": "southcentralus"
    },
    "storageAccountName": {
      "type": "securestring"
      "defaultValue": "yourStorageAccount"
    },
    "storageAccountKey": {
      "type": "securestring"
      "defaultValue": "yourStorageAccountKey"
    },
    "performanceScenario": {
      "type": "string",
      "defaultValue": "basic"
    },
    "srNumber": {
      "type": "string",
      "defaultValue": ""
    },
    "traceDurationInSeconds": {
      "type": "int",
      "defaultValue": 300
    },
    "perfCounterTrace": {
      "type": "string",
      "defaultValue": ""
    }
  }
}
```

```

        "defaultValue": "",
    },
    "networkTrace": {
        "type": "string",
        "defaultValue": ""
    },
    "xperfTrace": {
        "type": "string",
        "defaultValue": ""
    },
    "storPortTrace": {
        "type": "string",
        "defaultValue": ""
    },
    "requestTimeUtc": {
        "type": "string",
        "defaultValue": "10/2/2017 11:06:00 PM"
    }
},
"resources": [
{
    "name": "[concat(parameters('vmName'), '/AzurePerformanceDiagnostics')]",
    "type": "Microsoft.Compute/virtualMachines/extensions",
    "location": "[parameters('location')]",
    "apiVersion": "2015-06-15",
    "properties": {
        "publisher": "Microsoft.Azure.Performance.Diagnostics",
        "type": "AzurePerformanceDiagnostics",
        "typeHandlerVersion": "1.0",
        "autoUpgradeMinorVersion": true,
        "settings": {
            "performanceScenario": "[parameters('performanceScenario')]",
            "traceDurationInSeconds": "[parameters('traceDurationInSeconds')]",
            "perfCounterTrace": "[parameters('perfCounterTrace')]",
            "networkTrace": "[parameters('networkTrace')]",
            "xperfTrace": "[parameters('xperfTrace')]",
            "storPortTrace": "[parameters('storPortTrace')]",
            "srNumber": "[parameters('srNumber')]",
            "requestTimeUtc": "[parameters('requestTimeUtc')]"
        },
        "protectedSettings": {
            "storageAccountName": "[parameters('storageAccountName')]",
            "storageAccountKey": "[parameters('storageAccountKey')]"
        }
    }
}
]
}

```

PowerShell deployment

The `Set-AzureRmVMExtension` command can be used to deploy Azure Performance Diagnostics VM Extension to an existing virtual machine. Before running the command, store the public and private configurations in a PowerShell hash table.

PowerShell

```

$PublicSettings = @{
"performanceScenario": "basic", "traceDurationInSeconds": 300, "perfCounterTrace": "p", "networkTrace": "", "xperfTrace": "", "storPortTrace": "", "srNumber": "", "requestTimeUtc": "2017-09-28T22:08:53.736Z" }
$ProtectedSettings = @{"storageAccountName": "mystorageaccount", "storageAccountKey": "mystoragekey"}

Set-AzureRmVMExtension -ExtensionName "AzurePerformanceDiagnostics" ` 
    -ResourceGroupName "myResourceGroup" ` 
    -VMName "myVM" ` 
    -Publisher "Microsoft.Azure.Performance.Diagnostics" ` 
    -ExtensionType "AzurePerformanceDiagnostics" ` 
    -TypeHandlerVersion 1.0 ` 
    -Settings $PublicSettings ` 
    -ProtectedSettings $ProtectedSettings ` 
    -Location WestUS ` 

```

Information on the data captured

The PerfInsights tool collects various logs, configuration, and diagnostic data, depending on the selected scenario. For more information, see the [PerfInsights documentation](#).

View and share the results

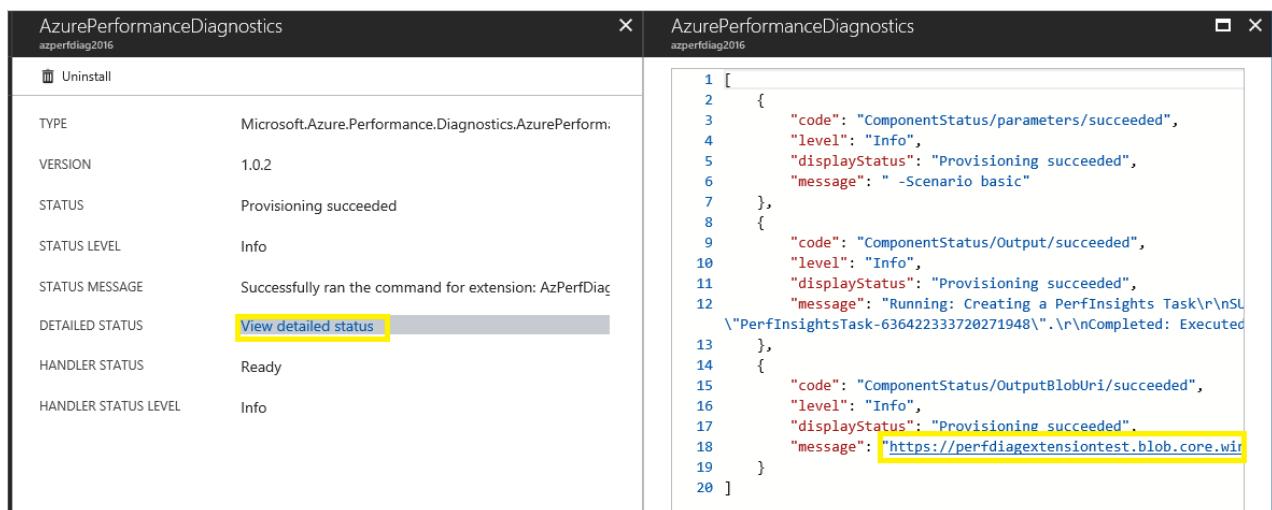
Output from the extension is stored in a folder. The folder is named log_collection, and can be found under the Temp drive (usually D:\log_collection) by default. Under this folder, you can see zip files containing the diagnostic logs, and a report with findings and recommendations.

You can also find the zip file in the storage account provided during the installation. It is shared for 30 days by using [Shared Access Signatures \(SAS\)](#). A text file named `zipfilename_saslink.txt` is also created in the `log_collection` folder. This file contains the SAS link created to download the zip file. Anyone who has this link is able to download the zip file.

To assist the support engineer working on your support ticket, Microsoft might use this SAS link to download the diagnostics data.

To view the report, extract the zip file and open the **PerfInsights Report.html** file.

You might also be able to download the zip file directly from the portal by selecting the extension.



NOTE

The SAS link displayed in the portal might not work. This can be caused by a malformed URL during the encoding and decoding operations. You can instead get the link directly from the *_saslink.txt file from the VM.

Troubleshoot and support

- Extension deployment status (in the notification area) might show “Deployment in progress” even though the extension is successfully provisioned.

This issue can be safely ignored, as long as the extension status indicates that the extension is successfully provisioned.

- You can address some issues during installation by using the extension logs. Extension execution output is logged to files found in the following directory:

```
C:\Packages\Plugins\Microsoft.Azure.Performance.Diagnostics.AzurePerformanceDiagnostics
```

If you need more help at any point in this article, you can contact the Azure experts on the [MSDN Azure and Stack Overflow forums](#). Alternatively, you can file an Azure support incident. Go to the [Azure support site](#), and select **Get support**. For information about using Azure support, read the [Microsoft Azure support FAQ](#).

Troubleshoot a problem Azure VM by using nested virtualization in Azure

11/7/2017 • 3 min to read • [Edit Online](#)

This article shows how to create a nested virtualization environment in Microsoft Azure, so you can mount the disk of the problem VM on the Hyper-V host (Recovery VM) for troubleshooting purposes.

Prerequisite

To mount the problem VM, the Recovery VM must meet the following prerequisite:

- The Recovery VM must be in the same location as the problem VM.
- The Recovery VM must be in the same resource group as the problem VM.
- The Recovery VM must use the same type of Storage Account (Standard or Premium) as the problem VM.

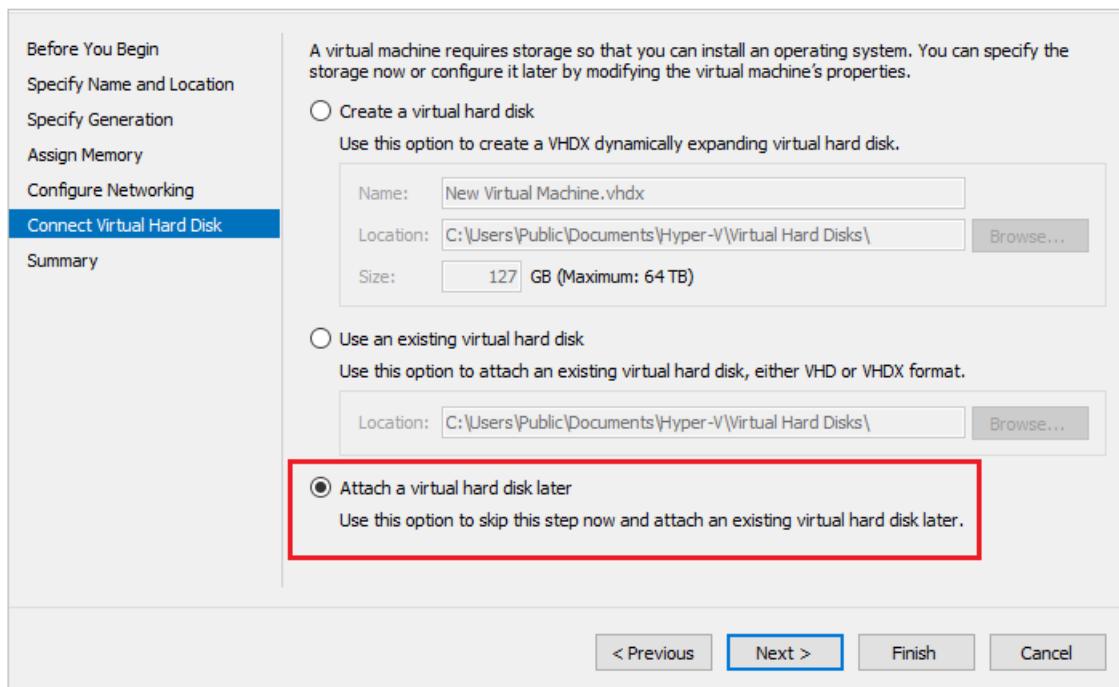
Step 1: Create a recovery VM and install Hyper-V role

1. Create a new Recovery VM:
 - Operating system: Windows Server 2016 Datacenter
 - Size: Any V3 series with at least two cores that support nested virtualization. For more information, see [Introducing the new Dv3 and Ev3 VM sizes](#).
 - Same location, Storage Account, and Resource Group as the problem VM.
 - Select the same storage type as the problem VM (Standard or Premium).
2. After the Recovery VM is created, remote desktop to the Recovery VM.
3. In Server Manager, select **Manage > Add Roles and Features**.
4. In the **Installation Type** section, select **Role-based or feature-based installation**.
5. In the **Select destination server** section, make sure that the Recovery VM is selected.
6. Select the **Hyper-V role > Add Features**.
7. Select **Next** on the **Features** section.
8. If a virtual switch is available, select it. Otherwise select **Next**.
9. On the **Migration** section, select **Next**
10. On the **Default Stores** section, select **Next**.
11. Check the box to restart the server automatically if required.
12. Select **Install**.
13. Allow the server to install the Hyper-V role. This takes a few minutes and the server will reboot automatically.

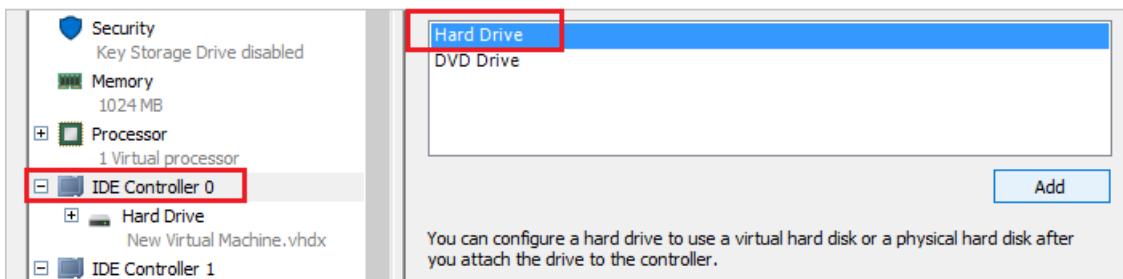
Step 2: Create the problem VM on the Recovery VM's Hyper-V server

1. Record the name of the disk in the problem VM, and then delete the problem VM. Make sure that you keep all attached disks.
2. Attach the OS disk of your problem VM as a data disk of the Recovery VM.
 - a. After the problem VM is deleted, go to the Recovery VM.
 - b. Select **Disks**, and then **Add data disk**.
 - c. Select the problem VM's disk, and then select **Save**.
3. After the disk has successfully attached, remote desktop to the Recovery VM.
4. Open Disk Management (diskmgmt.msc). Make sure that the disk of the problem VM is set to **Offline**.
5. Open Hyper-V Manager: In **Server Manager**, select the **Hyper-V role**. Right-click the server, and then select the **Hyper-V Manager**.
6. In the Hyper-V Manager, right-click the Recovery VM, and then select **New > Virtual Machine > Next**.
7. Type a name for the VM, and then select **Next**.
8. Select **Generation 1**.
9. Set the startup memory at 1024 MB or more.
10. If applicable select the Hyper-V Network Switch that was created. Else move to the next page.

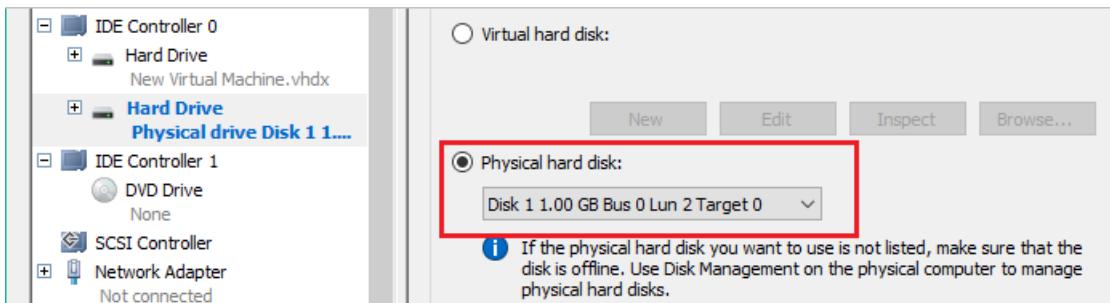
11. Select **Attach a virtual hard disk later.**



12. Select **Finish** when the VM is created.
13. Right-click the VM that you created, and then select **Settings**.
14. Select **IDE Controller 0**, select **Hard Drive**, and then click **Add**.



15. In **Physical Hard Disk**, select the disk of the problem VM that you attached to the Azure VM. If you do not see any disks listed, check if the disk is set to offline by using Disk management.



16. Select **Apply**, and then select **OK**.
17. Double-click on the VM, and then start it.
18. Now you can work on the VM as the on-premises VM. You could follow any troubleshooting steps you need.

Step 3: Re-create your Azure VM in Azure

1. After you get the VM back online, shut down the VM in the Hyper-V manager.
2. Go to the [Azure portal](#) and select the Recovery VM > Disks, copy the name of the disk. You will use the name in the next step. Detach the fixed disk from the recovery VM.
3. Go to **All resources**, search for the disk name, and then select the disk.

NAME	TYPE
VM2016_OsDisk_1_a593e978c2394355a54fe...	Disk

4. Click **Create VM**.

The screenshot shows the Azure portal interface for managing a disk. The top navigation bar includes a search bar, Save, Discard, Create snapshot, Create VM (which is highlighted with a red box), and Export options. The main content area displays details for a disk named 'VM2016_OsDisk_1_a593e978c2394355a54fe35e4c57276a'. It shows the disk is unattached and has a Premium (SSD) account type set to 128 GiB. On the left, there's a sidebar with links for Overview, Activity log, Access control (IAM), Tags, SETTINGS, and Locks.

You can also use Azure PowerShell to create the VM from the disk. For more information, see [Create the new VM from an existing disk by using PowerShell](#).

Next steps

If you are having issues connecting to your VM, see [Troubleshoot RDP connections to an Azure VM](#). For issues with accessing applications running on your VM, see [Troubleshoot application connectivity issues on a Windows VM](#).

Understand the structure and syntax of Azure Resource Manager templates

12/14/2017 • 5 min to read • [Edit Online](#)

This article describes the structure of an Azure Resource Manager template. It presents the different sections of a template and the properties that are available in those sections. The template consists of JSON and expressions that you can use to construct values for your deployment. For a step-by-step tutorial on creating a template, see [Create your first Azure Resource Manager template](#).

Template format

In its simplest structure, a template contains the following elements:

```
{  
  "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
  "contentVersion": "",  
  "parameters": { },  
  "variables": { },  
  "resources": [ ],  
  "outputs": { }  
}
```

ELEMENT NAME	REQUIRED	DESCRIPTION
\$schema	Yes	Location of the JSON schema file that describes the version of the template language. Use the URL shown in the preceding example.
contentVersion	Yes	Version of the template (such as 1.0.0.0). You can provide any value for this element. When deploying resources using the template, this value can be used to make sure that the right template is being used.
parameters	No	Values that are provided when deployment is executed to customize resource deployment.
variables	No	Values that are used as JSON fragments in the template to simplify template language expressions.
resources	Yes	Resource types that are deployed or updated in a resource group.
outputs	No	Values that are returned after deployment.

Each element contains properties you can set. The following example contains the full syntax for a template:

```
{
    "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "",
    "parameters": {
        "<parameter-name>": {
            "type" : "<type-of-parameter-value>",
            "defaultValue": "<default-value-of-parameter>",
            "allowedValues": [ "<array-of-allowed-values>" ],
            "minValue": <minimum-value-for-int>,
            "maxValue": <maximum-value-for-int>,
            "minLength": <minimum-length-for-string-or-array>,
            "maxLength": <maximum-length-for-string-or-array-parameters>,
            "metadata": {
                "description": "<description-of-the parameter>"
            }
        }
    },
    "variables": {
        "<variable-name>": "<variable-value>",
        "<variable-object-name>": {
            <variable-complex-type-value>
        },
        "<variable-object-name>": {
            "copy": [
                {
                    "name": "<name-of-array-property>",
                    "count": <number-of-iterations>,
                    "input": {
                        <properties-to-repeat>
                    }
                }
            ]
        },
        "copy": [
            {
                "name": "<variable-array-name>",
                "count": <number-of-iterations>,
                "input": {
                    <properties-to-repeat>
                }
            }
        ]
    },
    "resources": [
        {
            "condition": "<boolean-value-whether-to-deploy>",
            "apiVersion": "<api-version-of-resource>",
            "type": "<resource-provider-namespace/resource-type-name>",
            "name": "<name-of-the-resource>",
            "location": "<location-of-resource>",
            "tags": {
                "<tag-name1>": "<tag-value1>",
                "<tag-name2>": "<tag-value2>"
            },
            "comments": "<your-reference-notes>",
            "copy": {
                "name": "<name-of-copy-loop>",
                "count": "<number-of-iterations>",
                "mode": "<serial-or-parallel>",
                "batchSize": "<number-to-deploy-serially>"
            },
            "dependsOn": [
                "<array-of-related-resource-names>"
            ],
            "properties": {
                "<settings-for-the-resource>",
                "copy": [
                    {
                        "name": .
                    }
                ]
            }
        }
    ]
}
```

```

        "count": ,
        "input": {}
    }
]
},
"resources": [
    "<array-of-child-resources>"
]
}
],
"outputs": {
    "<outputName>": {
        "type" : "<type-of-output-value>",
        "value": "<output-value-expression>"
    }
}
}

```

This article describes the sections of the template in greater detail.

Syntax

The basic syntax of the template is JSON. However, expressions and functions extend the JSON values available within the template. Expressions are written within JSON string literals whose first and last characters are the brackets: [and], respectively. The value of the expression is evaluated when the template is deployed. While written as a string literal, the result of evaluating the expression can be of a different JSON type, such as an array or integer, depending on the actual expression. To have a literal string start with a bracket [, but not have it interpreted as an expression, add an extra bracket to start the string with [[.

Typically, you use expressions with functions to perform operations for configuring the deployment. Just like in JavaScript, function calls are formatted as `functionName(arg1,arg2,arg3)`. You reference properties by using the dot and [index] operators.

The following example shows how to use several functions when constructing a value:

```

"variables": {
    "storageName": "[concat(toLower(parameters('storageNamePrefix')), uniqueString(resourceGroup().id))]"
}

```

For the full list of template functions, see [Azure Resource Manager template functions](#).

Parameters

In the parameters section of the template, you specify which values you can input when deploying the resources. These parameter values enable you to customize the deployment by providing values that are tailored for a particular environment (such as dev, test, and production). You do not have to provide parameters in your template, but without parameters your template would always deploy the same resources with the same names, locations, and properties.

The following example shows a simple parameter definition:

```
"parameters": {  
    "siteNamePrefix": {  
        "type": "string",  
        "metadata": {  
            "description": "The name prefix of the web app that you wish to create."  
        }  
    },  
},
```

For information about defining parameters, see [Parameters section of Azure Resource Manager templates](#).

Variables

In the variables section, you construct values that can be used throughout your template. You do not need to define variables, but they often simplify your template by reducing complex expressions.

The following example shows a simple variable definition:

```
"variables": {  
    "webSiteName": "[concat(parameters('siteNamePrefix'), uniqueString(resourceGroup().id))]",  
},
```

For information about defining variables, see [Variables section of Azure Resource Manager templates](#).

Resources

In the resources section, you define the resources that are deployed or updated. This section can get complicated because you must understand the types you are deploying to provide the right values.

```
"resources": [  
    {  
        "apiVersion": "2016-08-01",  
        "name": "[variables('webSiteName')]",  
        "type": "Microsoft.Web/sites",  
        "location": "[resourceGroup().location]",  
        "properties": {  
            "serverFarmId": "/subscriptions/<subscription-id>/resourcegroups/<resource-group-name>/providers/Microsoft.Web/serverFarms/<plan-name>"  
        }  
    }  
,
```

For more information, see [Resources section of Azure Resource Manager templates](#).

Outputs

In the Outputs section, you specify values that are returned from deployment. For example, you could return the URI to access a deployed resource.

```
"outputs": {  
    "newHostName": {  
        "type": "string",  
        "value": "[reference(variables('webSiteName')).defaultHostName]"  
    }  
},
```

For more information, see [Outputs section of Azure Resource Manager templates](#).

Template limits

Limit the size of your template to 1 MB, and each parameter file to 64 KB. The 1-MB limit applies to the final state of the template after it has been expanded with iterative resource definitions, and values for variables and parameters.

You are also limited to:

- 256 parameters
- 256 variables
- 800 resources (including copy count)
- 64 output values
- 24,576 characters in a template expression

You can exceed some template limits by using a nested template. For more information, see [Using linked templates when deploying Azure resources](#). To reduce the number of parameters, variables, or outputs, you can combine several values into an object. For more information, see [Objects as parameters](#).

Next steps

- To view complete templates for many different types of solutions, see the [Azure Quickstart Templates](#).
- For details about the functions you can use from within a template, see [Azure Resource Manager Template Functions](#).
- To combine multiple templates during deployment, see [Using linked templates with Azure Resource Manager](#).
- You may need to use resources that exist within a different resource group. This scenario is common when working with storage accounts or virtual networks that are shared across multiple resource groups. For more information, see the [resourceId function](#).

Frequently asked question about Windows Virtual Machines

10/24/2017 • 4 min to read • [Edit Online](#)

This article addresses some common questions about Windows virtual machines created in Azure using the Resource Manager deployment model. For the Linux version of this topic, see [Frequently asked question about Linux Virtual Machines](#)

What can I run on an Azure VM?

All subscribers can run server software on an Azure virtual machine. For information about the support policy for running Microsoft server software in Azure, see [Microsoft server software support for Azure Virtual Machines](#)

Certain versions of Windows 7, Windows 8.1, and Windows 10 are available to MSDN Azure benefit subscribers and MSDN Dev and Test Pay-As-You-Go subscribers, for development and test tasks. For details, including instructions and limitations, see [Windows Client images for MSDN subscribers](#).

How much storage can I use with a virtual machine?

Each data disk can be up to 4 TB (4,095 GB). The number of data disks you can use depends on the size of the virtual machine. For details, see [Sizes for Virtual Machines](#).

Azure Managed Disks are the recommended disk storage offerings for use with Azure Virtual Machines for persistent storage of data. You can use multiple Managed Disks with each Virtual Machine. Managed Disks offer two types of durable storage options: Premium and Standard Managed Disks. For pricing information, see [Managed Disks Pricing](#).

Azure storage accounts can also provide storage for the operating system disk and any data disks. Each disk is a .vhdx file stored as a page blob. For pricing details, see [Storage Pricing Details](#).

How can I access my virtual machine?

Establish a remote connection using Remote Desktop Connection (RDP) for a Windows VM. For instructions, see [How to connect and log on to an Azure virtual machine running Windows](#). A maximum of two concurrent connections are supported, unless the server is configured as a Remote Desktop Services session host.

If you're having problems with Remote Desktop, see [Troubleshoot Remote Desktop connections to a Windows-based Azure Virtual Machine](#).

If you're familiar with Hyper-V, you might be looking for a tool similar to VMConnect. Azure doesn't offer a similar tool because console access to a virtual machine isn't supported.

Can I use the temporary disk (the D: drive by default) to store data?

Don't use the temporary disk to store data. It is only temporary storage, so you would risk losing data that can't be recovered. Data loss can occur when the virtual machine moves to a different host. Resizing a virtual machine, updating the host, or a hardware failure on the host are some of the reasons a virtual machine might move.

If you have an application that needs to use the D: drive letter, you can reassign drive letters so that the temporary disk uses something other than D:. For instructions, see [Change the drive letter of the Windows temporary disk](#).

How can I change the drive letter of the temporary disk?

You can change the drive letter by moving the page file and reassigning drive letters, but you need to make sure you do the steps in a specific order. For instructions, see [Change the drive letter of the Windows temporary disk](#).

Can I add an existing VM to an availability set?

No. If you want your VM to be part of an availability set, you need to create the VM within the set. There currently isn't a way to add a VM to an availability set after it has been created.

Can I upload a virtual machine to Azure?

Yes. For instructions, see [Migrating on-premises VMs to Azure](#).

Can I resize the OS disk?

Yes. For instructions, see [How to expand the OS drive of a Virtual Machine in an Azure Resource Group](#).

Can I copy or clone an existing Azure VM?

Yes. Using managed images, you can create an image of a virtual machine and then use the image to build multiple new VMs. For instructions, see [Create a custom image of a VM](#).

Why am I not seeing Canada Central and Canada East regions through Azure Resource Manager?

The two new regions of Canada Central and Canada East are not automatically registered for virtual machine creation for existing Azure subscriptions. This registration is done automatically when a virtual machine is deployed through the Azure portal to any other region using Azure Resource Manager. After a virtual machine is deployed to any other Azure region, the new regions should be available for subsequent virtual machines.

Does Azure support Linux VMs?

Yes. To quickly create a Linux VM to try out, see [Create a Linux VM on Azure using the Portal](#).

Can I add a NIC to my VM after it's created?

Yes, this is now possible. The VM first needs to be stopped deallocated. Then you can add or remove a NIC (unless it's the last NIC on the VM).

Are there any computer name requirements?

Yes. The computer name can be a maximum of 15 characters in length. See [Naming conventions rules and restrictions](#) for more information around naming your resources.

Are there any resource group name requirements?

Yes. The resource group name can be a maximum of 90 characters in length. See [Naming conventions rules and restrictions](#) for more information about resource groups.

What are the username requirements when creating a VM?

Usernames can be a maximum of 20 characters in length and cannot end in a period (".").

The following usernames are not allowed:

administrator	admin	user	user1
test	user2	test1	user3
admin1	1	123	a
actuser	adm	admin2	aspnet
backup	console	david	guest
john	owner	root	server
sql	support	support_388945a0	sys
test2	test3	user4	user5

What are the password requirements when creating a VM?

Passwords must be 12 - 123 characters in length and meet 3 out of the following 4 complexity requirements:

- Have lower characters
- Have upper characters
- Have a digit
- Have a special character (Regex match [\W_])

The following passwords are not allowed:

abc@123	P@\$\$w0rd	P@ssw0rd	P@ssword123	Pa\$\$word
pass@word1	Password!	Password1	Password22	iloveyou!