

Table of Contents

[Overview](#)

[About VMs](#)

[Storage](#)

[Virtual networks](#)

[Ports, endpoints and security](#)

[Load balancers](#)

[VM sizes](#)

[Compute-intensive sizes](#)

[Compute benchmark scores](#)

[Regions and availability](#)

[VM availability](#)

[Security Center](#)

[Resource Manager](#)

[Deployment models](#)

[VM Scale Sets](#)

[Containers](#)

[FAQ](#)

[Get started](#)

[Create a VM using the portal](#)

[Log on to a VM](#)

[Install a role and open ports](#)

[Different ways to create a VM](#)

[Install Azure PowerShell](#)

[Create a VM with PowerShell](#)

[Template walkthrough](#)

[Create and deploy complex VM templates](#)

[Application architecture](#)

[Access and security](#)

[Availability and scale](#)

Application deployment

How to

Use Storage

Attach a data disk

Detach a data disk

Expand the OS disk

Use D: as a data disk

Disk encryption

Network

Allow access to a VM using the portal

Allow access to a VM using PowerShell

Create an FDQN using the portal

Create a VM with multiple NICs

Create VNETs using the portal

Create NSGs using the portal

Create a load balancer

Create a static public IP

Connect Classic VNets to Resource Manager VNets

Use Azure DNS with VMs

Use Azure Traffic Manager with VMs

Deploy

Use Software Assurance licensing

Find VM images using Powershell

Use Windows client images

Use a template to create a VM

Scale multiple VMs with VMSS

Create multiple Azure virtual machines

Create a VM with monitoring and diagnostics

Deploy using Visual Studio

Deploy using C#

Deploy using C# and templates

Deploy application frameworks from a template

Images

Configure

[Create availability set](#)

[Change the availability set for a VM](#)

[Move a VM between subscriptions](#)

[Resize a VM](#)

[Reset a Windows password for Azure VM](#)

[Use tags](#)

[Tag a VM](#)

[Powershell common tasks](#)

[CLI common tasks](#)

Manage

[Create a work or school identity in Azure AD](#)

[Manage access](#)

[Set up Key Vault](#)

[Set up WinRM access](#)

[Backup using Recovery Services](#)

[Manage backups using PowerShell](#)

[Resource Manager policies](#)

[Manage VMs using PowerShell](#)

[Manage VMs using C#](#)

[Manage VMs using CLI](#)

Automate

[VM Extensions overview](#)

[Custom Script extension](#)

[PowerShell DSC extension](#)

[Azure Log Collector extension](#)

[Azure diagnostics extension](#)

[OMS agent extension](#)

[Extensions in templates](#)

[Exporting virtual machine extensions](#)

[Configuration samples](#)

[Troubleshoot extensions](#)

[Azure Automation overview](#)

[Download the template for a VM](#)

[Automate with Chef](#)

Migrate

[Overview of migration](#)

[Plan for migration](#)

[Migrate using PowerShell](#)

[Common migration errors](#)

[Community tools for migrating](#)

Plan

[Windows VM best practices](#)

[Infrastructure guidelines](#)

[Subscriptions and accounts](#)

[Naming](#)

[Resource groups](#)

[Storage](#)

[Networking](#)

[Availability sets](#)

[Infrastructure example](#)

[Azure planned maintenance](#)

Manage workloads

[High-performance Computing \(HPC\)](#)

[MATLAB](#)

[MongoDB](#)

[SQL Server](#)

[Deploying SAP IDES EHP7 SP3 for SAP ERP 6.0 on Microsoft Azure](#)

[SAP](#)

[SharePoint](#)

[Set up a web-based LOB application in a hybrid cloud for testing](#)

[Set up a simulated hybrid cloud environment for testing](#)

Troubleshoot

[Troubleshoot Remote Desktop connections](#)

[Reset RDP password](#)

[Troubleshooting specific RDP error messages](#)

[Troubleshoot creating a new VM](#)

[Troubleshoot restarting or resizing a VM](#)

[Troubleshoot application access](#)

[Troubleshoot allocation failures](#)

[Redeploy a VM](#)

[Attach virtual hard disk to troubleshooting VM](#)

Reference

[PowerShell](#)

[Azure CLI 2.0 \(Preview\)](#)

[.NET](#)

[Java](#)

[Node.js](#)

[Python](#)

[Compute REST](#)

Resources

[Author Resource Manager templates](#)

[Community templates](#)

[Pricing](#)

[Regional availability](#)

[Stack Overflow](#)

[Videos](#)

Overview of Windows virtual machines in Azure

1/17/2017 • 7 min to read • [Edit on GitHub](#)

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 Manage Azure Virtual Machines using Resource Manager and PowerShell .
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](#).
- Make sure you follow the [Best Practices for running a Windows VM on Azure](#).

About disks and VHDs for Azure virtual machines

1/17/2017 • 5 min to read • [Edit on GitHub](#)

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. 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.

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 1023 gigabytes (GB).

Temporary disk

The temporary disk is automatically created for you. The temporary disk is labeled as the D: drive by default and it used for storing pagefile.sys.

The size of the temporary disk varies, based on the size of the virtual machine. For more information, see [Sizes for Windows virtual machines](#).

WARNING

Don't store data on the temporary disk. It provides temporary storage for applications and processes and is intended to only store data such as page or swap files. To remap this disk to a different drive letter, see [Change the drive letter of the Windows temporary disk](#).

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 1023 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 more information about page blobs, see [Understanding block blobs and page blobs](#). For more information about premium storage, see [Premium storage: High-performance storage for Azure virtual machine workloads](#).

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 information, 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. 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.

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
```

Next steps

- [Attach a disk](#) to add additional storage for your VM.

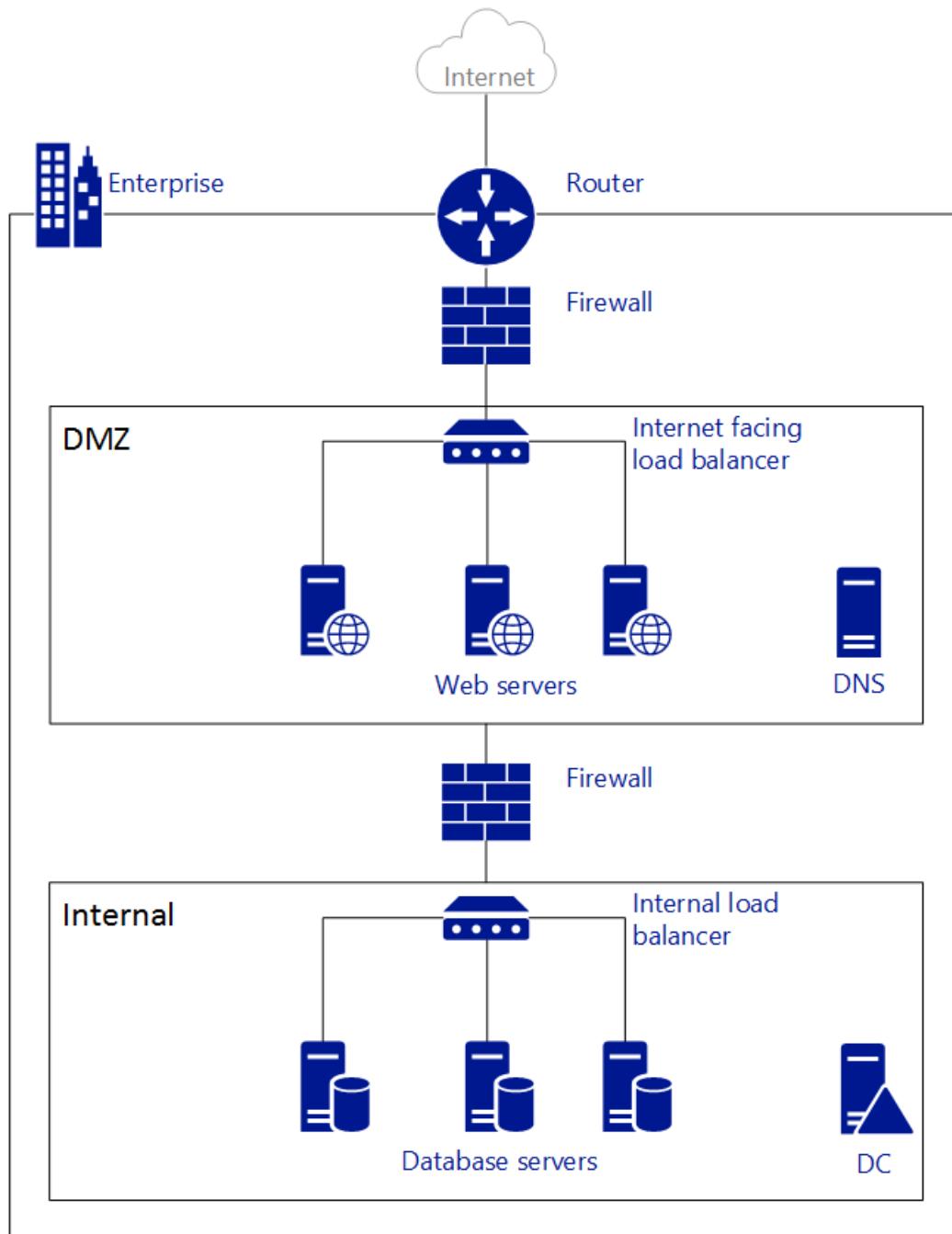
- [Upload a Windows VM image to Azure](#) to use when creating a new VM.
- [Change the drive letter of the Windows temporary disk](#) so your application can use the D: drive for data.

Virtual networks

1/17/2017 • 5 min to read • [Edit on GitHub](#)

An Azure virtual network (VNet) is a representation of your own network in the cloud. It is a logical isolation of the Azure cloud dedicated to your subscription. You can fully control the IP address blocks, DNS settings, security policies, and route tables within this network. You can also further segment your VNet into subnets and launch Azure IaaS virtual machines (VMs) and/or [Cloud services \(PaaS role instances\)](#). Additionally, you can connect the virtual network to your on-premises network using one of the [connectivity options](#) available in Azure. In essence, you can expand your network to Azure, with complete control on IP address blocks with the benefit of enterprise scale Azure provides.

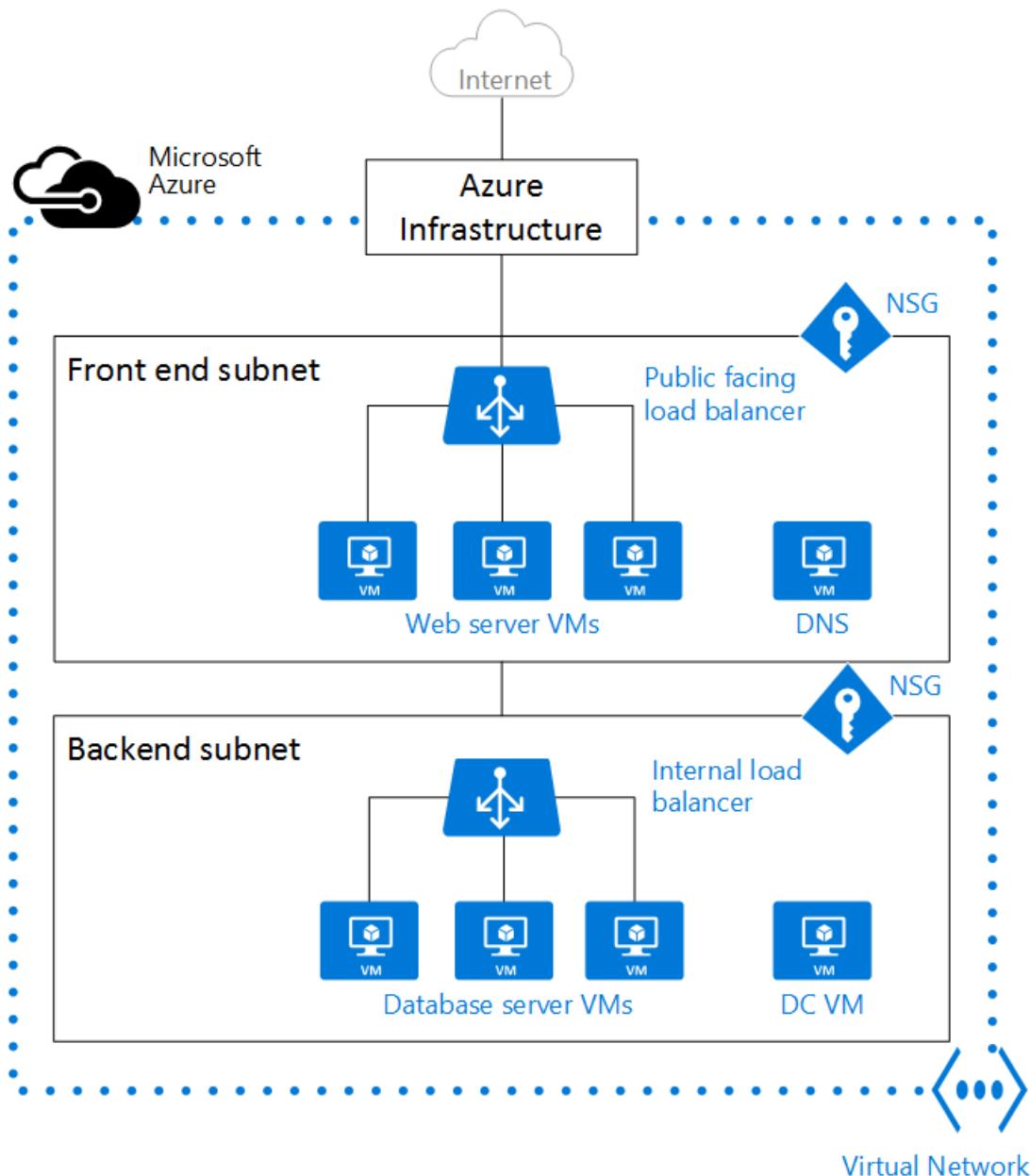
To better understand VNets, take a look at the figure below, which shows a simplified on-premises network.



The figure above shows an on-premises network connected to the public Internet through a router. You can also see a firewall between the router and a DMZ hosting a DNS server and a web server farm. The web server farm is

load balanced using a hardware load balancer that is exposed to the Internet, and consumes resources from the internal subnet. The internal subnet is separated from the DMZ by another firewall, and hosts Active Directory Domain Controller servers, database servers, and application servers.

The same network can be hosted in Azure as shown in the figure below.



Notice how the Azure infrastructure takes on the role of the router, allowing access from your VNet to the public Internet without the need of any configuration. Firewalls can be substituted by Network Security Groups (NSGs) applied to each individual subnet. And physical load balancers are substituted by internet facing and internal load balancers in Azure.

NOTE

There are two deployment modes in Azure: classic (also known as Service Management) and Azure Resource Manager (ARM). Classic VNets could be added to an affinity group, or created as a regional VNet. If you have a VNet in an affinity group, it is recommended to [migrate it to a regional VNet](#).

Benefits

- **Isolation.** VNets are completely isolated from one another. That allows you to create disjoint networks for

development, testing, and production that use the same CIDR address blocks.

- **Access to the public Internet.** All IaaS VMs and PaaS role instances in a VNet can access the public Internet by default. You can control access by using Network Security Groups (NSGs).
- **Access to VMs within the VNet.** PaaS role instances and IaaS VMs can be launched in the same virtual network and they can connect to each other using private IP addresses even if they are in different subnets without the need to configure a gateway or use public IP addresses.
- **Name resolution.** Azure provides internal name resolution for IaaS VMs and PaaS role instances deployed in your VNet. You can also deploy your own DNS servers and configure the VNet to use them.
- **Security.** Traffic entering and exiting the virtual machines and PaaS role instances in a VNet can be controlled using Network Security groups.
- **Connectivity.** VNets can be connected to each other using network gateways or VNet peering. VNets can be connected to on-premises data centers through site-to-site VPN networks or Azure ExpressRoute. To learn more about site-to-site VPN connectivity, visit [About VPN gateways](#). To learn more about ExpressRoute, visit [ExpressRoute technical overview](#). To learn more about VNet peering, visit [VNet peering](#).

NOTE

Make sure you create a VNet before deploying any IaaS VMs or PaaS role instances to your Azure environment. ARM based VMs require a VNet, and if you do not specify an existing VNet, Azure creates a default VNet that might have a CIDR address block clash with your on-premises network. Making it impossible for you to connect your VNet to your on-premises network.

Subnets

Subnet is a range of IP addresses in the VNet, you can divide a VNet into multiple subnets for organization and security. VMs and PaaS role instances deployed to subnets (same or different) within a VNet can communicate with each other without any extra configuration. You can also configure route tables and NSGs to a subnet.

IP addresses

There are two types of IP addresses assigned to resources in Azure: *public* and *private*. Public IP Addresses allow Azure resources to communicate with Internet and other Azure public-facing services like [Azure Redis Cache](#), [Azure Event Hubs](#). Private IP Addresses allows communication between resources in a virtual network, along with those connected through a VPN, without using an Internet-routable IP addresses.

To learn more about IP addresses in Azure, visit [IP addresses in virtual network](#)

Azure load balancers

Virtual machines and cloud services in a Virtual network can be exposed to Internet using Azure Load balancers. Line of Business applications that are internal facing only can be load balanced using Internal load balancer.

- **External load balancer.** You can use an external load balancer to provide high availability for IaaS VMs and PaaS role instances accessed from the public Internet.
- **Internal load balancer.** You can use an internal load balancer to provide high availability for IaaS VMs and PaaS role instances accessed from other services in your VNet.

To learn more about load balancing in Azure, visit [Load balancer overview](#).

Network Security Groups (NSG)

You can create NSGs to control inbound and outbound access to network interfaces (NICs), VMs, and subnets. Each NSG contains one or more rules specifying whether or not traffic is approved or denied based on source IP

address, source port, destination IP address, and destination port. To learn more about NSGs, visit [What is a Network Security Group](#).

Virtual appliances

A virtual appliance is just another VM in your VNet that runs a software based appliance function, such as firewall, WAN optimization, or intrusion detection. You can create a route in Azure to route your VNet traffic through a virtual appliance to use its capabilities.

For instance, NSGs can be used to provide security on your VNet. However, NSGs provide layer 4 Access Control List (ACL) to incoming and outgoing packets. If you want to use a layer 7 security model, you need to use a firewall appliance.

Virtual appliances depend on [user defined routes and IP forwarding](#).

Limits

There are limits on the number of Virtual Networks allowed in a subscription, please refer to [Azure Networking limits](#) for more information.

Pricing

There is no extra cost for using Virtual Networks in Azure. The compute instances launched within the Vnet will be charged the standard rates as described in [Azure VM Pricing](#). The [VPN Gateways](#) and [Public IP Addresses](#) used in the VNet will also be charged standard rates.

Next steps

- [Create a VNet](#) and subnets.
- [Create a VM in a VNet](#).
- Learn about [NSGs](#).
- Learn about [user defined routes and IP forwarding](#).

Classic Endpoints in Resource Manager

1/17/2017 • 7 min to read • [Edit on GitHub](#)

The approach to Azure endpoints works a little differently between the Classic and Resource Manager deployment models. In the Resource Manager deployment model, you now have the flexibility to create network filters that control the flow of traffic in and out of your VMs. These filters allow you to create complex networking environments beyond a simple endpoint as in the Classic deployment model. This article provides an overview of Network Security Groups and how they differ from using Classic endpoints, creating these filtering rules, and sample deployment scenarios.

Overview of Resource Manager deployments

Endpoints in the Classic deployment model are replaced by Network Security Groups and access control list (ACL) rules. Quick steps for implementing Network Security Group ACL rules are:

- Create a Network Security Group.
- To allow or deny traffic, define your Network Security Group ACL rules.
- Assign your Network Security Group to a network interface or virtual network subnet.

If you are wanting to also perform port-forwarding, you need to place a load balancer in front of your VM and use NAT rules. Quick steps for implementing a load balancer and NAT rules would be as follows:

- Create a load balancer.
- Create a backend pool and add your VMs to the pool.
- Define your NAT rules for the required port forwarding.
- Assign your NAT rules to your VMs.

Network Security Group overview

Network Security Groups are a new feature that provides a layer of security for you to allow specific ports and subnets to access your VMs. You typically always have a Network Security Group providing this layer of security between your VMs and the outside world. Network Security Groups can be applied to a virtual network subnet or a specific network interface for a VM. Rather than creating endpoint ACL rules, you now create Network Security Group ACL rules. These ACL rules provide much greater control than simply creating an endpoint to forward a given port. For more information, see [What is a Network Security Group?](#).

TIP

You can assign Network Security Groups to multiple subnets or network interfaces. There is no 1:1 mapping, meaning that you can create a Network Security Group with a common set of ACL rules and apply to multiple subnets or network interfaces. Further, Network Security Group can be applied to resources across your subscription (based on [Role Based Access Controls](#)).

Load Balancers overview

In the Classic deployment model, Azure would perform all the Network Address Translation (NAT) and port forwarding on a Cloud Service for you. When creating an endpoint, you would specify the external port to expose along with the internal port to direct traffic to. Network Security Groups by themselves do not perform this same NAT and port forwarding.

To allow you to create NAT rules for such port forwarding, create an Azure load balancer in your resource group. Again, the load balancer is granular enough to only apply to specific VMs if needed. The Azure load balancer NAT rules work alongside Network Security Group ACL rules to provide much more flexibility and control than was achievable using Cloud Service endpoints. For more information, see the [load balancer overview](#).

Network Security Group ACL rules

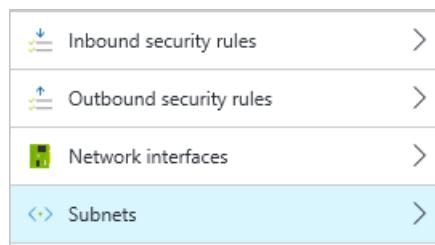
ACL rules let you define what traffic can flow in and out of your VM based on specific ports, port ranges, or protocols. Rules are assigned to individual VMs or to a subnet. The following screenshot is an example of ACL rules for a common webserver:

PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION	...
100	SSH	Any	Any	SSH (TCP/22)	Allow	...
110	HTTP	Any	Any	HTTP (TCP/80)	Allow	...
120	HTTPS	Any	Any	HTTPS (TCP/443)	Allow	...
65000	AllowVnetInBound	VirtualNetwork	VirtualNetwork	Custom (ANY/Any)	Allow	...
65001	AllowAzureLoadBalancerInBound	AzureLoadBalancer	Any	Custom (ANY/Any)	Allow	...
65500	DenyAllInBound	Any	Any	Custom (ANY/Any)	Deny	...

ACL rules are applied based on a priority metric that you specify - the higher the value, the lower the priority. Every Network Security Group has three default rules that are designed to handle the flow of Azure networking traffic, with an explicit `DenyAllInbound` as the final rule. Default ACL rules are given a low priority to not interfere with rules you create.

Assigning Network Security Groups

You assign a Network Security Group to a subnet or a network interface. This approach allows you to be as granular as needed when applying your ACL rules to only a specific VM, or ensure a common set of ACL rules are applied to all VMs part of a subnet:



The behavior of the Network Security Group doesn't change depending on being assigned to a subnet or a network interface. A common deployment scenario has the Network Security Group assigned to a subnet to ensure compliance of all VMs attached to that subnet. For more information, see [applying Network Security groups to resources](#).

Default behavior of Network Security Groups

Depending on how and when you create your Network Security Group, default rules may be created for Windows

VMs to permit RDP access on TCP port 3389. Default rules for Linux VMs permit SSH access on TCP port 22. These automatic ACL rules are created under the following conditions:

- If you create a Windows VM through the portal and accept the default action to create a Network Security Group, an ACL rule to allow TCP port 3389 (RDP) is created.
- If you create a Linux VM through the portal and accept the default action to create a Network Security Group, an ACL rule to allow TCP port 22 (SSH) is created.

Under all other conditions, these default ACL rules are not created. You cannot connect to your VM without creating the appropriate ACL rules. These conditions include the following common actions:

- Creating a Network Security Group through the portal as a separate action to creating the VM.
- Creating a Network Security Group programmatically through PowerShell, Azure CLI, Rest APIs, etc.
- Creating a VM and assigning it to an existing Network Security Group that does not already have the appropriate ACL rule defined.

In all the preceding cases, you need to create ACL rules for your VM to allow the appropriate remote management connections.

Default behavior of a VM without a Network Security Group

You can create a VM without creating a Network Security Group. In these situations, you can connect to your VM using RDP or SSH without creating any ACL rules. Similarly, if you installed a web service on port 80, that service is automatically accessible remotely. The VM has all ports open.

NOTE

You still need to have a public IP address assigned to a VM in order for any remote connections. Not having a Network Security Group for the subnet or network interface doesn't expose the VM to any external traffic. The default action when creating a VM through the portal is to create a new public IP. For all other forms of creating a VM such as PowerShell, Azure CLI, or Resource Manager template, a public IP is not automatically created unless explicitly requested. The default action through the portal is also to create a Network Security Group. This default action means you shouldn't end up in a situation with an exposed VM that has no network filtering in place.

Understanding Load Balancers and NAT rules

In the Classic deployment model, you could create endpoints that also performed port forwarding. When you create a VM in the Classic deployment model, ACL rules for RDP or SSH would be automatically created. These rules would not expose TCP port 3389 or TCP port 22 respectively to the outside world. Instead, a high-value TCP port would be exposed that maps to the appropriate internal port. You could also create your own ACL rules in a similar manner, such as expose a webserver on TCP port 4280 to the outside world. You can see these ACL rules and port mappings in the following screenshot from the Classic portal:

Cloud Services				
DASHBOARD	MONITOR	ENDPOINTS	CONFIGURE	
NAME	PROTOCOL	PUBLIC PORT	PRIVATE PORT	
PowerShell	TCP	5986	5986	
Remote Desktop	TCP	53514	3389	
HTTP	TCP	4280	80	

With Network Security Groups, that port-forwarding function is handled by a load balancer. For more information, see [load balancers in Azure](#). The following example shows a load balancer with a NAT rule to perform port-forwarding TCP port 4222 to the internal TCP port 22 a VM:

The screenshot shows two windows side-by-side. The left window is titled 'Inbound NAT rules' and shows a list of rules for a load balancer named 'TestLB'. One rule is selected, 'TestNATRule', which maps port 4222 on the external IP 13.88.10.204 to port 22 on the internal VM 'TestVM'. The right window is titled 'TestNATRule' and provides detailed configuration for this specific rule. It shows the destination as 13.88.10.204 (TestPIP), service as Custom (TCP/4222), protocol as TCP, port as 4222, target as TestVM, and floating IP as Disabled. A note at the bottom indicates that traffic to the VM must be allowed via a Network Security Group ACL rule.

NOTE

When you implement a load balancer, you typically don't assign the VM itself a public IP address. Instead, the load balancer has a public IP address assigned to it. You still need to create your Network Security Group and ACL rules to define the flow of traffic in and out of your VM. The load balancer NAT rules are simply to define what ports are allowed through the load balancer and how they get distributed across the backend VMs. As such, you need to create a NAT rule for traffic to flow through the load balancer. To allow the traffic to reach the VM, create a Network Security Group ACL rule.

Next steps

You can read more detailed information on the technologies discussed here, along with quickstart guides for using Network Security Groups, in the following articles:

- [Quick-start - Create a Network Security Group and ACL rules using the Azure portal](#)
- [Quick-start - Create a Network Security Group and ACL rules using the Azure PowerShell](#)
- [Azure Resource Manager overview](#)
- [What is a Network Security Group \(NSG\)?](#)
- [Azure Resource Manager Overview for Load Balancers](#)

Load balancing for Azure infrastructure services

1/17/2017 • 4 min to read • [Edit on GitHub](#)

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.

There are two levels of load balancing available for Azure infrastructure services:

- **DNS Level:** Load balancing for traffic to different cloud services located in different data centers, to different Azure websites located in different data centers, or to external endpoints. This is done with Azure Traffic Manager and the Round Robin load balancing method.
- **Network Level:** Load balancing of incoming Internet traffic to different virtual machines of a cloud service, or load balancing of traffic between virtual machines in a cloud service or virtual network. This is done with the Azure load balancer.

Traffic Manager load balancing for cloud services and websites

Traffic Manager allows you to control the distribution of user traffic to endpoints, which can include cloud services, websites, external sites, and other Traffic Manager profiles. Traffic Manager works by applying an intelligent policy engine to Domain Name System (DNS) queries for the domain names of your Internet resources. Your cloud services or websites can be running in different datacenters across the world.

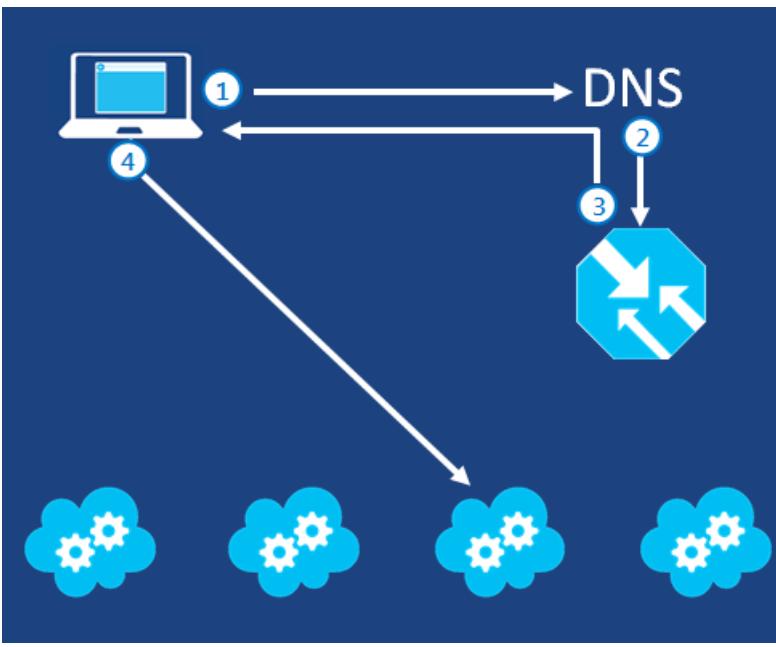
You must use either REST or Windows PowerShell to configure external endpoints or Traffic Manager profiles as endpoints.

Traffic Manager uses three load-balancing methods to distribute traffic:

- **Failover:** Use this method when you want to use a primary endpoint for all traffic, but provide backups in case the primary becomes unavailable.
- **Performance:** Use this method when you have endpoints in different geographic locations and you want requesting clients to use the "closest" endpoint in terms of the lowest latency.
- **Round Robin:** Use this method when you want to distribute load across a set of cloud services in the same datacenter or across cloud services or websites in different datacenters.

For more information, see [About Traffic Manager Load Balancing Methods](#).

The following diagram shows an example of the Round Robin load balancing method for distributing traffic between different cloud services.



The basic process is the following:

1. An Internet client queries a domain name corresponding to a web service.
2. DNS forwards the name query request to Traffic Manager.
3. Traffic Manager chooses the next cloud service in the Round Robin list and sends back the DNS name. The Internet client's DNS server resolves the name to an IP address and sends it to the Internet client.
4. The Internet client connects with the cloud service chosen by Traffic Manager.

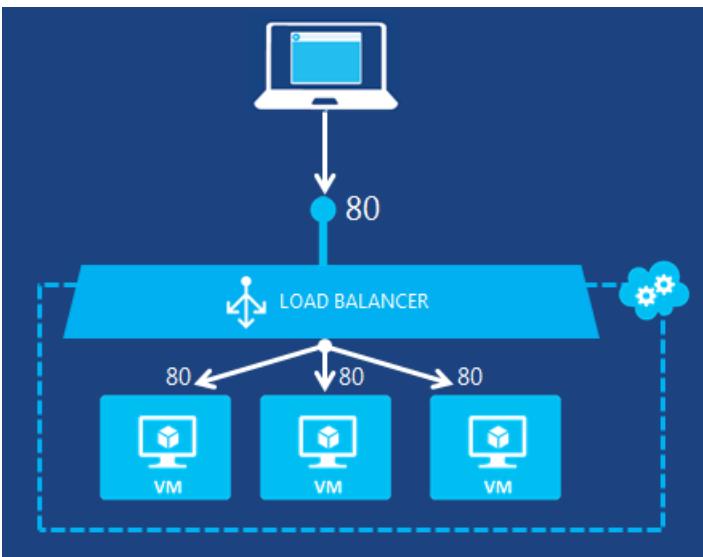
For more information, see [Traffic Manager](#).

Azure load balancing for virtual machines

Virtual machines in the same cloud service or virtual network can communicate with each other directly using their private IP addresses. Computers and services outside the cloud service or virtual network can only communicate with virtual machines in a cloud service or virtual network with a configured endpoint. An endpoint is a mapping of a public IP address and port to that private IP address and port of a virtual machine or web role within an Azure cloud service.

The Azure Load Balancer randomly distributes a specific type of incoming traffic across multiple virtual machines or services in a configuration known as a load-balanced set. For example, you can spread the load of web request traffic across multiple web servers or web roles.

The following diagram shows a load-balanced endpoint for standard (unencrypted) web traffic that is shared among three virtual machines for the public and private TCP port of 80. These three virtual machines are in a load-balanced set.



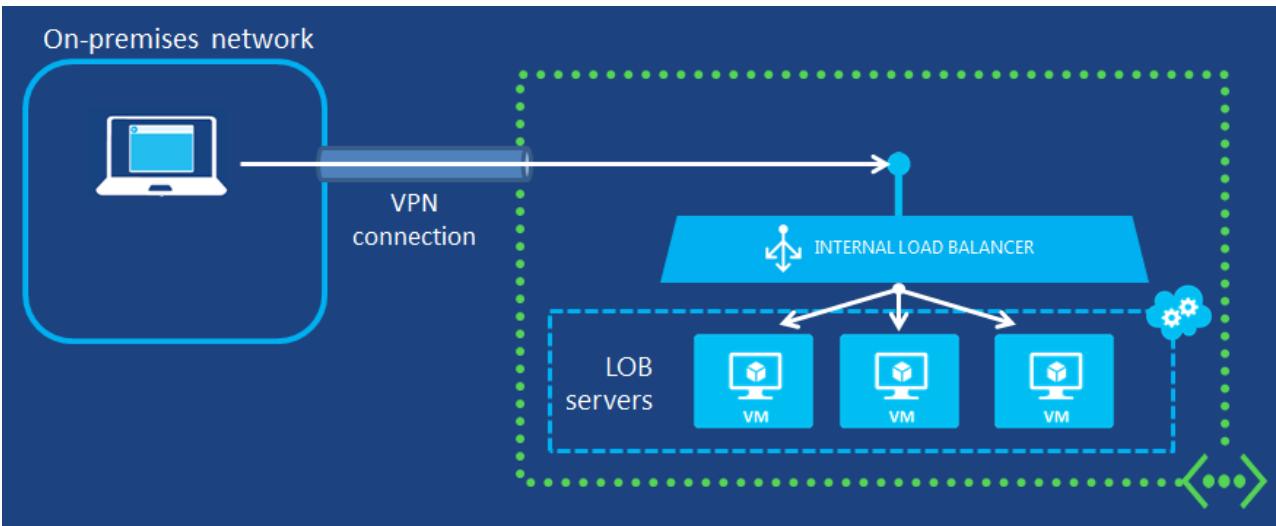
For more information, see [Azure Load Balancer](#). For the steps to create a load-balanced set, see [Configure a load-balanced set](#).

Azure can also load balance within a cloud service or virtual network. This is known as internal load balancing and can be used in the following ways:

- To load balance between servers in different tiers of a multi-tier application (for example, between web and database tiers).
- To load balance line-of-business (LOB) applications hosted in Azure without requiring additional load balancer hardware or software.
- To include on-premises servers in the set of computers whose traffic is load balanced.

Similar to Azure load balancing, internal load balancing is facilitated by configuring an internal load-balanced set.

The following diagram shows an example of an internal load-balanced endpoint for a line of business (LOB) application that is shared among three virtual machines in a cross-premises virtual network.



Load balancer considerations

A load balancer is configured by default to timeout an idle session in 4 minutes. If your application behind a load balancer leaves a connection idle for more than 4 minutes and it doesn't have a Keep-Alive configuration, the connection will be dropped. You can change the load balancer behavior to allow a [longer timeout setting for Azure load balancer](#).

Other consideration is the type of distribution mode supported by Azure Load Balancer. You can configure source IP affinity (source IP, destination IP) or source IP protocol (source IP , destination IP and protocol). Check out [Azure](#)

[Load Balancer distribution mode \(source IP affinity\)](#) for more information.

Next steps

For the steps to create a load-balanced set, see [Configure an internal load-balanced set](#).

For more information about load balancer, see [Internal load balancing](#).

Sizes for virtual machines in Azure

1/17/2017 • 16 min to read • [Edit on GitHub](#)

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](#).

IMPORTANT

- For information about pricing of the various sizes, see [Virtual Machines Pricing](#).
- For availability of VM sizes in Azure regions, see [Products available by region](#).
- To see general limits on Azure VMs, see [Azure subscription and service limits, quotas, and constraints](#).

There are multiple standard sizes to choose from on Azure. Considerations for some of these sizes include:

- 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-core 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](#).
- 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.
- 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. At a lower per-hour list price, the F-series is the best value in price-performance in the Azure portfolio based on the Azure Compute Unit (ACU) per core.

The F-series also introduces a new standard in VM size naming for Azure. For this series and VM sizes released in the future, the numeric value after the family name letter will match the number of CPU cores. Additional capabilities, such as optimized for premium storage, will be designated by letters following the numeric CPU core count. This naming format will be used for future VM sizes released but will not retroactively change the names of any existing VM sizes which have been released.

- G-series VMs offer the most memory and run on hosts that have Intel Xeon E5 V3 family processors.
- DS-series, DSv2-series, Fs-series and GS-series VMs can use Premium Storage, which provides high-performance, low-latency storage for I/O intensive workloads. These VMs use solid-state drives (SSDs) to host a virtual machine's disks and also provide a local SSD disk cache. Premium Storage is available in certain regions. For details, see [Premium Storage: High-performance storage for Azure virtual machine workloads](#).
- 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.
- 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.

The size of the virtual machine affects the pricing. The size also affects the processing, memory, and storage

capacity of the virtual machine. Storage costs are calculated separately based on used pages in the storage account. For details, see [Virtual Machines Pricing Details](#) and [Azure Storage Pricing](#).

The following considerations might help you decide on a size:

- 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. For detailed information and considerations about using these sizes, see [About the H-series and compute-intensive A-series VMs](#).
- Dv2-series, D-series, G-series, and the DS/GS counterparts are ideal for applications that demand faster CPUs, better local disk performance, or have higher memory demands. They offer a powerful combination for many enterprise-grade applications.
- The F-series VMs are an excellent choice for workloads that demand faster CPUs but do not need as much memory or local SSD per CPU core. Workloads such as analytics, gaming servers, web servers, and batch processing will benefit from the value of the F-series.
- Some of the physical hosts in Azure data centers may not support larger virtual machine sizes, such as A5 – A11. As a result, you may see the error message **Failed to configure virtual machine** or **Failed to create virtual machine** when resizing an existing virtual machine to a new size; creating a new virtual machine in a virtual network created before April 16, 2013; or adding a new virtual machine to an existing cloud service. See [Error: "Failed to configure virtual machine"](#) on the support forum for workarounds for each deployment scenario.
- Your subscription might also limit the number of cores you can deploy in certain size families. To increase a quota, contact Azure Support.

Performance considerations

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/CORE
Standard_A0	50
Standard_A1-4	100
Standard_A5-7	100
Standard_A1-8v2	100
Standard_A2m-8mv2	100
A8-A11	225*

SKU FAMILY	ACU/CORE
D1-14	160
D1-15v2	210 - 250*
DS1-14	160
DS1-15v2	210-250*
F1-F16	210-250*
F1s-F16s	210-250*
G1-5	180 - 240*
GS1-5	180 - 240*
H	290 - 300*

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.

Size tables

The following tables show the sizes and the capacities they provide.

- 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**.
- Maximum network bandwidth is the maximum aggregated bandwidth allocated and assigned per VM type. The maximum bandwidth provides guidance for selecting the right VM type to ensure adequate network capacity is available. When moving between Low, Moderate, High and Very High, the throughput will increase accordingly. Actual network performance will depend on many factors including network and application loads, and application network settings.

A-series

SIZE	CPU CORES	MEMORY: GiB	LOCAL HDD: GiB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_A0	1	0.768	20	1	1x500	1 / low
Standard_A1	1	1.75	70	2	2x500	1 / moderate
Standard_A2	2	3.5	135	4	4x500	1 / moderate

SIZE	CPU CORES	MEMORY: GIB	LOCAL HDD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_A3	4	7	285	8	8x500	2 / high
Standard_A4	8	14	605	16	16x500	4 / high
Standard_A5	2	14	135	4	4x500	1 / moderate
Standard_A6	4	28	285	8	8x500	2 / high
Standard_A7	8	56	605	16	16x500	4 / high

A-series - compute-intensive instances

For information and considerations about using these sizes, see [About the H-series and compute-intensive A-series VMs](#).

SIZE	CPU CORES	MEMORY: GIB	LOCAL HDD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_A8*	8	56	382	16	16x500	2 / high
Standard_A9*	16	112	382	16	16x500	4 / very high
Standard_A10	8	56	382	16	16x500	2 / high
Standard_A11	16	112	382	16	16x500	4 / very high

*RDMA capable

Av2-series

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_A1_v2	1	2	10	2	2x500	1 / moderate
Standard_A2_v2	2	4	20	4	4x500	2 / moderate
Standard_A4_v2	4	8	40	8	8x500	4 / high

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_A8_v2	8	16	80	16	16x500	8 / high
Standard_A2_m_v2	2	16	20	4	4x500	2 / moderate
Standard_A4_m_v2	4	32	40	8	8x500	4 / high
Standard_A8_m_v2	8	64	80	16	16x500	8 / high

D-series

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_D1	1	3.5	50	2	2x500	1 / moderate
Standard_D2	2	7	100	4	4x500	2 / high
Standard_D3	4	14	200	8	8x500	4 / high
Standard_D4	8	28	400	16	16x500	8 / high
Standard_D1_1	2	14	100	4	4x500	2 / high
Standard_D1_2	4	28	200	8	8x500	4 / high
Standard_D1_3	8	56	400	16	16x500	8 / high
Standard_D1_4	16	112	800	32	32x500	8 / very high

Dv2-series

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_D1_v2	1	3.5	50	2	2x500	1 / moderate

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DATA DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_D2_v2	2	7	100	4	4x500	2 / high
Standard_D3_v2	4	14	200	8	8x500	4 / high
Standard_D4_v2	8	28	400	16	16x500	8 / high
Standard_D5_v2	16	56	800	32	32x500	8 / extremely high
Standard_D1_1_v2	2	14	100	4	4x500	2 / high
Standard_D1_2_v2	4	28	200	8	8x500	4 / high
Standard_D1_3_v2	8	56	400	16	16x500	8 / high
Standard_D1_4_v2	16	112	800	32	32x500	8 / extremely high
Standard_D1_5_v2	20	140	1,000	40	40x500	8 / extremely high*

*In some regions, accelerated networking is available for the Standard_D15_v2 size. For more information about usage and availability, see [Accelerated Networking is in Preview](#) and [Accelerated Networking for a virtual machine](#).

DS-series*

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX CACHED DISK THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_DS1	1	3.5	7	2	4,000 / 32 (43)	3,200 / 32	1 / moderate
Standard_DS2	2	7	14	4	8,000 / 64 (86)	6,400 / 64	2 / high
Standard_DS3	4	14	28	8	16,000 / 128 (172)	12,800 / 128	4 / high

SIZE	CPU CORES	MEMORY: GiB	LOCAL SSD: GiB	MAX DATA DISKS	MAX CACHED DISK THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GiB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_D S4	8	28	56	16	32,000 / 256 (344)	25,600 / 256	8 / high
Standard_D S11	2	14	28	4	8,000 / 64 (72)	6,400 / 64	2 / high
Standard_D S12	4	28	56	8	16,000 / 128 (144)	12,800 / 128	4 / high
Standard_D S13	8	56	112	16	32,000 / 256 (288)	25,600 / 256	8 / high
Standard_D S14	16	112	224	32	64,000 / 512 (576)	51,200 / 512	8 / very high

MBps = 10^6 bytes per second, and GiB = 1024^3 bytes.

*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](#).

DSv2-series*

SIZE	CPU CORES	MEMORY: GiB	LOCAL SSD: GiB	MAX DATA DISKS	MAX CACHED DISK THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GiB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_D S1_v2	1	3.5	7	2	4,000 / 32 (43)	3,200 / 48	1 moderate
Standard_D S2_v2	2	7	14	4	8,000 / 64 (86)	6,400 / 96	2 high
Standard_D S3_v2	4	14	28	8	16,000 / 128 (172)	12,800 / 192	4 high
Standard_D S4_v2	8	28	56	16	32,000 / 256 (344)	25,600 / 384	8 high
Standard_D S5_v2	16	56	112	32	64,000 / 512 (688)	51,200 / 768	8 extremely high
Standard_D S11_v2	2	14	28	4	8,000 / 64 (72)	6,400 / 96	2 high

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX CACHED DISK THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GIB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_D_S12_v2	4	28	56	8	16,000 / 128 (144)	12,800 / 192	4 high
Standard_D_S13_v2	8	56	112	16	32,000 / 256 (288)	25,600 / 384	8 high
Standard_D_S14_v2	16	112	224	32	64,000 / 512 (576)	51,200 / 768	8 extremely high
Standard_D_S15_v2	20	140	280	40	80,000 / 640 (720)	64,000 / 960	8 extremely high**

MBps = 10^6 bytes per second, and GiB = 1024^3 bytes.

*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](#).

**In some regions, accelerated networking is available for the Standard_DS15_v2 size. For more information about usage and availability, see [Accelerated Networking is in Preview](#) and [Accelerated Networking for a virtual machine](#).

F-series

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_F1	1	2	16	2	2x500	1 / moderate
Standard_F2	2	4	32	4	4x500	2 / high
Standard_F4	4	8	64	8	8x500	4 / high
Standard_F8	8	16	128	16	16x500	8 / high
Standard_F16	16	32	256	32	32x500	8 / extremely high

Fs-series*

SIZE	CPU CORES	MEMORY: GiB	LOCAL SSD: GiB	MAX DATA DISKS	MAX CACHED DISK THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GiB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_F1s	1	2	4	2	4,000 / 32 (12)	3,200 / 48	1 / moderate
Standard_F2s	2	4	8	4	8,000 / 64 (24)	6,400 / 96	2 / high
Standard_F4s	4	8	16	8	16,000 / 128 (48)	12,800 / 192	4 / high
Standard_F8s	8	16	32	16	32,000 / 256 (96)	25,600 / 384	8 / high
Standard_F16s	16	32	64	32	64,000 / 512 (192)	51,200 / 768	8 / extremely high

MBps = 10^6 bytes per second, and GiB = 1024^3 bytes.

*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](#).

G-series

SIZE	CPU CORES	MEMORY: GiB	LOCAL SSD: GiB	MAX DATA DISKS	MAX DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_G1	2	28	384	4	4 x 500	1 / high
Standard_G2	4	56	768	8	8 x 500	2 / high
Standard_G3	8	112	1,536	16	16 x 500	4 / very high
Standard_G4	16	224	3,072	32	32 x 500	8 / extremely high
Standard_G5	32	448	6,144	64	64 x 500	8 / extremely high

GS-series*

SIZE	CPU CORES	MEMORY: GiB	LOCAL SSD: GiB	MAX DATA DISKS	MAX CACHED DISK THROUGHPUT: IOPS / MBPS (CACHE SIZE IN GiB)	MAX UNCACHED DISK THROUGHPUT: IOPS / MBPS	MAX NICs / NETWORK BANDWIDTH
Standard_G S1	2	28	56	4	10,000 / 100 (264)	5,000 / 125	1 / high
Standard_G S2	4	56	112	8	20,000 / 200 (528)	10,000 / 250	2 / High
Standard_G S3	8	112	224	16	40,000 / 400 (1,056)	20,000 / 500	4 / very high
Standard_G S4	16	224	448	32	80,000 / 800 (2,112)	40,000 / 1,000	8 / extremely high
Standard_G S5	32	448	896	64	160,000 / 1,600 (4,224)	80,000 / 2,000	8 / extremely high

MBps = 10^6 bytes per second, and GiB = 1024^3 bytes.

*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).

H-series

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 core VMs are built on the Intel Haswell E5-2667 V3 processor technology featuring DDR4 memory and local SSD based 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.

For information and considerations about using these sizes, see [About the H-series and compute-intensive A-series VMs](#).

SIZE	CPU CORES	MEMORY: GiB	LOCAL SSD: GiB	MAX DATA DISKS	MAX DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_H8	8	56	1000	16	16 x 500	2 / high
Standard_H16	16	112	2000	32	32 x 500	4 / very high
Standard_H8m	8	112	1000	16	16 x 500	2 / high

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	MAX DATA DISKS	MAX DISK THROUGHPUT: IOPS	MAX NICs / NETWORK BANDWIDTH
Standard_H1_6m	16	224	2000	32	32 x 500	4 / very high
Standard_H1_6r*	16	112	2000	32	32 x 500	4 / very high
Standard_H1_6mr*	16	224	2000	32	32 x 500	4 / very high

*RDMA capable

N-series

The NC and NV sizes are also known as GPU-enabled instances. These are specialized virtual machines that include NVIDIA's GPU cards, optimized for different scenarios and use cases. The NV sizes are optimized and designed for remote visualization, streaming, gaming, encoding and VDI scenarios utilizing frameworks such as OpenGL and DirectX. The NC sizes are more optimized for compute-intensive and network-intensive applications and algorithms, including CUDA- and OpenCL-based applications and simulations.

NV instances

The NV instances are powered by NVIDIA's Tesla M60 GPU card and NVIDIA GRID for desktop accelerated applications and virtual desktops where customers will be able to visualize their data or simulations. Users will be 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. The Tesla M60 delivers 4096 CUDA cores in a dual-GPU design with up to 36 streams of 1080p H.264.

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	GPU
Standard_NV6	6	56	380	1
Standard_NV12	12	112	680	2
Standard_NV24	24	224	1440	4

1 GPU = one-half M60 card.

Supported operating systems

- Windows Server 2016, Windows Server 2012 R2 - see [N-series driver setup for Windows](#)

NC instances

The NC instances are powered by NVIDIA's Tesla K80 card. Users can now crunch through data much faster by leveraging CUDA for energy exploration applications, crash simulations, ray traced rendering, deep learning and more. The Tesla K80 delivers 4992 CUDA cores with a dual-GPU design, up to 2.91 Teraflops of double-precision and up to 8.93 Teraflops of single-precision performance.

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	GPU
Standard_NC6	6	56	380	1

SIZE	CPU CORES	MEMORY: GIB	LOCAL SSD: GIB	GPU
Standard_NC12	12	112	680	2
Standard_NC24	24	224	1440	4
Standard_NC24r*	24	224	1440	4

1 GPU = one-half K80 card.

*RDMA capable

Supported operating systems

- Windows Server 2016, Windows Server 2012 R2 - see [N-series driver setup for Windows](#)
- Ubuntu 16.04 LTS - see [N-series driver setup for Linux](#)

Notes: 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

Next steps

- Learn about [azure subscription and service limits, quotas, and constraints](#).
- Learn more [about the H-series and compute-intensive A-series VMs](#) for workloads like High-performance Computing (HPC).

About H-series and compute-intensive A-series VMs

1/17/2017 • 5 min to read • [Edit on GitHub](#)

Here is background information and some considerations for using the newer Azure H-series and the earlier A8, A9, A10, and A11 instances, also known as *compute-intensive* instances. This article focuses on using these instances for Windows VMs. This article is also available for [Linux VMs](#).

For basic specs, storage capacities, and disk details, see [Sizes for virtual machines](#).

Key features

- **High-performance hardware** – These instances are designed and optimized for compute-intensive and network-intensive applications, including high-performance computing (HPC) and batch applications, modeling, and large-scale simulations.
- Details about the Intel Xeon E5-2667 v3 processor (used in the H-series) and Intel Xeon E5-2670 processor (in A8 - A11), including supported instruction set extensions, are at the Intel.com website.
- **Designed for HPC clusters** – Deploy multiple compute-intensive instances in Azure to create a stand-alone HPC cluster or to add capacity to an on-premises cluster. If you want to, deploy cluster management and job scheduling tools. Or, use the instances for compute-intensive work in another Azure service such as Azure Batch.
- **RDMA network connection for MPI applications** – A subset of the compute-intensive instances (H16r, H16mr, A8, and A9) feature a second 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 RDMA-capable instances to communicate with each other over an InfiniBand network, operating at FDR rates for H16r and H16mr virtual machines, and QDR rates for A8 and A9 virtual machines. The RDMA capabilities exposed in these virtual machines can boost the scalability and performance of certain Linux and Windows Message Passing Interface (MPI) applications. See [Access to the RDMA network](#) in this article for requirements.

Deployment considerations

- **Azure subscription** – If you want to deploy more than a small number of 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** - The compute-intensive 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 of 20 cores per subscription (if you use the classic deployment model) or 20 cores per region (if you use the Resource Manager deployment model). 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, you may need at least a cloud-based Azure virtual network for many deployment scenarios, or a site-to-site connection if you need to access on-premises resources such as an application license server. If one is 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.
- **Cloud service or availability set** – To use the Azure RDMA network, deploy the RDMA-capable VMs in the same cloud service (if you use the classic deployment model) or the same availability set (if you use the Azure Resource Manager deployment model). If you use Azure Batch, the RDMA-capable VMs must be in the same pool.
- **Resizing** – Because of the specialized hardware used in the compute-intensive instances, 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 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.

Access to the RDMA network

To access the Azure RDMA network for Windows MPI traffic, RDMA-capable instances must meet the following requirements:

- **Operating system**
 - **Virtual machines** - Windows Server 2012 R2, Windows Server 2012
 - **Cloud services** - Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 Guest OS family
- **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.

- **HpcVmDrivers VM extension** - On RDMA-capable VMs, the HpcVmDrivers extension must be added to install Windows network device drivers that enable RDMA connectivity. (In some deployments of A8 and A9 instances, the HpcVmDrivers extension is added automatically.) If you need to add the VM extension to a VM, you can use [Azure PowerShell](#) cmdlets for Azure Resource Manager.

To get information about the latest HpcVmDrivers extension:

```
Get-AzureVMAvailableExtension -ExtensionName "HpcVmDrivers"
```

To install the latest version 1.1 HpcVmDrivers extension on an existing RDMA-capable VM named myVM:

```
Set-AzureRmVMExtension -ResourceGroupName "myResourceGroup" -Location "westus" -VMName "myVM" -ExtensionName "HpcVmDrivers" -Publisher "Microsoft.HpcCompute" -Type "HpcVmDrivers" -TypeHandlerVersion "1.1"
```

For more information, see [Manage VM extensions](#). You can also work with extensions for VMs in the [classic deployment model](#).

Considerations for HPC Pack and Windows

[Microsoft HPC Pack](#), Microsoft's free HPC cluster and job management solution, is not required for you to use the compute-intensive instances with Windows Server. However, it 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 can use the Azure RDMA network when deployed on RDMA-capable VMs.

For more information and 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](#).

Next steps

- For details about availability and pricing of the compute-intensive sizes, see [Virtual Machines pricing](#) and [Cloud Services pricing](#).
- For storage capacities and disk details, see [Sizes for virtual machines](#).
- To get started deploying and using compute-intensive instances with HPC Pack on Windows, see [Set up a Windows RDMA cluster with HPC Pack to run MPI applications](#).
- For information about using compute-intensive instances to run MPI applications with Azure Batch, see [Use multi-instance tasks to run Message Passing Interface \(MPI\) applications in Azure Batch](#).

Compute benchmark scores for Windows VMs

1/17/2017 • 2 min to read • [Edit on GitHub](#)

The following SPECInt benchmark scores show compute performance for Azure's high-performance VM lineup running Windows Server. Compute benchmark scores are also available for [Linux VMs](#).

A-series - compute-intensive

SIZE	VCPUS	NUMA NODES	CPU	RUNS	AVG BASE RATE	STDDEV
Standard_A8	8	1	Intel Xeon CPU E5-2670 0 @ 2.6 GHz	10	236.1	1.1
Standard_A9	16	2	Intel Xeon CPU E5-2670 0 @ 2.6 GHz	10	450.3	7.0
Standard_A10	8	1	Intel Xeon CPU E5-2670 0 @ 2.6 GHz	5	235.6	0.9
Standard_A11	16	2	Intel Xeon CPU E5-2670 0 @ 2.6 GHz	7	454.7	4.8

Dv2-series

SIZE	VCPUS	NUMA NODES	CPU	RUNS	AVG BASE RATE	STDDEV
Standard_D1_v2	1	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	83	36.6	2.6
Standard_D2_v2	2	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	27	70.0	3.7
Standard_D3_v2	4	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	19	130.5	4.4
Standard_D4_v2	8	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	19	238.1	5.2
Standard_D5_v2	16	2	Intel Xeon E5-2673 v3 @ 2.4 GHz	14	460.9	15.4

SIZE	VCPUS	NUMA NODES	CPU	RUNS	Avg Base Rate	StdDev
Standard_D11_v2	2	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	19	70.1	3.7
Standard_D12_v2	4	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	2	132.0	1.4
Standard_D13_v2	8	1	Intel Xeon E5-2673 v3 @ 2.4 GHz	17	235.8	3.8
Standard_D14_v2	16	2	Intel Xeon E5-2673 v3 @ 2.4 GHz	15	460.8	6.5

G-series, GS-series

SIZE	VCPUS	NUMA NODES	CPU	RUNS	Avg Base Rate	StdDev
Standard_G1, Standard_GS1	2	1	Intel Xeon E5-2698B v3 @ 2 GHz	31	71.8	6.5
Standard_G2, Standard_GS2	4	1	Intel Xeon E5-2698B v3 @ 2 GHz	5	133.4	13.0
Standard_G3, Standard_GS3	8	1	Intel Xeon E5-2698B v3 @ 2 GHz	6	242.3	6.0
Standard_G4, Standard_GS4	16	1	Intel Xeon E5-2698B v3 @ 2 GHz	15	398.9	6.0
Standard_G5, Standard_GS5	32	2	Intel Xeon E5-2698B v3 @ 2 GHz	22	762.8	3.7

H-series

SIZE	VCPUS	NUMA NODES	CPU	RUNS	Avg Base Rate	StdDev
Standard_H8	8	1	Intel Xeon E5-2667 v3 @ 3.2 GHz	5	297.4	0.9
Standard_H16	16	2	Intel Xeon E5-2667 v3 @ 3.2 GHz	5	575.8	6.8

SIZE	VCPUS	NUMA NODES	CPU	RUNS	Avg Base Rate	STDDEV
Standard_H8m	8	1	Intel Xeon E5-2667 v3 @ 3.2 GHz	5	297.0	1.2
Standard_H16m	16	2	Intel Xeon E5-2667 v3 @ 3.2 GHz	5	572.2	3.9
Standard_H16r	16	2	Intel Xeon E5-2667 v3 @ 3.2 GHz	5	573.2	2.9
Standard_H16mr	16	2	Intel Xeon E5-2667 v3 @ 3.2 GHz	7	569.6	2.8

About SPECint

Windows numbers were computed by running [SPECint 2006](#) on Windows Server. SPECint was run using the base rate option (SPECint_rate2006), with one copy per core. SPECint consists of 12 separate tests, each run three times, taking the median value from each test and weighting them to form a composite score. Those tests were then run across multiple VMs to provide the average scores shown.

Next steps

- For storage capacities, disk details, and additional considerations for choosing among VM sizes, see [Sizes for virtual machines](#).

Regions and availability for virtual machines in Azure

1/17/2017 • 6 min to read • [Edit on GitHub](#)

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. 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. This article provides you with an overview of the availability and redundancy features of Azure.

What are Azure regions?

Azure allows you to create resources, such as VMs, in defined geographic regions like 'West US', 'North Europe', or 'Southeast Asia'. There are currently 30 Azure regions around the world. 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 when building your applications to create VMs closest to your users and to meet any legal, compliance, or tax purposes.

Special Azure regions

There are some special Azure regions for compliance or legal purposes that you may wish to use when building out your applications. 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](#).

- **Central India, South India, and West India**

- These regions are currently available to volume licensing customers and partners with a local enrollment in India. In 2016, users can access them if the users have purchased direct online subscriptions.

- **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 currently 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](#).

Storage availability

Understanding Azure regions and geographies becomes important when you consider the available Azure Storage replication options. When you create a storage account, you must select one of the following replication options:

- 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
Number of copies of data maintained on separate nodes.	3	3	6	6

You can read more about [Azure Storage replication options here](#).

Storage costs

Prices vary depending on the storage type and availability that you select.

- 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.

Azure images

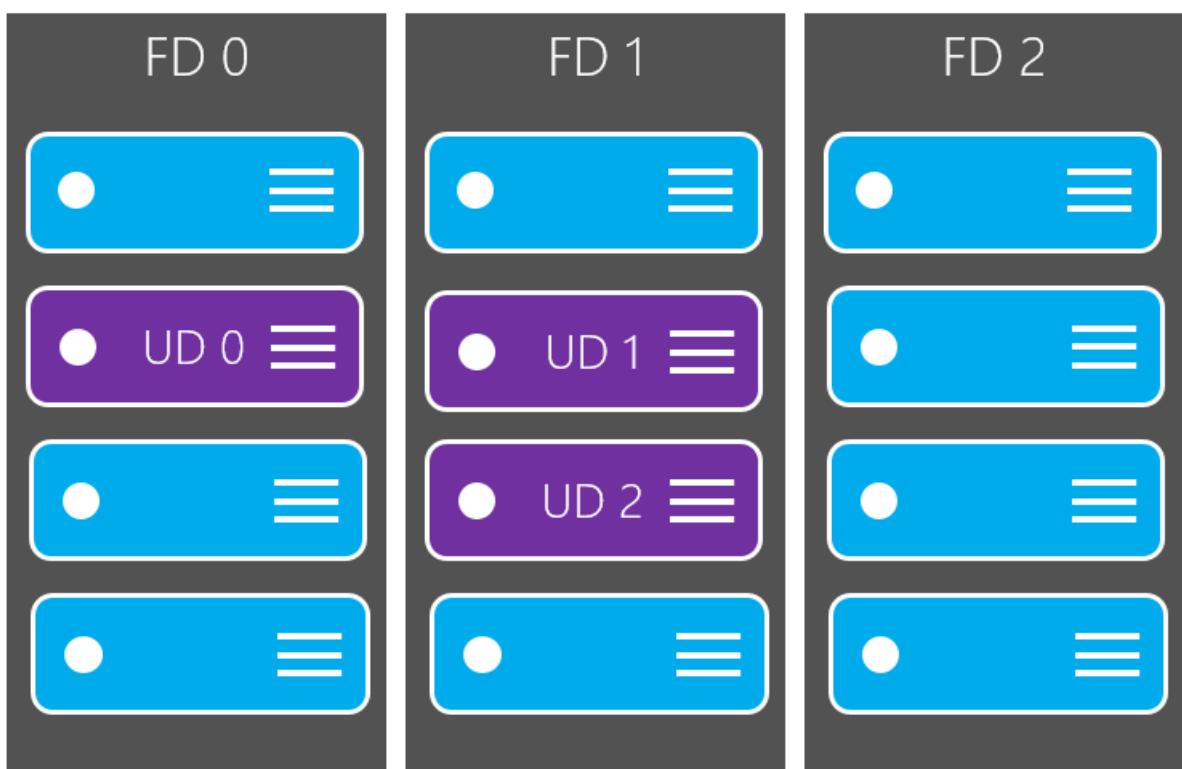
In Azure, VMs are created from an image. Typically, images are from the [Azure Marketplace](#) where partners can provide pre-configured complete OS or application images.

When you create a VM from an image in the Azure Marketplace, you are actually working with templates. Azure Resource Manager templates are declarative JavaScript Object Notation (JSON) files that can be used to create complex application environments comprising VMs, storage, virtual networking, etc. You can read more about using [Azure Resource Manager templates](#), including how to [build your own templates](#).

You can also create your own custom images and upload them using [Azure CLI](#) or [Azure PowerShell](#) to quickly create custom VMs to your specific build requirements.

Availability sets

An availability set is a logical grouping of VMs that allows Azure to understand how your application is built to provide for redundancy and availability. It is recommended 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](#). The availability set is comprised 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.

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](#).

Manage the availability of virtual machines

1/17/2017 • 5 min to read • [Edit on GitHub](#)

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](#).

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.

For instructions on creating and using availability sets when using the classic deployment model, see [How to Configure an Availability Set](#).

Understand planned vs. unplanned maintenance

There are two types of Microsoft Azure platform events that can affect the availability of your virtual machines: planned maintenance and unplanned maintenance.

- **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. However, there are instances where these updates require a reboot of your virtual machine to apply the required updates to the platform infrastructure.
- **Unplanned maintenance events** occur when the hardware or 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 will automatically migrate your virtual machine from the unhealthy physical machine hosting your virtual machine to a healthy physical machine. Such events are rare, but may also cause your virtual machine to reboot.

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](#)
- [Configure each application tier into separate availability sets](#)
- [Combine a Load Balancer with availability sets](#)
- [Use multiple storage accounts for each availability set](#)

Configure multiple virtual machines in an availability set for redundancy

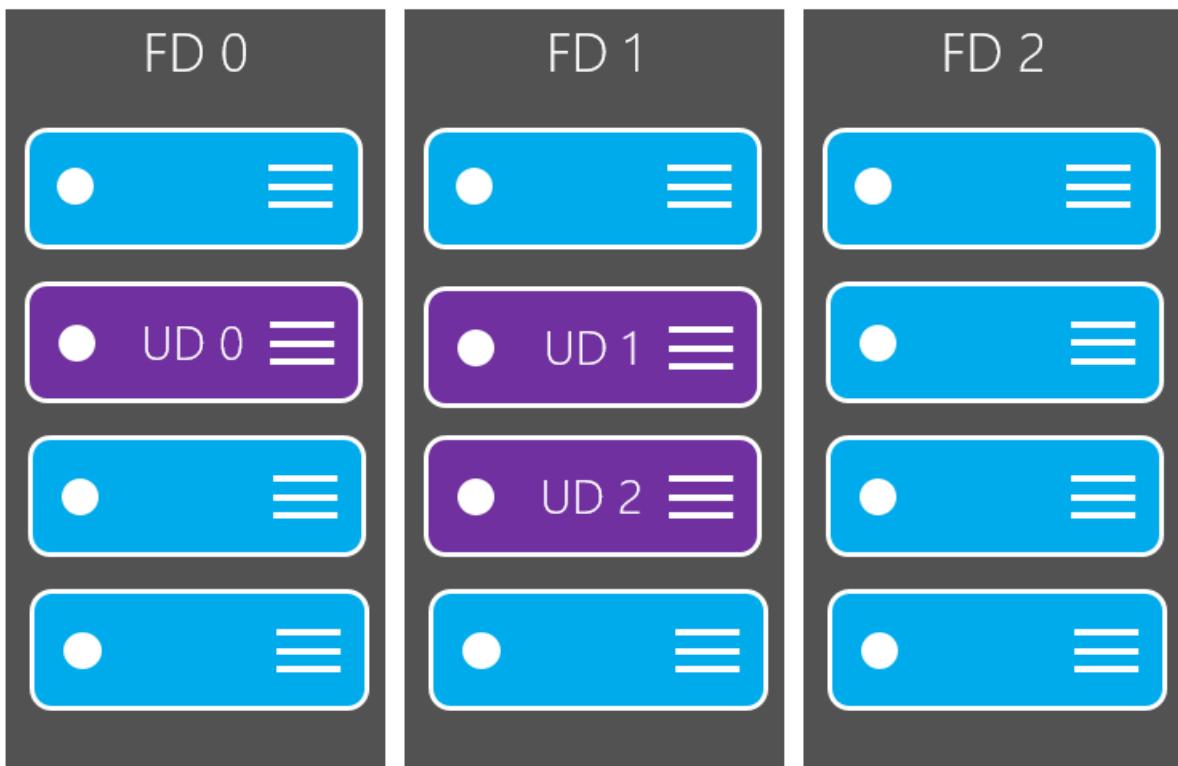
To provide redundancy to your application, we recommend that you group two or more virtual machines in an availability set. This configuration ensures that during either a planned or unplanned maintenance event, at least one virtual machine will be available and meet the 99.95% Azure SLA. For more information, see the [SLA for Virtual Machines](#).

IMPORTANT

Avoid leaving a single instance virtual machine in an availability set by itself. Virtual machines in this configuration do not qualify for a SLA guarantee and will face downtime during Azure planned maintenance events.

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.

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.

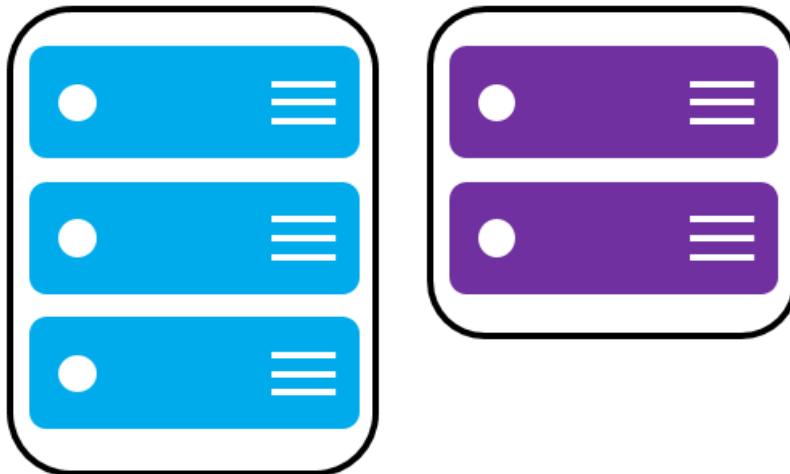


Configure each application tier into separate availability sets

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 will be 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 multiple storage accounts for each availability set

There are best practices to be followed with regards to the storage accounts used by the Virtual Hard Disks (VHDs) within the VM. Each disk (VHD) is a page blob in an Azure Storage account. It is important to ensure that there is redundancy and isolation across the storage accounts in order to provide high availability for the VMs within the Availability Set.

1. **Keep all disks (OS and data) associated with a VM in the same storage account**
2. **Storage account limits should be considered** when adding more VHDs to a storage account
3. **Use separate storage account for each VM in an Availability Set.** Multiple VMs in the same availability set must NOT share the same storage account. It is acceptable for VMs across different Availability Sets to share storage accounts as long as the best practices above are followed

Next steps

To learn more about load balancing your virtual machines, see [Load Balancing virtual machines](#).

Azure Security Center and Azure Virtual Machines

1/17/2017 • 4 min to read • [Edit on GitHub](#)

[Azure Security Center](#) helps you prevent, detect, and respond to threats. It 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.

This article shows how Security Center can help you secure your Azure Virtual Machines (VM).

Why use Security Center?

Security Center helps you safeguard virtual machine data in Azure by providing visibility into your virtual machine's security settings. When Security Center safeguards your VMs, the following capabilities will be available:

- Operating System (OS) security settings with the recommended configuration rules
- System security and critical updates that are missing
- Endpoint protection recommendations
- Disk encryption validation
- Vulnerability assessment and remediation
- Threat detection

In addition to helping protect your Azure VMs, Security Center also provides security monitoring and management for Cloud Services, App Services, Virtual Networks, and more.

NOTE

See [Introduction to Azure Security Center](#) to learn more about Azure Security Center.

Prerequisites

To get started with Azure Security Center, you'll need to know and consider the following:

- You must have a subscription to Microsoft Azure. See [Security Center Pricing](#) for more information on Security Center's free and standard tiers.
- Plan your Security Center adoption, see [Azure Security Center planning and operations guide](#) to learn more about planning and operations considerations.
- For information regarding operating system supportability, see [Azure Security Center frequently asked questions \(FAQ\)](#).

Set security policy

Data collection needs to be enabled so that Azure Security Center can gather the information it needs to provide recommendations and alerts that are generated based on the security policy you configure. In the figure below, you can see that **Data collection** has been turned **On**.

A security policy defines the set of controls which are recommended for resources within the specified subscription or resource group. Before enabling security policy, you must have data collection enabled. Security Center collects data from your virtual machines in order to assess their security state, provide security recommendations, and alert you to threats. In Security Center, you define policies for your Azure subscriptions or resource groups according to your company's security needs and the type of applications or sensitivity of the data in each subscription.

The screenshot shows two side-by-side policy configuration windows. On the left, under 'Security policy', 'Data collection' is turned 'On'. On the right, under 'Prevention policy', several recommendations are listed with their status: System updates (On), OS vulnerabilities (On), Endpoint protection (On), Disk encryption (On), Network security groups (On), Web application firewall (On), Next generation firewall (On), Vulnerability Assessment (On), SQL auditing & Threat detection (On), and SQL transparent data encryption (On).

NOTE

To learn more about each **Prevention policy** available, see [Set security policies](#) article.

Manage security recommendations

Security Center analyzes the security state of your Azure resources. When Security Center identifies potential security vulnerabilities, it creates recommendations. The recommendations guide you through the process of configuring the needed controls.

After setting a security policy, Security Center analyzes the security state of your resources to identify potential vulnerabilities. The recommendations are shown in a table format where each line represents one particular recommendation. The table below provides some examples of recommendations for Azure VMs and what each one will do if you apply it. When you select a recommendation, you will be provided information that shows you how to implement the recommendation in Security Center.

RECOMMENDATION	DESCRIPTION
Enable data collection for subscriptions	Recommends that you turn on data collection in the security policy for each of your subscriptions and all virtual machines (VMs) in your subscriptions.
Remediate OS vulnerabilities	Recommends that you align your OS configurations with the recommended configuration rules, e.g. do not allow passwords to be saved.
Apply system updates	Recommends that you deploy missing system security and critical updates to VMs.
Reboot after system updates	Recommends that you reboot a VM to complete the process of applying system updates.
Install Endpoint Protection	Recommends that you provision antimalware programs to VMs (Windows VMs only).

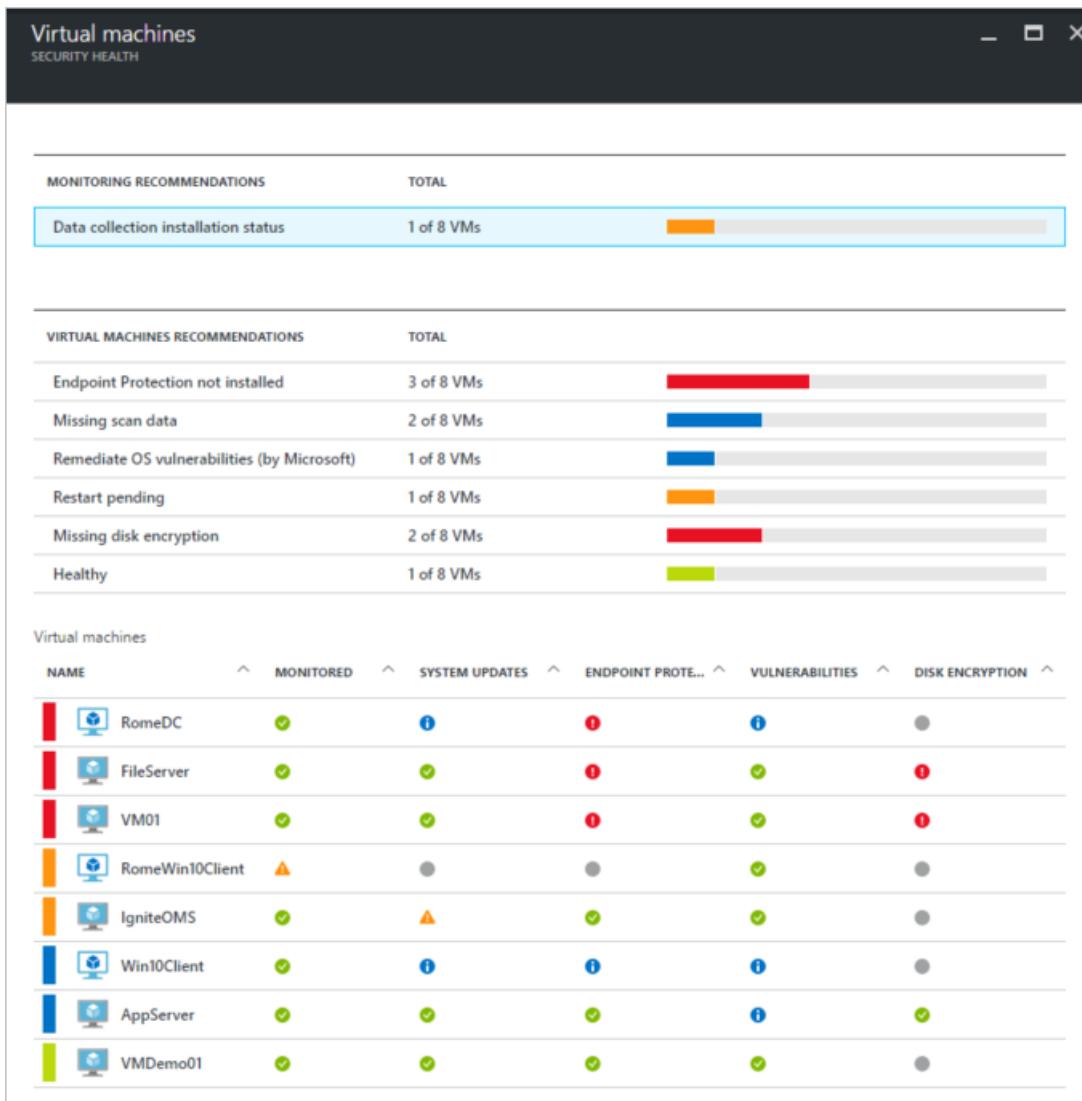
RECOMMENDATION	DESCRIPTION
Resolve Endpoint Protection health alerts	Recommends that you resolve endpoint protection failures.
Enable VM Agent	Enables you to see which VMs require the VM Agent. The VM Agent must be installed on VMs in order to provision patch scanning, baseline scanning, and antimalware programs. The VM Agent is installed by default for VMs that are deployed from the Azure Marketplace. The article VM Agent and Extensions – Part 2 provides information on how to install the VM Agent.
Apply disk encryption	Recommends that you encrypt your VM disks using Azure Disk Encryption (Windows and Linux VMs). Encryption is recommended for both the OS and data volumes on your VM.
Vulnerability assessment not installed	Recommends that you install a vulnerability assessment solution on your VM.
Remediate vulnerabilities	Enables you to see system and application vulnerabilities detected by the vulnerability assessment solution installed on your VM.

NOTE

To learn more about recommendations, see [Managing security recommendations](#) article.

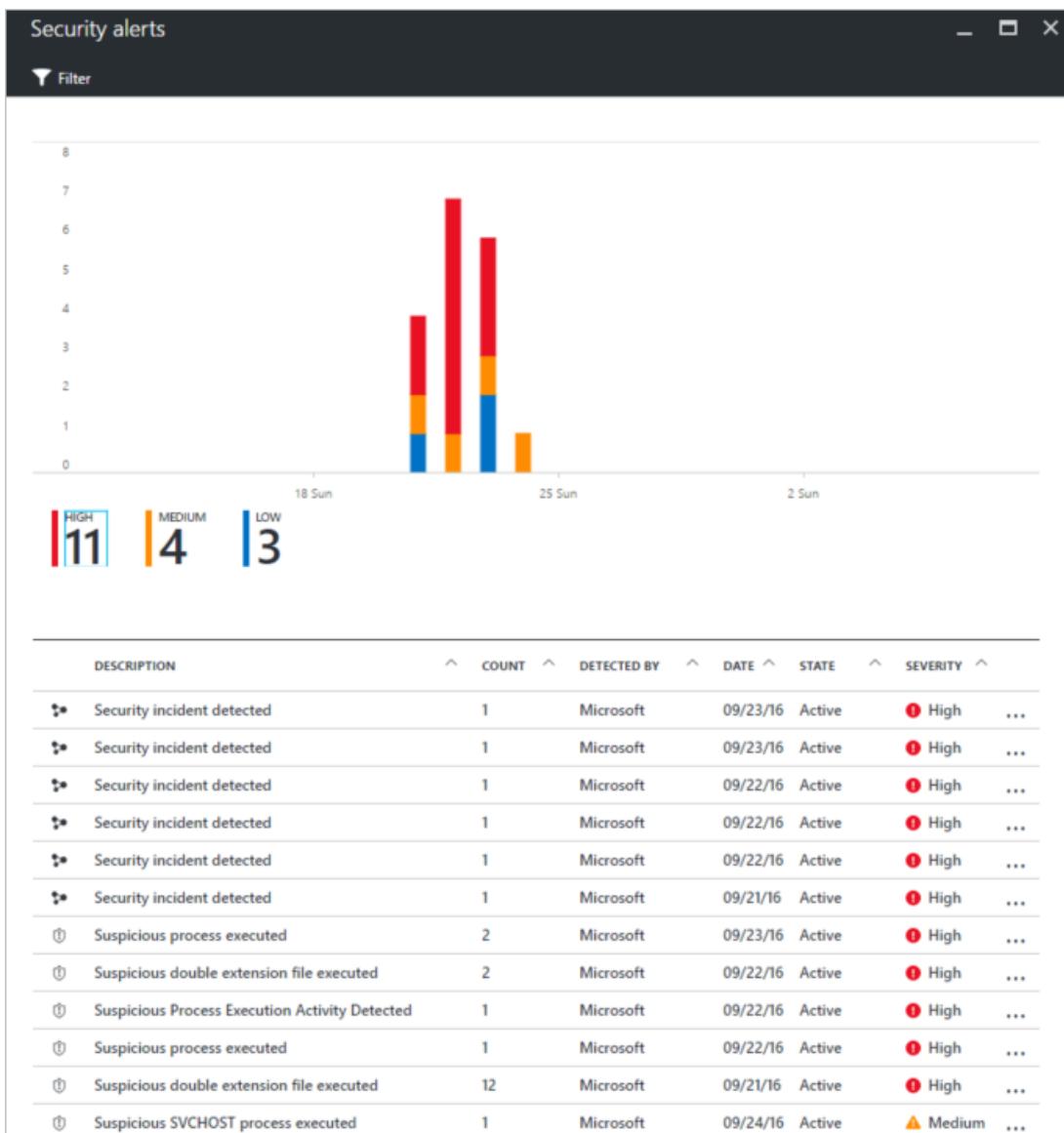
Monitor security health

After you enable [security policies](#) for a subscription's resources, Security Center will analyze the security of your resources to identify potential vulnerabilities. You can view the security state of your resources, along with any issues in the **Resource security health** blade. When you click **Virtual machines** in the **Resource security** health tile, the **Virtual machines** blade will open with recommendations for your VMs.



Manage and respond to security alerts

Security Center automatically collects, analyzes, and integrates log data from your Azure resources, the network, and connected partner solutions (like firewall and endpoint protection solutions), to detect real threats and reduce false positives. By leveraging a diverse aggregation of [detection capabilities](#), Security Center is able to generate prioritized security alerts to help you quickly investigate the problem and provide recommendations for how to remediate possible attacks.



Select a security alert to learn more about the event(s) that triggered the alert and what, if any, steps you need to take to remediate an attack. Security alerts are grouped by [type](#) and date.

See also

To learn more about Security Center, see the following:

- [Setting security policies in Azure Security Center](#) -- Learn how to configure security policies for your Azure subscriptions and resource groups.
- [Managing and responding to security alerts in Azure Security Center](#) -- Learn how to manage and respond to security alerts.
- [Azure Security Center FAQ](#) -- Find frequently asked questions about using the service.

Azure Resource Manager overview

1/17/2017 • 16 min to read • [Edit on GitHub](#)

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:

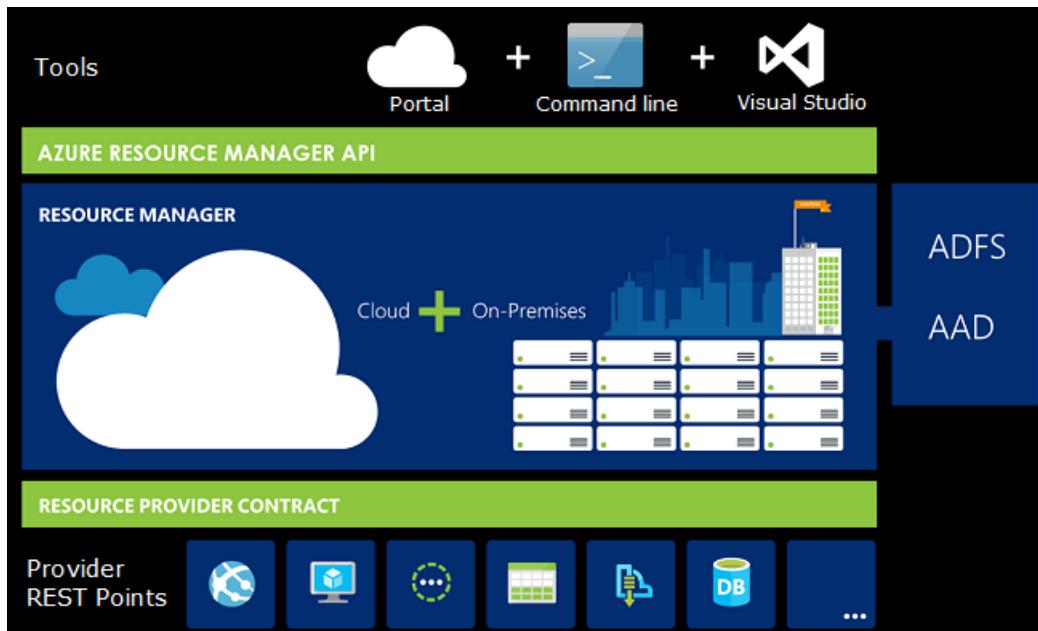
- You can deploy, manage, and monitor all the resources for your solution as a group, rather than handling these resources individually.
- You can repeatedly deploy your solution throughout the development lifecycle and have confidence your resources are deployed in a consistent state.
- You can manage 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, and a resource type called **vaults/secrets** for creating a secret in the key vault.

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.

You can see all resource providers through the portal. In the blade for your subscription, select **Resource providers**:

PROVIDER	STATUS	
Microsoft.ADHybridHealthService	Registered	Unregister
Microsoft.Authorization	Registered	Unregister
Microsoft.ClassicStorage	Registered	Unregister
Microsoft.Features	Registered	Unregister
Microsoft.OperationalInsights	Registered	Unregister
Microsoft.Resources	Registered	Unregister
Microsoft.Storage	Registered	Unregister
microsoft.support	Registered	Unregister
84codes.CloudAMQP	NotRegistered	Register
AppDynamics.APM	NotRegistered	Register

You retrieve all resource providers with the following PowerShell cmdlet:

```
Get-AzureRmResourceProvider -ListAvailable
```

Or, with Azure CLI, you retrieve all resource providers with the following command:

```
azure provider list
```

You can look through the returned list for the resource providers that you need to use.

To get details about a resource provider, add the provider namespace to your command. The command returns the supported resource types for the resource provider, and the supported locations and API versions for each resource type. The following PowerShell cmdlet gets details about Microsoft.Compute:

```
(Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Compute).ResourceTypes
```

Or, with Azure CLI, retrieve the supported resource types, locations, and API versions for Microsoft.Compute, with the following command:

```
azure provider show Microsoft.Compute --json > c:\Azure\compute.json
```

For more information, see [Resource Manager providers, regions, API versions, and schemas](#).

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 more about the format of the template and how you construct it, see [Authoring Azure Resource Manager Templates](#) and [Resource Manager Template Walkthrough](#).

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": {}
  }
]
```

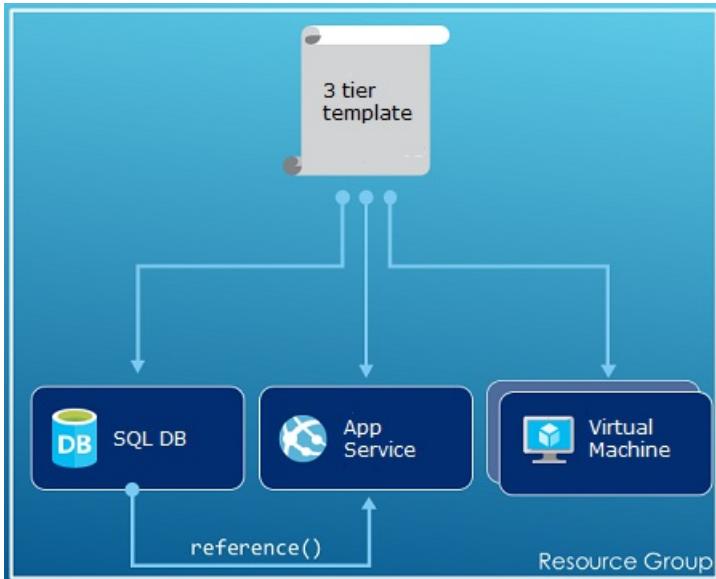
It converts the definition to the following REST API operation, which is sent to the Microsoft.Storage resource provider:

```

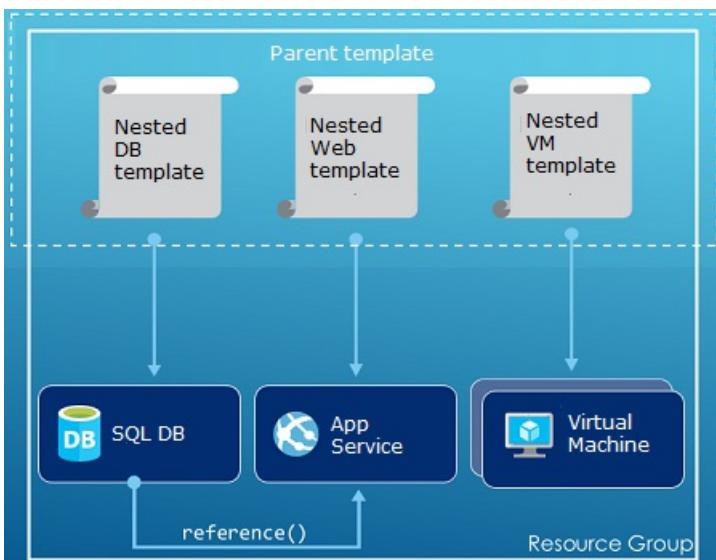
PUT
https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Storage/storageAccounts/mystorageaccount?api-version=2016-01-01
REQUEST BODY
{
  "location": "westus",
  "properties": {
  }
  "sku": {
    "name": "Standard_LRS"
  },
  "kind": "Storage"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

```

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.

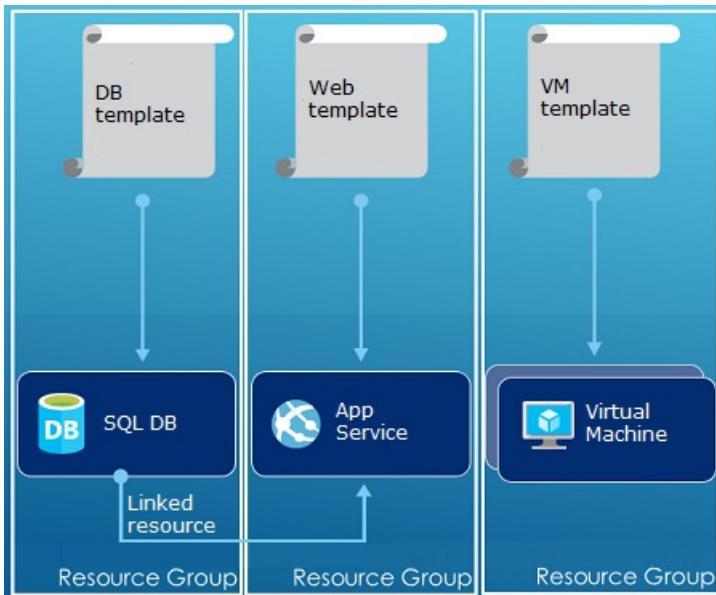


But, you do not have to define your entire infrastructure in a single template. Often, it makes sense to divide your deployment requirements into a set of targeted, purpose-specific templates. You can easily reuse these templates for different solutions. To deploy a particular solution, you create a 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](#).

For a four part series about automating deployment, see [Automating application deployments to Azure Virtual Machines](#). This series covers application architecture, access and security, availability and scale, and application deployment.

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 command:

```
azure resource list -t costCenter=Finance --json
```

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 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 [Use Policy to manage resources and control access](#).

SDKs

Azure SDKs are available for multiple languages and platforms. Each of these language implementations is available through its ecosystem package manager and GitHub.

The code in each of these SDKs is generated from Azure RESTful API specifications. These specifications are open source and based on the Swagger 2.0 specification. The SDK code is generated via an open-source project called AutoRest. AutoRest transforms these RESTful API specifications into client libraries in multiple languages. If you want to improve any aspects of the generated code in the SDKs, the entire set of tools to create the SDKs are open, freely available, and based on a widely adopted API specification format.

Here are our Open Source SDK repositories. We welcome feedback, issues, and pull requests.

[.NET](#) | [Java](#) | [Node.js](#) | [PHP](#) | [Python](#) | [Ruby](#)

NOTE

If the SDK doesn't provide the required functionality, you can also call to the [Azure REST API](#) directly.

Samples

.NET

- [Manage Azure resources and resource groups](#)
- [Deploy an SSH enabled VM with a template](#)

Java

- [Manage Azure resources](#)
- [Manage Azure resource groups](#)
- [Deploy an SSH enabled VM with a template](#)

Node.js

- [Manage Azure resources and resource groups](#)
- [Deploy an SSH enabled VM with a template](#)

Python

- [Manage Azure resources and resource groups](#)
- [Deploy an SSH enabled VM with a template](#)

Ruby

- [Manage Azure resources and resource groups](#)

- [Deploy an SSH enabled VM with a template](#)

In addition to these samples, you can search through the gallery samples.

[.NET](#) | [Java](#) | [Node.js](#) | [Python](#) | [Ruby](#)

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 [Resource Manager Template Walkthrough](#).
- 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](#).
- For information about using VS Code with Resource Manager, see [Working with Azure Resource Manager Templates in Visual Studio Code](#).

Here's a video demonstration of this overview:



Azure Resource Manager vs. classic deployment: Understand deployment models and the state of your resources

1/18/2017 • 12 min to read • [Edit on GitHub](#)

In this topic, you learn about Azure Resource Manager and classic deployment models, the state of your resources, and why your resources were deployed with one or the other. The Resource Manager and classic deployment models represent two different ways of deploying and managing your Azure solutions. You work with them through two different API sets, and the deployed resources can contain important differences. The two models are not completely compatible with each other. This topic describes those differences.

To simplify the deployment and management of resources, Microsoft recommends that you use Resource Manager for all new resources. If possible, Microsoft recommends that you redeploy existing resources through Resource Manager.

If you are new to Resource Manager, you may want to first review the terminology defined in the [Azure Resource Manager overview](#).

History of the deployment models

Azure originally provided only the classic deployment model. In this model, each resource existed independently; there was no way to group related resources together. Instead, you had to manually track which resources made up your solution or application, and remember to manage them in a coordinated approach. To deploy a solution, you had to either create each resource individually through the classic portal or create a script that deployed all the resources in the correct order. To delete a solution, you had to delete each resource individually. You could not easily apply and update access control policies for related resources. Finally, you could not apply tags to resources to label them with terms that help you monitor your resources and manage billing.

In 2014, Azure introduced Resource Manager, which added the concept of a resource group. A resource group is a container for resources that share a common lifecycle. The Resource Manager deployment model provides several benefits:

- You can deploy, manage, and monitor all the services for your solution as a group, rather than handling these services individually.
- You can repeatedly deploy your solution throughout its lifecycle and have confidence your resources are deployed in a consistent state.
- You can apply access control to all resources in your resource group, and those policies are automatically applied when new resources are added to the resource group.
- You can apply tags to resources to logically organize all the resources in your subscription.
- You can use JavaScript Object Notation (JSON) to define the infrastructure for your solution. The JSON file is known as a Resource Manager template.
- You can define the dependencies between resources so they are deployed in the correct order.

When Resource Manager was added, all resources were retroactively added to default resource groups. If you create a resource through classic deployment now, the resource is automatically created within a default resource group for that service, even though you did not specify that resource group at deployment. However, just existing within a resource group does not mean that the resource has been converted to the Resource Manager model. We'll look at how each service handles the two deployment models in the next section.

Understand support for the models

When deciding which deployment model to use for your resources, there are three scenarios to be aware of:

1. The service supports Resource Manager and provides only a single type.
2. The service supports Resource Manager but provides two types - one for Resource Manager and one for classic. This scenario applies only to virtual machines, storage accounts, and virtual networks.
3. The service does not support Resource Manager.

To discover whether a service supports Resource Manager, see [Resource Manager supported providers](#).

If the service you wish to use does not support Resource Manager, you must continue using classic deployment.

If the service supports Resource Manager and **is not** a virtual machine, storage account or virtual network, you can use Resource Manager without any complications.

For virtual machines, storage accounts, and virtual networks, if the resource was created through classic deployment, you must continue to operate on it through classic operations. If the virtual machine, storage account, or virtual network was created through Resource Manager deployment, you must continue using Resource Manager operations. This distinction can get confusing when your subscription contains a mix of resources created through Resource Manager and classic deployment. This combination of resources can create unexpected results because the resources do not support the same operations.

In some cases, a Resource Manager command can retrieve information about a resource created through classic deployment, or can perform an administrative task such as moving a classic resource to another resource group. But, these cases should not give the impression that the type supports Resource Manager operations. For example, suppose you have a resource group that contains a virtual machine that was created with classic deployment. If you run the following Resource Manager PowerShell command:

```
Get-AzureRmResource -ResourceGroupName ExampleGroup -ResourceType Microsoft.ClassicCompute/virtualMachines
```

It returns the virtual machine:

```
Name      : ExampleClassicVM
ResourceId   :
/subscriptions/{guid}/resourceGroups/ExampleGroup/providers/Microsoft.ClassicCompute/virtualMachines/ExampleClassicVM
ResourceName   : ExampleClassicVM
ResourceType    : Microsoft.ClassicCompute/virtualMachines
ResourceGroupName : ExampleGroup
Location      : westus
SubscriptionId  : {guid}
```

However, the Resource Manager cmdlet **Get-AzureRmVM** only returns virtual machines deployed through Resource Manager. The following command does not return the virtual machine created through classic deployment.

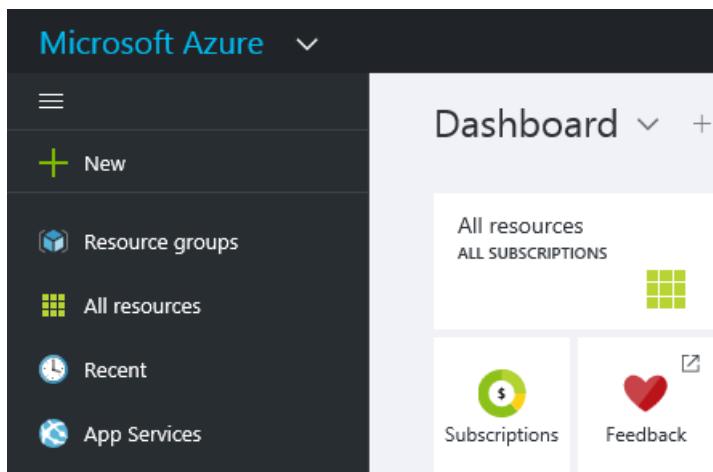
```
Get-AzureRmVM -ResourceGroupName ExampleGroup
```

Only resources created through Resource Manager support tags. You cannot apply tags to classic resources.

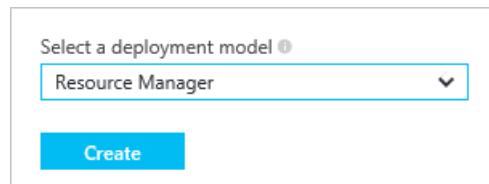
Resource Manager characteristics

To help you understand the two models, let's review the characteristics of Resource Manager types:

- Created through the [Azure portal](#).



For Compute, Storage, and Networking resources, you have the option of using either Resource Manager or Classic deployment. Select **Resource Manager**.



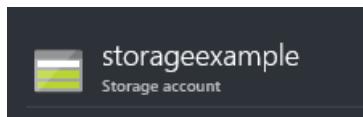
- Created with the Resource Manager version of the Azure PowerShell cmdlets. These commands have the format *Verb-AzureRmNoun*.

```
New-AzureRmResourceGroupDeployment
```

- Created through the [Azure Resource Manager REST API](#) for REST operations.
- Created through Azure CLI commands run in the **arm** mode.

```
azure config mode arm  
azure group deployment create
```

- The resource type does not include (**classic**) in the name. The following image shows the type as **Storage account**.



Classic deployment characteristics

You may also know the classic deployment model as the Service Management model.

Resources created in the classic deployment model share the following characteristics:

- Created through the [classic portal](#)

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the 'Microsoft Azure' logo and a dropdown menu. A blue banner on the right says 'Check out the new portal'. The left sidebar has a 'ALL ITEMS' link and three other categories: 'WEB APPS' (0), 'VIRTUAL MACHINES' (0), and 'MOBILE SERVICES' (0). The main content area is titled 'all items' and contains a table with one row. The table has a header row with 'NAME' and a data row with 'Tom FitzMacken'.

Or, the Azure portal and you specify **Classic** deployment (for Compute, Storage, and Networking).

A screenshot of a 'Create' dialog box. At the top, it says 'Select a deployment model' with a help icon. A dropdown menu is open, showing 'Classic' as the selected option. Below the dropdown is a large blue 'Create' button.

- Created through the Service Management version of the Azure PowerShell cmdlets. These command names have the format *Verb-AzureNoun*.

```
New-AzureVM
```

- Created through the [Service Management REST API](#) for REST operations.
- Created through Azure CLI commands run in **asm** mode.

```
azure config mode asm  
azure vm create
```

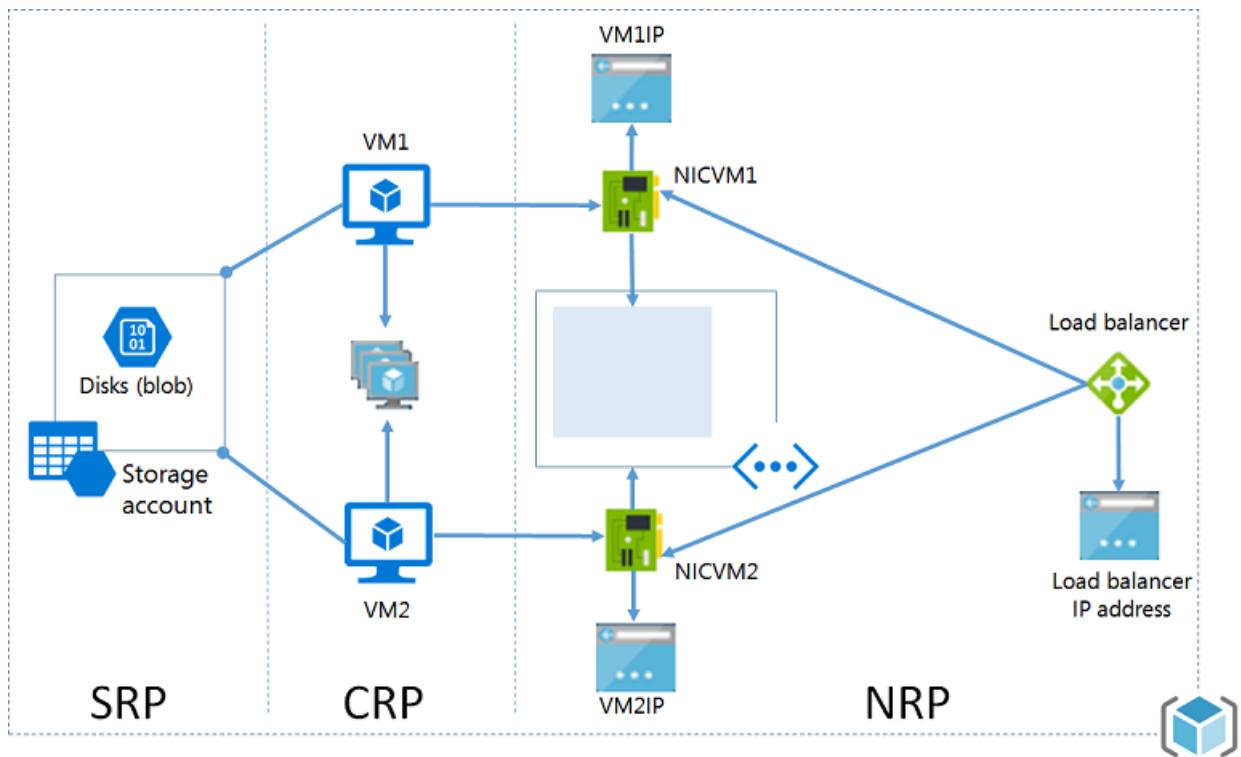
- The resource type includes **(classic)** in the name. The following image shows the type as **Storage account (classic)**.

A screenshot of a storage account card. At the top, it says 'ONLINE'. Below that is a blue square icon with white horizontal bars. To the right of the icon is the account name 'storageexample'. Underneath the name is the text 'Storage account (classic)'.

You can use the Azure portal to manage resources that were created through classic deployment.

Changes for compute, network, and storage

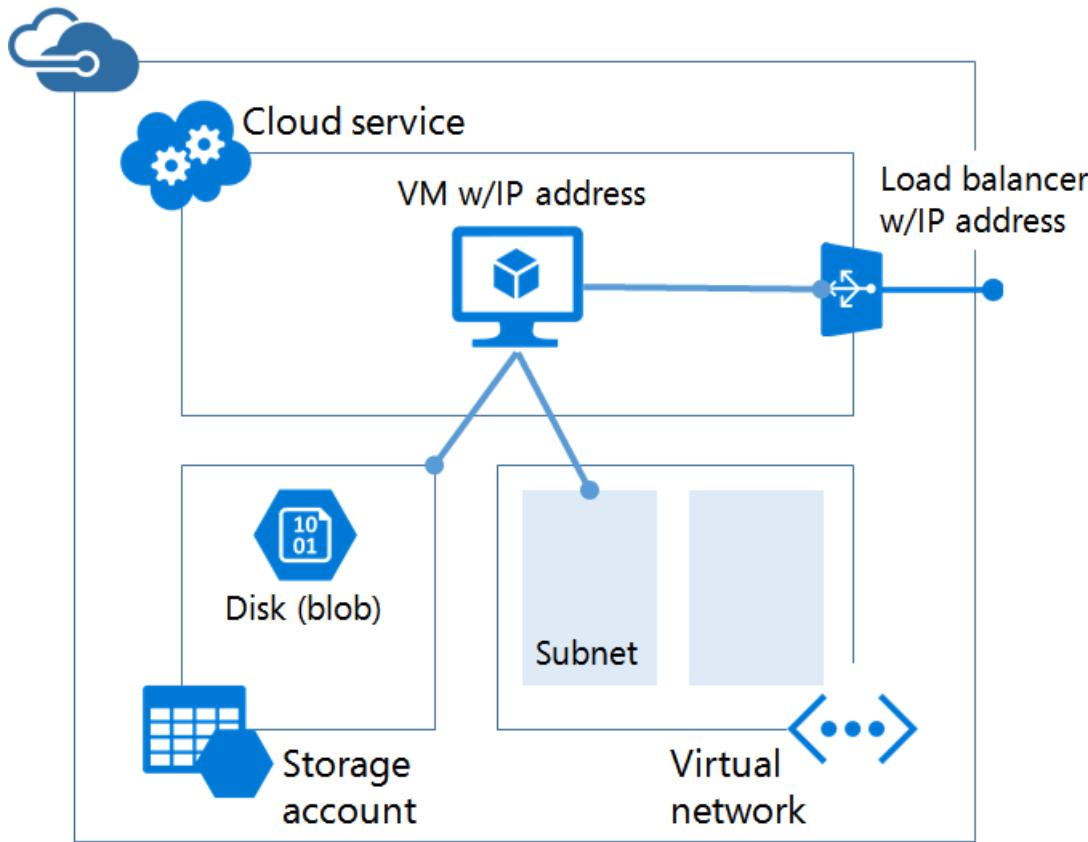
The following diagram displays compute, network, and storage resources deployed through Resource Manager.



Note the following relationships between the resources:

- All the resources exist within a resource group.
- The virtual machine depends on a specific storage account defined in the Storage resource provider to store its disks in blob storage (required).
- The virtual machine references a specific NIC defined in the Network resource provider (required) and an availability set defined in the Compute resource provider (optional).
- The NIC references the virtual machine's assigned IP address (required), the subnet of the virtual network for the virtual machine (required), and to a Network Security Group (optional).
- The subnet within a virtual network references a Network Security Group (optional).
- The load balancer instance references the backend pool of IP addresses that include the NIC of a virtual machine (optional) and references a load balancer public or private IP address (optional).

Here are the components and their relationships for classic deployment:



The classic solution for hosting a virtual machine includes:

- A required cloud service that acts as a container for hosting virtual machines (compute). Virtual machines are automatically provided with a network interface card (NIC) and an IP address assigned by Azure. Additionally, the cloud service contains an external load balancer instance, a public IP address, and default endpoints to allow remote desktop and remote PowerShell traffic for Windows-based virtual machines and Secure Shell (SSH) traffic for Linux-based virtual machines.
- A required storage account that stores the VHDs for a virtual machine, including the operating system, temporary, and additional data disks (storage).
- An optional virtual network that acts as an additional container, in which you can create a subnetted structure and designate the subnet on which the virtual machine is located (network).

The following table describes changes in how Compute, Network, and Storage resource providers interact:

ITEM	CLASSIC	RESOURCE MANAGER
Cloud Service for Virtual Machines	Cloud Service was a container for holding the virtual machines that required Availability from the platform and Load Balancing.	Cloud Service is no longer an object required for creating a Virtual Machine using the new model.
Virtual Networks	A virtual network is optional for the virtual machine. If included, the virtual network cannot be deployed with Resource Manager.	Virtual machine requires a virtual network that has been deployed with Resource Manager.
Storage Accounts	The virtual machine requires a storage account that stores the VHDs for the operating system, temporary, and additional data disks.	The virtual machine requires a storage account to store its disks in blob storage.

ITEM	CLASSIC	RESOURCE MANAGER
Availability Sets	Availability to the platform was indicated by configuring the same "AvailabilitySetName" on the Virtual Machines. The maximum count of fault domains was 2.	Availability Set is a resource exposed by Microsoft.Compute Provider. Virtual Machines that require high availability must be included in the Availability Set. The maximum count of fault domains is now 3.
Affinity Groups	Affinity Groups were required for creating Virtual Networks. However, with the introduction of Regional Virtual Networks, that was not required anymore.	To simplify, the Affinity Groups concept doesn't exist in the APIs exposed through Azure Resource Manager.
Load Balancing	Creation of a Cloud Service provides an implicit load balancer for the Virtual Machines deployed.	The Load Balancer is a resource exposed by the Microsoft.Network provider. The primary network interface of the Virtual Machines that needs to be load balanced should be referencing the load balancer. Load Balancers can be internal or external. A load balancer instance references the backend pool of IP addresses that include the NIC of a virtual machine (optional) and references a load balancer public or private IP address (optional). Read more .
Virtual IP Address	Cloud Services get a default VIP (Virtual IP Address) when a VM is added to a cloud service. The Virtual IP Address is the address associated with the implicit load balancer.	Public IP address is a resource exposed by the Microsoft.Network provider. Public IP Address can be Static (Reserved) or Dynamic. Dynamic Public IPs can be assigned to a Load Balancer. Public IPs can be secured using Security Groups.
Reserved IP Address	You can reserve an IP Address in Azure and associate it with a Cloud Service to ensure that the IP Address is sticky.	Public IP Address can be created in "Static" mode and it offers the same capability as a "Reserved IP Address". Static Public IPs can only be assigned to a Load balancer right now.
Public IP Address (PIP) per VM	Public IP Addresses can also be associated to a VM directly.	Public IP address is a resource exposed by the Microsoft.Network provider. Public IP Address can be Static (Reserved) or Dynamic. However, only dynamic Public IPs can be assigned to a Network Interface to get a Public IP per VM right now.
Endpoints	Input Endpoints needed to be configured on a Virtual Machine to be open up connectivity for certain ports. One of the common modes of connecting to virtual machines done by setting up input endpoints.	Inbound NAT Rules can be configured on Load Balancers to achieve the same capability of enabling endpoints on specific ports for connecting to the VMs.

ITEM	CLASSIC	RESOURCE MANAGER
DNS Name	A cloud service would get an implicit globally unique DNS Name. For example: <code>mycoffeeshop.cloudapp.net</code>	DNS Names are optional parameters that can be specified on a Public IP Address resource. The FQDN is in the following format - <code><domainlabel>. <region>.cloudapp.azure.com</code>
Network Interfaces	Primary and Secondary Network Interface and its properties were defined as network configuration of a Virtual machine.	Network Interface is a resource exposed by Microsoft.Network Provider. The lifecycle of the Network Interface is not tied to a Virtual Machine. It references the virtual machine's assigned IP address (required), the subnet of the virtual network for the virtual machine (required), and to a Network Security Group (optional).

To learn about connecting virtual networks from different deployment models, see [Connect virtual networks from different deployment models in the portal](#).

Migrate from classic to Resource Manager

If you are ready to migrate your resources from classic deployment to Resource Manager deployment, see:

1. [Technical deep dive on platform-supported migration from classic to Azure Resource Manager](#)
2. [Platform supported migration of IaaS resources from Classic to Azure Resource Manager](#)
3. [Migrate IaaS resources from classic to Azure Resource Manager by using Azure PowerShell](#)
4. [Migrate IaaS resources from classic to Azure Resource Manager by using Azure CLI](#)

Frequently Asked Questions

Can I create a virtual machine using Azure Resource Manager to deploy in a virtual network created using classic deployment?

This is not supported. You cannot use Azure Resource Manager to deploy a virtual machine into a virtual network that was created using classic deployment.

Can I create a virtual machine using the Azure Resource Manager from a user image that was created using the Azure Service Management APIs?

This is not supported. However, you can copy the VHD files from a storage account that was created using the Service Management APIs, and add them to a new account created through Azure Resource Manager.

What is the impact on the quota for my subscription?

The quotas for the virtual machines, virtual networks, and storage accounts created through the Azure Resource Manager are separate from other quotas. Each subscription gets quotas to create the resources using the new APIs. You can read more about the additional quotas [here](#).

Can I continue to use my automated scripts for provisioning virtual machines, virtual networks, and storage accounts through the Resource Manager APIs?

All the automation and scripts that you've built continue to work for the existing virtual machines, virtual networks created under the Azure Service Management mode. However, the scripts have to be updated to use

the new schema for creating the same resources through the Resource Manager mode.

Where can I find examples of Azure Resource Manager templates?

A comprehensive set of starter templates can be found on [Azure Resource Manager QuickStart Templates](#).

Next steps

- To walk through the creation of template that defines a virtual machine, storage account, and virtual network, see [Resource Manager template walkthrough](#).
- To see the commands for deploying a template, see [Deploy an application with Azure Resource Manager template](#).

Virtual Machine Scale Sets Overview

1/17/2017 • 9 min to read • [Edit on GitHub](#)

Virtual machine scale sets are an Azure Compute resource you can use to deploy and manage a set of identical VMs. With all VMs configured the same, VM scale sets are designed to support true autoscale – no pre-provisioning of VMs is required – and as such makes it easier to build large-scale services targeting big compute, big 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 an introduction to VM scale sets refer to the [Azure blog announcement](#).

Take a look at these videos for more about VM scale sets:

- [Mark Russinovich talks Azure Scale Sets](#)
- [Virtual Machine Scale Sets with Guy Bowerman](#)

Creating and managing VM scale sets

You can create a VM Scale Set in the [Azure portal](#) by selecting *new* and typing in "scale" in the search bar. You will see "Virtual machine scale set" in the results. From there you can fill in the required fields to customize and deploy your scale set.

VM scale sets can also be defined and deployed using JSON templates and [REST APIs](#) just like individual Azure Resource Manager VMs. Therefore, any standard Azure Resource Manager deployment methods can be used. For more information about templates, see [Authoring Azure Resource Manager templates](#).

A set of example templates for VM scale sets can be found in the Azure Quickstart templates GitHub repository [here](#). (look for templates with `vmss` in the title)

In the detail pages for these templates you'll see a button that links to the portal deployment feature. To deploy the VM scale set, click on the button and then fill in any parameters that are required in the portal. If you're not sure whether a resource supports upper or mixed case it is safer to always use lower case parameter values. There is also a handy video dissection of a VM scale set template [here](#):

[VM Scale Set Template Dissection](#)

Scaling a VM scale set out and in

To increase or decrease the number of virtual machines in a VM scale set, simply change the *capacity* property and redeploy the template. This simplicity makes it easy to write your own custom scaling layer if you want to define custom scale events that are not supported by Azure autoscale.

If you are redeploying a template to change the capacity, you could define a much smaller template which only includes the SKU and the updated capacity. An example of this is shown [here](#).

To walk through the steps that create a scale set that is automatically scaled, see [Automatically Scale Machines in a Virtual Machine Scale Set](#)

Monitoring your VM scale set

The [Azure portal](#) lists scale sets and shows basic properties and operations, including listing VMs in the set and a resource usage graph. For more detail you can use the [Azure Resource Explorer](#) to view VM scale sets. VM scale sets are a resource under Microsoft.Compute, so from this site you can see them by expanding the following links:

Subscriptions -> your subscription -> resourceGroups -> providers -> Microsoft.Compute -> virtualMachineScaleSets -> your VM scale set -> etc.

VM scale set scenarios

This section lists some typical VM scale set scenarios. Some higher level Azure services (like Batch, Service Fabric, Azure Container Service) will use these scenarios.

- **RDP / SSH to VM scale set instances** - A VM scale set is created inside a VNET and individual VMs in the scale set are not allocated public IP addresses. This is a good thing because you don't generally want the expense and management overhead of allocating separate public IP addresses to all the stateless resources in your compute grid, and you can easily connect to these VMs from other resources in your VNET including ones which have public IP addresses like load balancers or standalone virtual machines.
- **Connect to VMs using NAT rules** - You can create a public IP address, assign it to a load balancer, and define inbound NAT rules which map a port on the IP address to a port on a VM in the VM 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

Here's an example of creating a VM scale set which uses NAT rules to enable SSH connection to every VM in a scale set using a single public IP: <https://github.com/Azure/azure-quickstart-templates/tree/master/201-vmss-linux-nat>

Here's an example of doing the same with RDP and Windows: <https://github.com/Azure/azure-quickstart-templates/tree/master/201-vmss-windows-nat>

- **Connect to VMs using a "jumpbox"** - If you create a VM scale set and a standalone VM in the same VNET, the standalone VM and the VM scale set VMs can connect to one another using their internal IP addresses as defined by the VNET/Subnet. If you create a public IP address and assign it to the standalone VM you can RDP or SSH to the standalone VM and then connect from that machine to your VM scale set instances. You may notice at this point that a simple VM 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: <https://github.com/Azure/azure-quickstart-templates/tree/master/201-vmss-linux-jumpbox>

- **Load balancing to VM scale set instances** - If you want to deliver work to a compute cluster of VMs using a "round-robin" approach, you can configure an Azure load balancer with load-balancing rules accordingly. You can define probes to verify your application is running by pinging ports with a specified protocol, interval and request path. The Azure [Application Gateway](#) also supports scale sets, along with more sophisticated load balancing scenarios.

Here is an example which creates a VM Scale Set running Apache web servers, and uses a load balancer to balance the load that each VM receives: <https://github.com/Azure/azure-quickstart-templates/tree/master/201-vmss-ubuntu-web-ssl> (look at the Microsoft.Network/loadBalancers resource type and the networkProfile and extensionProfile in the virtualMachineScaleSet)

- **Deploying a VM scale set as a compute cluster in a PaaS cluster manager** - VM scale sets are sometimes described as a next-generation worker role. It's a valid description but does run the risk of

confusing scale set features with PaaS v1 Worker role features. In a sense VM scale sets provide a true "worker role" or worker resource, in that they provide a generalized compute resource which is platform/runtime independent, customizable and integrates into Azure Resource Manager IaaS.

A PaaS v1 worker role, while limited in terms of platform/runtime support (Windows platform images only) also includes services such as VIP swap, configurable upgrade settings, runtime/app deployment specific settings which are either not yet available in VM scale sets, or will be delivered by other higher level PaaS services like Service Fabric. With this in mind you can look at VM scale sets as an infrastructure which supports PaaS. I.e. PaaS solutions like Service Fabric or cluster managers like Mesos can build on top of VM scale sets as a scalable compute layer.

For an example of this approach, the Azure Container Service deploys a cluster based on scale sets with a container orchestrator: <https://github.com/Azure/azure-quickstart-templates/tree/master/101-acs-dcos>.

VM scale set performance and scale guidance

- Do not create more than 500 VMs in multiple VM Scale Sets at a time.
- Plan for no more than 20 VMs per storage account (unless you set the *overprovision* property to "false", in which case you can go up to 40).
- Spread out the first letters of storage account names as much as possible. The example VMSS templates in [Azure Quickstart templates](#) provide examples of how to do this.
- If using custom VMs, plan for no more than 40 VMs per VM scale set, in a single storage account. You will need the image pre-copied into the storage account before you can begin VM scale set deployment. See the FAQ for more information.
- Plan for no more than 4096 VMs per VNET.
- The number of VMs you can create is limited by the core quota in the region in which you are deploying. You may need to contact Customer Support to increase your Compute quota limit increased even if you have a high limit of cores for use with cloud services or IaaS v1 today. To query your quota you can run the following Azure CLI command: `azure vm list-usage`, and the following PowerShell command: `Get-AzureRmVMUsage` (if using a version of PowerShell below 1.0 use `Get-AzureVMUsage`).

VM scale set frequently asked questions

Q. How many VMs can you have in a VM scale set?

A. 100 if you use platform images which can be distributed across multiple storage accounts. If you use custom images, up to 40 (if the *overprovision* property is set to "false", 20 by default), since custom images are currently limited to a single storage account.

Q What other resource limits exist for VM scale sets?

A. You are limited to creating no more than 500 VMs in multiple scale sets per region during a 10 minute period. The existing [Azure Subscription Service Limits](#) apply.

Q. Are Data Disks Supported within VM scale sets?

A. Not in the initial release (though data disks are currently available in preview). Your options for storing data are:

- Azure files (SMB shared drives)
- OS drive
- Temp drive (local, not backed by Azure storage)
- Azure data service (e.g. Azure tables, Azure blobs)
- External data service (e.g. remote DB)

Q. Which Azure regions support VM scale sets?

A. Any region which supports Azure Resource Manager supports VM Scale Sets.

Q. How do you create a VM scale set using a custom image?

A. Leave the vhdContainers property blank, for example:

```
"storageProfile": {  
    "osDisk": {  
        "name": "vmssosdisk",  
        "caching": "ReadOnly",  
        "createOption": "FromImage",  
        "image": {  
            "uri":  
                "https://mycustomimage.blob.core.windows.net/system/Microsoft.Compute/Images/mytemplates/template-osDisk.vhd"  
            },  
            "osType": "Windows"  
        }  
    },  
},
```

Q. If I reduce my VM scale set capacity from 20 to 15, which VMs will be removed?

A. Virtual machines are removed from the scale set evenly across upgrade domains and fault domains to maximize availability. VMs with the highest id's are removed first.

Q. How about it if I then increase the capacity from 15 to 18?

A. If you increase capacity to 18, then 3 new VMs will be created. Each time the VM instance id will be incremented from the previous highest value (e.g. 20, 21, 22). VMs are balanced across FDs and UDs.

Q. When using multiple extensions in a VM scale set, can I enforce an execution sequence?

A. Not directly, but for the customScript extension, your script could wait for another extension to complete ([for example by monitoring the extension log](#)). Additional guidance on extension sequencing can be found in this blog post: [Extension Sequencing in Azure VM Scale Sets](#).

Q. Do VM scale sets work with Azure availability sets?

A. Yes. A VM scale set is an implicit availability set with 5 FDs and 5 UDs. You don't need to configure anything under virtualMachineProfile. In future releases, VM scale sets are likely to span multiple tenants but for now a scale set is a single availability set.

Virtual machines and containers in Azure

1/17/2017 • 12 min to read • [Edit on GitHub](#)

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 offers you great cloud solutions, built on virtual machines—based on the emulation of physical computer hardware—to enable agile movement of software deployments and dramatically better resource consolidation than physical hardware. In the past few years, largely thanks to the [Docker](#) approach to containers and the docker ecosystem, Linux container technology has dramatically expanded the ways you can develop and manage distributed software. Application code in a container is isolated from the host Azure VM as well as other containers on the same VM, which gives you more development and deployment agility at the application level—in addition to the agility that Azure VMs already give you.

But that's old news. The *new* news is that Azure offers you even more Docker goodness:

- Many different ways to create Docker hosts for containers to suit your situation
- The [Azure Container Service](#) creates clusters of container hosts using orchestrators such as **marathon** and **swarm**.
- [Azure Resource Manager](#) and [resource group templates](#) to simplify deploying and updating complex distributed applications
- integration with a large array of both proprietary and open-source configuration management tools

And because you can programmatically create VMs and Linux containers on Azure, you can also use VM and container *orchestration* tools to create groups of Virtual Machines (VMs) and to deploy applications inside both Linux containers and now [Windows Containers](#).

This article not only discusses these concepts at a high level, it also contains tons of links to more information, tutorials, and products related to container and cluster usage on Azure. If you know all this, and just want the links, they're right here at [tools for working with containers](#).

The difference between virtual machines and containers

Virtual machines run inside an isolated hardware virtualization environment provided by a [hypervisor](#). In Azure, the [Virtual Machines](#) service handles all that for you: You just create Virtual Machines by choosing the operating system and configuring it to run the way you want—or by uploading your own custom VM image. Virtual Machines are a time-tested, "battle-hardened" technology, and there are many tools available to manage operating systems and to configure the applications you install and run. Anything running in a virtual machine is hidden from the host operating system and, from the point of view of an application or user running inside a virtual machine, the virtual machine appears to be an autonomous physical computer.

[Linux containers](#)—which includes those created and hosted using docker tools, and there are other approaches—do not require or use a hypervisor to provide isolation. Instead, the container host uses the process and file system isolation features of the Linux kernel to expose to the container (and its application) only certain kernel features and its own isolated file system (at a minimum). From the point of view of an application running inside a container, the container appears to be a unique operating system instance. A contained application cannot see processes or any other resources outside of its container.

Because in this isolation and execution model the kernel of the Docker host computer is shared, and because the disk requirements of the container now do not include an entire operating system, both the start-up time of the container and the required disk storage overhead are much, much smaller.

It's pretty cool.

Windows Containers provide the same advantages as Linux containers for applications that run on Windows. Windows Containers support the Docker image format and Docker API, but they can also be managed using PowerShell. Two container runtimes are available with Windows Containers, Windows Server Containers and Hyper-V Containers. Hyper-V Containers provide an additional layer of isolation by hosting each container in a super-optimized virtual machine. To learn more about Windows Containers see [About Windows Containers](#). To get started with Windows Containers in Azure, learn how to [deploy an Azure Container Service cluster](#).

That's pretty cool, too.

Is this too good to be true?

Well, yes—and no. Containers, like any other technology, do not magically wipe away all the hard work required by distributed applications. Yet, at the same time containers do really change:

- how fast application code can be developed and shared widely
- how fast and with what confidence it can be tested
- how fast and with what confidence it can be deployed

That said, remember containers execute on a container host—an operating system, and in Azure that means an Azure Virtual Machine. Even if you already love the idea of containers, you're still going to need a VM infrastructure hosting the containers, but the benefits are that containers do not care on which VM they are running (although whether the container wants a Linux or Windows execution environment will be important, for example).

What are containers good for?

They're great for many things, but they encourage—as do [Azure Cloud Services](#) and [Azure Service Fabric](#)—the creation of single-service, microservice-oriented distributed applications, in which application design is based on more small, composable parts rather than on larger, more strongly coupled components.

This is especially true in public cloud environments like Azure, in which you rent VMs when and where you want them. Not only do you get isolation and rapid deployment and orchestration tools, but you can make more efficient application infrastructure decisions.

For example, you might currently have a deployment consisting of 9 Azure VMs of a large size for a highly-available, distributed application. If the components of this application can be deployed in containers, you might be able to use only 4 VMs and deploy your application components inside 20 containers for redundancy and load balancing.

This is just an example, of course, but if you can do this in your scenario, you can adjust to usage spikes with more containers rather than more Azure VMs, and use the remaining overall CPU load much more efficiently than before.

In addition, there are many scenarios that do not lend themselves to a microservices approach; you will know best whether microservices and containers will help you.

Container benefits for developers

In general, it's easy to see that container technology is a step forward, but there are more specific benefits as well. Let's take the example of Docker containers. This topic will not dive deeply into Docker right now (read [What is Docker?](#) for that story, or [wikipedia](#)), but Docker and its ecosystem offer tremendous benefits to both developers and IT professionals.

Developers take to Docker containers quickly, because above all it makes using Linux and Windows containers easy:

- They can use simple, incremental commands to create a fixed image that is easy to deploy and can automate building those images using a dockerfile
- They can share those images easily using simple, [git](#)-style push and pull commands to [public](#) or [private docker registries](#)
- They can think of isolated application components instead of computers
- They can use a large number of tools that understand docker containers and different base images

Container benefits for operations and IT professionals

IT and operations professionals also benefit from the combination of containers and virtual machines.

- contained services are isolated from VM host execution environment
- contained code is verifiably identical
- contained services can be started, stopped, and moved quickly between development, test, and production environments

Features like these—and there are more—excite established businesses, where professional information technology organizations have the job of fitting resources—including pure processing power—to the tasks required to not only stay in business, but increase customer satisfaction and reach. Small businesses, ISVs, and startups have exactly the same requirement, but they might describe it differently.

What are virtual machines good for?

Virtual machines provide the backbone of cloud computing, and that doesn't change. If virtual machines start more slowly, have a larger disk footprint, and do not map directly to a microservices architecture, they do have very important benefits:

1. By default, they have much more robust default security protections for host computer
2. They support any major OS and application configurations
3. They have longstanding tool ecosystems for command and control
4. They provide the execution environment to host containers

The last item is important, because a contained application still requires a specific operating system and CPU type, depending upon the calls the application will make. It's important to remember that you install containers on VMs because they contain the applications you want to deploy; containers are not replacements for VMs or operating systems.

High-level feature comparison of VMs and containers

The following table describes at a very high level the kind of feature differences that—with much extra work—exist between VMs and Linux containers. Note that some features maybe more or less desirable depending upon your own application needs, and that as with all software, extra work provides increased feature support, especially in the area of security.

FEATURE	VMS	CONTAINERS
"Default" security support	to a greater degree	to a slightly lesser degree
Memory on disk required	Complete OS plus apps	App requirements only
Time taken to start up	Substantially Longer: Boot of OS plus app loading	Substantially shorter: Only apps need to start because kernel is already running

FEATURE	VMS	CONTAINERS
Portability	Portable With Proper Preparation	Portable within image format; typically smaller
Image Automation	Varies widely depending on OS and apps	Docker registry ; others

Creating and managing groups of VMs and containers

At this point, any architect, developer, or IT operations specialist might be thinking, "I can automate ALL of this; this really IS Data-Center-As-A-Service!".

You're right, it can be, and there are any number of systems, many of which you may already use, that can either manage groups of Azure VMs and inject custom code using scripts, often with the [CustomScriptingExtension for Windows](#) or the [CustomScriptingExtension for Linux](#). You can—and perhaps already have—automated your Azure deployments using PowerShell or Azure CLI scripts.

These abilities are often then migrated to tools like [Puppet](#) and [Chef](#) to automate the creation of and configuration for VMs at scale. (Here are some links to [using these tools with Azure](#).)

Azure resource group templates

More recently, Azure released the [Azure resource management](#) REST API, and updated PowerShell and Azure CLI tools to use it easily. You can deploy, modify, or redeploy entire application topologies using [Azure Resource Manager templates](#) with the Azure resource management API using:

- the [Azure portal using templates](#)—hint, use the "DeployToAzure" button
- the [Azure CLI](#)
- the [Azure PowerShell modules](#)

Deployment and management of entire groups of Azure VMs and containers

There are several popular systems that can deploy entire groups of VMs and install Docker (or other Linux container host systems) on them as an automatable group. For direct links, see the [containers and tools](#) section, below. There are several systems that do this to a greater or lesser extent, and this list is not exhaustive. Depending upon your skill set and scenarios, they may or may not be useful.

Docker has its own set of VM-creation tools ([docker-machine](#)) and a load-balancing, docker-container cluster management tool ([swarm](#)). In addition, the [Azure Docker VM Extension](#) comes with default support for [docker-compose](#), which can deploy configured application containers across multiple containers.

In addition, you can try out [Mesosphere's Data Center Operating System \(DCOS\)](#). DCOS is based on the open-source [mesos](#) "distributed systems kernel" that enables you to treat your datacenter as one addressable service. DCOS has built-in packages for several important systems such as [Spark](#) and [Kafka](#) (and others) as well as built-in services such as [Marathon](#) (a container control system) and [Chronos](#) (a distributed scheduler). Mesos was derived from lessons learned at Twitter, AirBnb, and other web-scale businesses. You can also use [swarm](#) as the orchestration engine.

Also, [kubernetes](#) is an open-source system for VM and container group management derived from lessons learned at Google. You can even use [kubernetes with weave to provide networking support](#).

[Deis](#) is an open source "Platform-as-a-Service" (PaaS) that makes it easy to deploy and manage applications on your own servers. Deis builds upon Docker and CoreOS to provide a lightweight PaaS with a Heroku-inspired workflow. You can easily [create a 3-Node Azure VM group and install Deis](#) on Azure and then [install a Hello World Go application](#).

CoreOS, a Linux distribution with an optimized footprint, Docker support, and their own container system called [rkt](#),

also has a container group management tool called [fleet](#).

Ubuntu, another very popular Linux distribution, supports Docker very well, but also supports [Linux \(LXC-style\) clusters](#).

Tools for working with Azure VMs and containers

Working with containers and Azure VMs uses tools. This section provides a list of only some of the most useful or important concepts and tools about containers, groups, and the larger configuration and orchestration tools used with them.

NOTE

This area is changing amazingly rapidly, and while we will do our best to keep this topic and its links up to date, it might well be an impossible task. Make sure you search on interesting subjects to keep up to date!

Containers and VM technologies

Some Linux container technologies:

- [Docker](#)
- [LXC](#)
- [CoreOS and rkt](#)
- [Open Container Project](#)
- [RancherOS](#)

Windows Container links:

- [Windows Containers](#)

Visual Studio Docker links:

- [Visual Studio 2015 RC Tools for Docker - Preview](#)

Docker tools:

- [Docker daemon](#)
- Docker clients
 - [Windows Docker Client on Chocolatey](#)
 - [Docker installation instructions](#)

Docker on Microsoft Azure:

- [Docker VM Extension for Linux on Azure](#)
- [Azure Docker VM Extension User Guide](#)
- [Using the Docker VM Extension from the Azure Command-line Interface \(Azure CLI\)](#)
- [Using the Docker VM Extension from the Azure portal](#)
- [How to use docker-machine on Azure](#)
- [How to use docker with swarm on Azure](#)
- [Get Started with Docker and Compose on Azure](#)
- [Using an Azure resource group template to create a Docker host on Azure quickly](#)
- [The built-in support for `compose` for contained applications](#)
- [Implement a Docker private registry on Azure](#)

Linux distributions and Azure examples:

- [CoreOS](#)

Configuration, cluster management, and container orchestration:

- [Fleet on CoreOS](#)
- [Deis](#)
 - [Create a 3-Node Azure VM group, install Deis, and start a Hello World Go application](#)
- [Kubernetes](#)
 - [Complete guide to automated Kubernetes cluster deployment with CoreOS and Weave](#)
 - [Kubernetes Visualizer](#)
- [Mesos](#)
 - [Mesosphere's Data Center Operating System \(DCOS\)](#)
- [Jenkins and Hudson](#)
 - [Blog: Jenkins Slave Plug-in for Azure](#)
 - [GitHub repo: Jenkins Storage Plug-in for Azure](#)
 - [Third Party: Hudson Slave Plug-in for Azure](#)
 - [Third Party: Hudson Storage Plug-in for Azure](#)
- [Azure Automation](#)
 - [Video: How to Use Azure Automation with Linux VMs](#)
- [Powershell DSC for Linux](#)
 - [Blog: How to do Powershell DSC for Linux](#)
 - [GitHub: Docker Client DSC](#)

Next steps

Check out [Docker](#) and [Windows Containers](#).

Frequently asked question about Windows Virtual Machines

1/17/2017 • 4 min to read • [Edit on GitHub](#)

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 and Windows 8.1 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 1 TB. The number of data disks you can use depends on the size of the virtual machine. For details, see [Sizes for Virtual Machines](#).

An Azure storage account provides storage for the operating system disk and any data disks. Each disk is a .vhd 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 [Upload a Windows VM image 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. For instructions, see [How to create a copy of a Windows virtual machine in the Resource Manager deployment model](#).

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?

No. Adding a NIC can only be done at creation time.

Are there any computer name requirements?

Yes. The computer name can be a maximum of 15 characters in length. See [Infrastructure naming guidelines](#) for more information around naming your resources.

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 8 - 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</td><td style="text-align:center">P@ssword123</td><td style="text-align:center">Pa\$\$word	
pass@word1</td><td style="text-align:center">Password!	Password1	Password22	iloveyou!

Create your first Windows virtual machine in the Azure portal

1/17/2017 • 3 min to read • [Edit on GitHub](#)

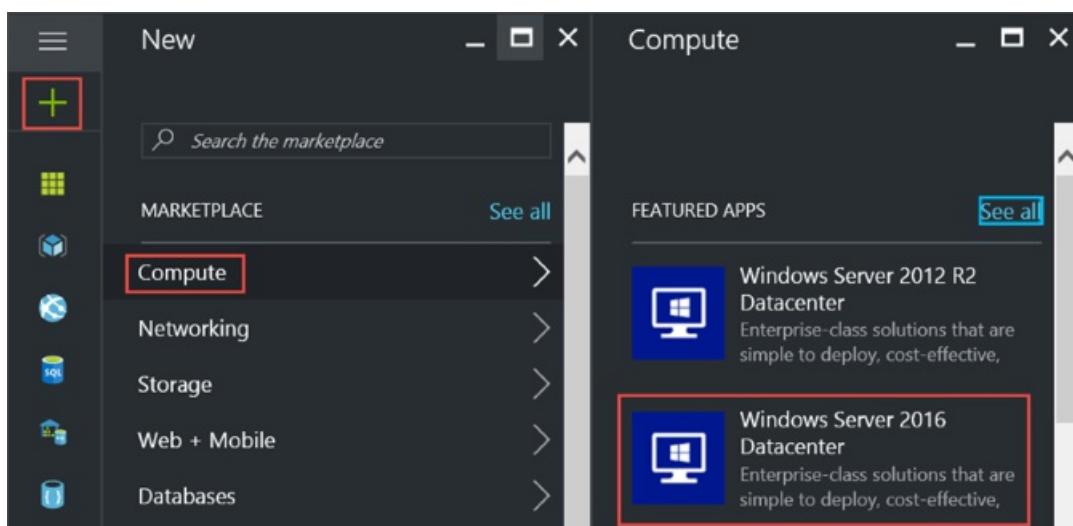
This tutorial shows you how easy it is to create a Windows virtual machine (VM) in just a few minutes, by using the Azure portal.

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

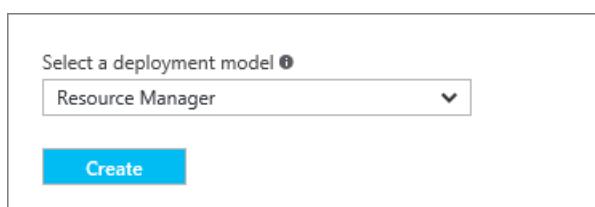
Choose the VM image from the marketplace

We use a Windows Server 2016 Datacenter image as an example, but that's just one of the many images Azure offers. Your image choices depend on your subscription. For example, some desktop images are available to [MSDN subscribers](#).

1. Sign in to the [Azure portal](#).
2. Starting in the upper-left, click **New > Compute > Windows Server 2016 Datacenter**.



3. On the **Windows Server 2016 Datacenter** blade, in **Select a deployment model**, verify that **Resource Manager** is selected. Click **Create**.



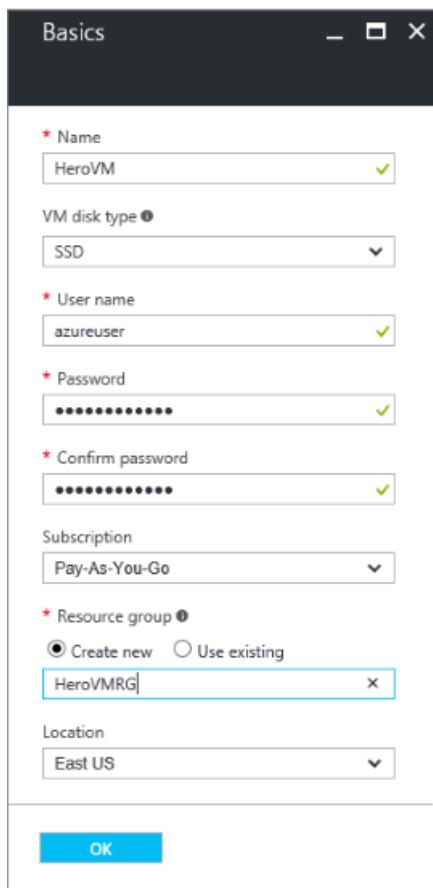
Create the Windows virtual machine

After you select the image, you can use the default settings and quickly create the virtual machine.

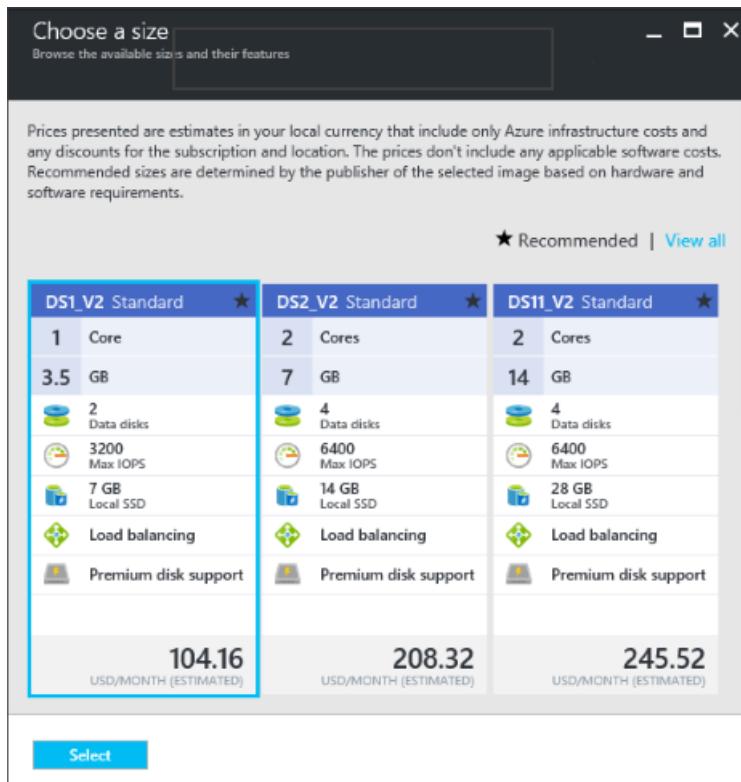
1. On the **Basics** blade, enter a **Name** for the virtual machine. In this example, *HeroVM* is the name of the virtual machine. The name must be 1-15 characters long and it cannot contain special characters.
2. Enter a **User name**, and a strong **Password** that will be used to create a local account on the VM. The local account is used to sign in to and manage the VM. In this example, *azureuser* is the user name.

The password must be 8-123 characters long and meet three out of the four following complexity requirements: one lower case character, one upper case character, one number, and one special character. See more about [username and password requirements](#).

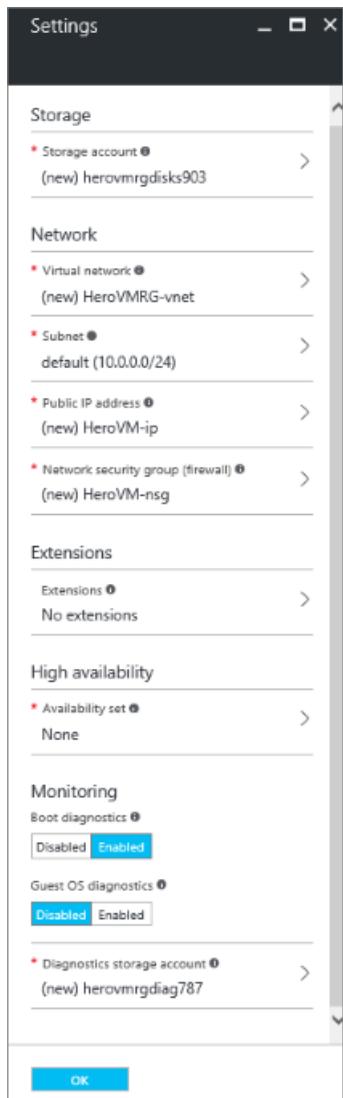
3. Select an existing [Resource group](#) or type the name for a new one. In this example, *HeroVMRG* is the name of the resource group.
4. Select an Azure datacenter **Location**. In this example, *East US** is the location.
5. When you are done, click **OK** to continue to the next section.



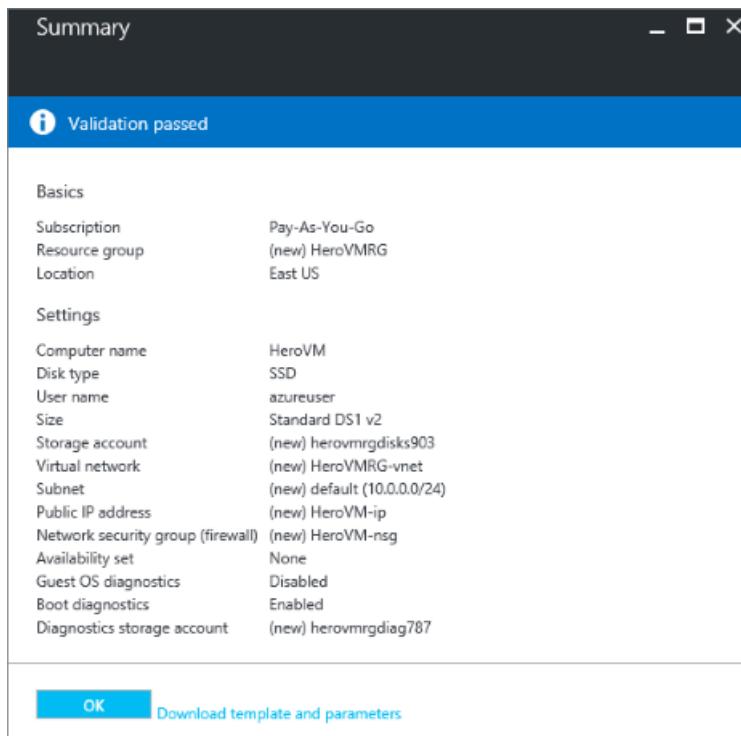
6. Choose a VM [size](#), and then click **Select** to continue. In this example, *DS1_V2 Standard* is the VM size.



- On the **Settings** blade, you can change the storage and network options. For this tutorial, accept the default settings. If you selected a virtual machine size that supports it, you can try Azure Premium Storage by selecting **Premium (SSD)** in **Disk type**. When you're done making changes, click **OK**.



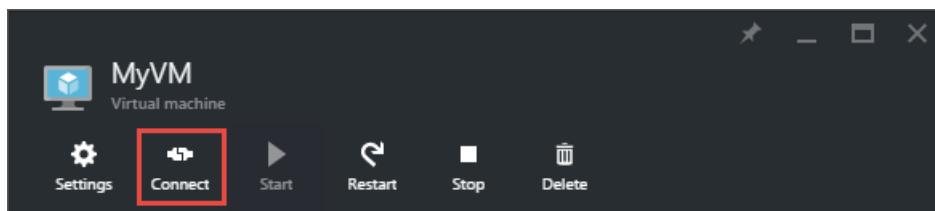
- Click **Summary** to review your choices. When you see the **Validation passed** message, click **OK**.



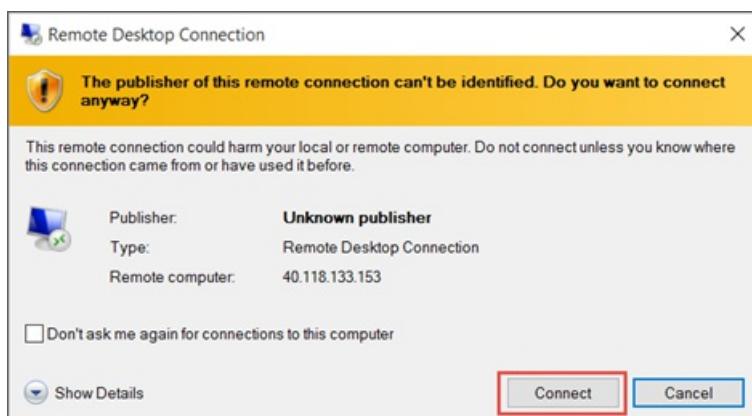
- While Azure creates the virtual machine, you can track the progress by clicking on **Virtual Machines** on left. When the VM has been created, the status will change to **Running**.

Connect to the virtual machine and sign on

- On the left, click **Virtual Machines**.
- Select the virtual machine from the list.
- On the blade for the virtual machine, click **Connect**. This creates and downloads a Remote Desktop Protocol file (.rdp file) that is like a shortcut to connect to your machine. You might want to save the file to your desktop for easy access. **Open** this file to connect to your VM.

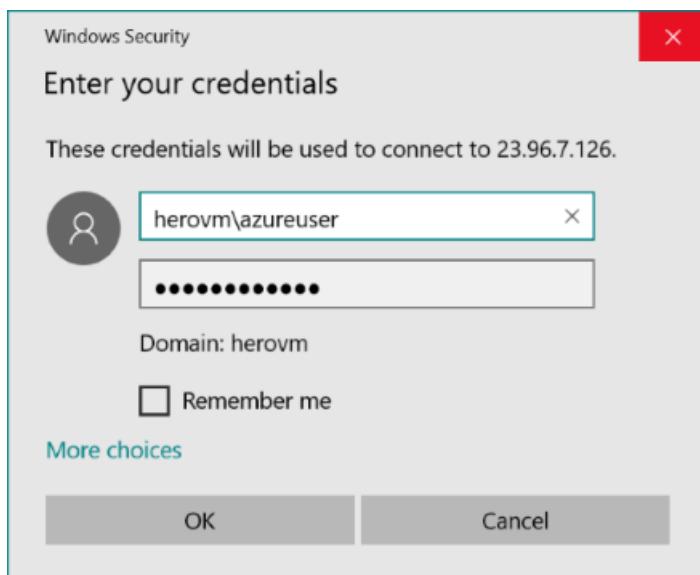


- You get a warning that the .rdp is from an unknown publisher. This is normal. In the Remote Desktop window, click **Connect** to continue.

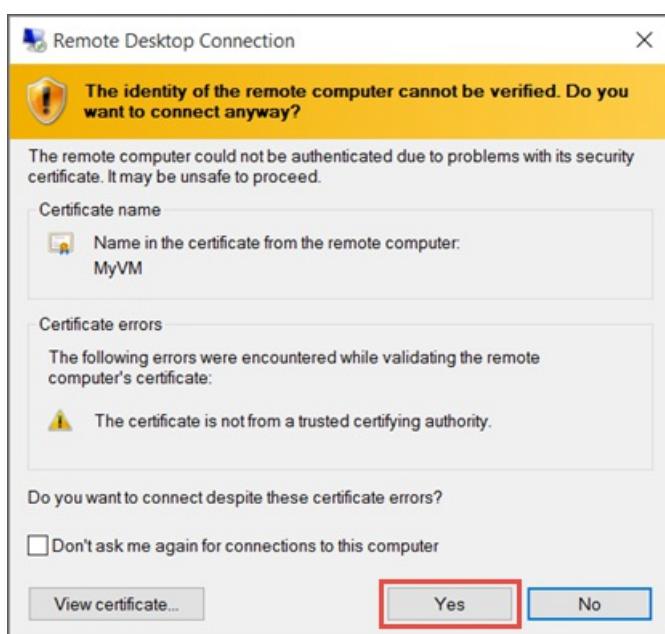


- In the Windows Security window, type the username and password for the local account that you created

when you created the VM. The username is entered as `vmname\username`, then click **OK**.



6. You get a warning that the certificate cannot be verified. This is normal. Click **Yes** to verify the identity of the virtual machine and finish logging on.

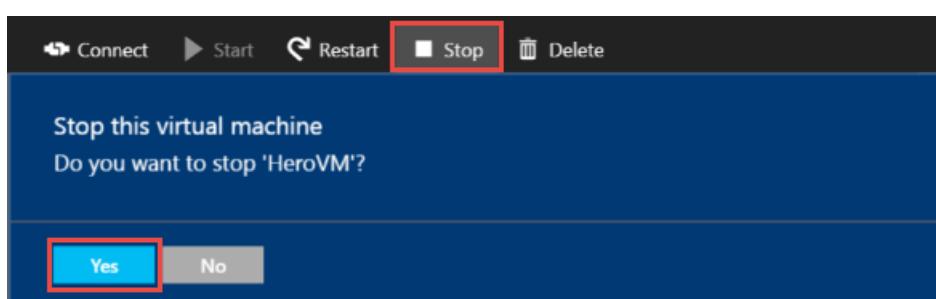


If you run in to trouble when you try to connect, see [Troubleshoot Remote Desktop connections to a Windows-based Azure Virtual Machine](#).

You can now work with the virtual machine as you would with any other server.

Optional: Stop the VM

It is a good idea to stop the VM so you don't incur charges when you aren't actually using it. Just click **Stop** and then click **Yes**.



Click the **Start** button to restart the VM when you're ready to use it again.

Next steps

- You can experiment with your new VM by [installing IIS](#). This tutorial also shows how to open port 80 to incoming web traffic using a network security group (NSG).
- You can also [create a Windows VM by using PowerShell](#) or [create a Linux virtual machine](#) by using the Azure CLI.
- If you're interested in automating deployments, see [Create a Windows virtual machine by using a Resource Manager template](#).

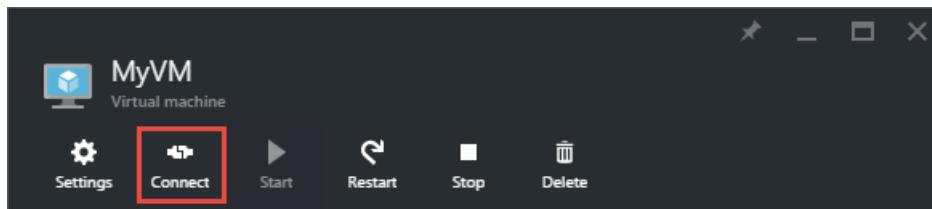
How to connect and log on to an Azure virtual machine running Windows

1/17/2017 • 1 min to read • [Edit on GitHub](#)

You'll use the **Connect** button in the Azure portal to start a Remote Desktop (RDP) session. First you connect to the virtual machine, then you log on.

Connect to the virtual machine

1. If you haven't already done so, sign in to the [Azure portal](#).
2. On the Hub menu, click **Virtual Machines**.
3. Select the virtual machine from the list.
4. On the blade for the virtual machine, click **Connect**.

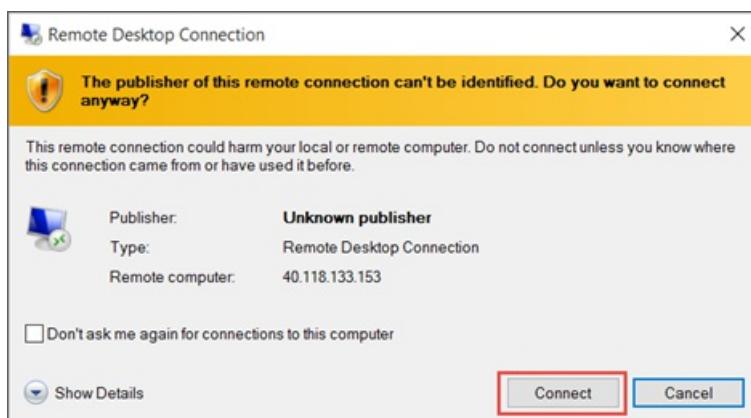


TIP

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 will get a warning that the .rdp is from an unknown publisher. This is normal. In the Remote Desktop window, click **Connect** to continue.



3. In the **Windows Security** window, type the credentials for an account on the virtual machine and then click **OK**.

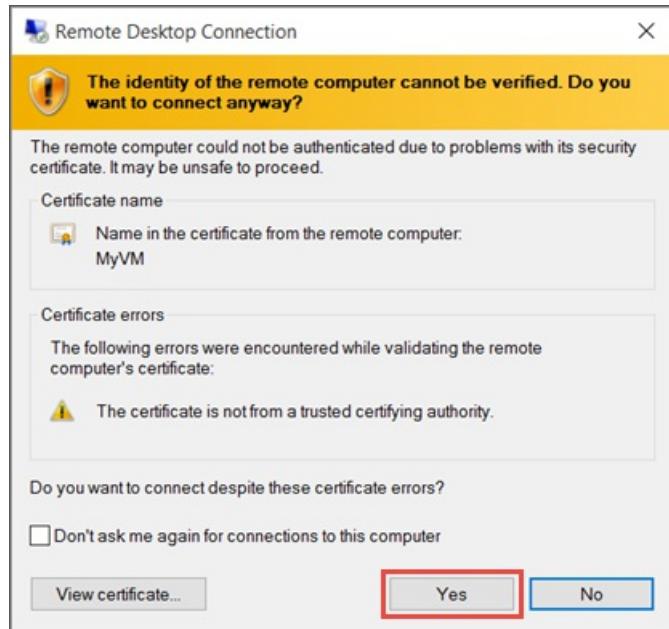
Local account - this is usually the local account user name and password that you specified when you created the virtual machine. In this case, the domain is the name of the virtual machine and it is entered as

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.

Experiment with installing a role on your Windows VM

1/17/2017 • 3 min to read • [Edit on GitHub](#)

Once you have your first virtual machine (VM) up and running, you can move on to installing software and services. For this tutorial, we are going to use Server Manager on the Windows Server VM to install IIS. Then, we will create a Network Security Group (NSG) using the Azure portal to open port 80 to IIS traffic.

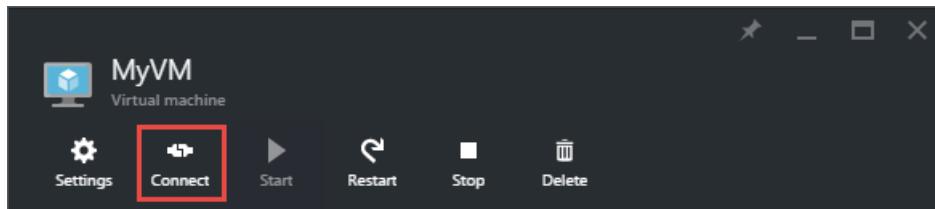
If you haven't already created your first VM, you should go back to [Create your first Windows virtual machine in the Azure portal](#) before continuing with this tutorial.

Make sure the VM is running

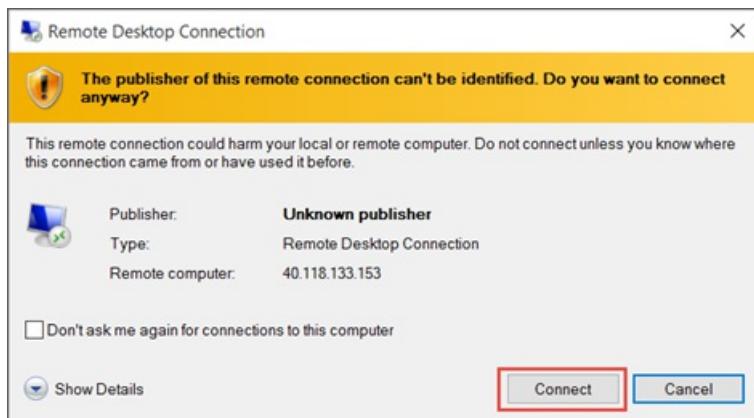
1. Open the [Azure portal](#).
2. On the hub menu, click **Virtual Machines**. Select the virtual machine from the list.
3. If the status is **Stopped (Deallocated)**, click the **Start** button on the **Essentials** blade of the VM. If the status is **Running**, you can move on to the next step.

Connect to the virtual machine and sign in

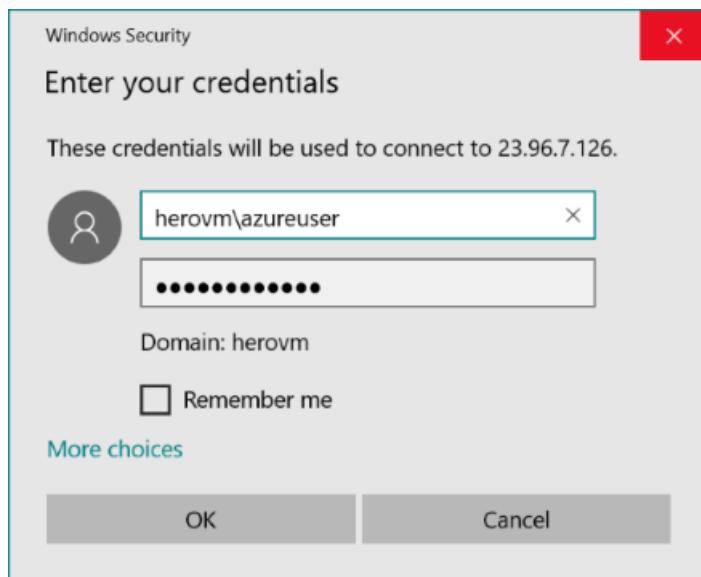
1. On the hub menu, click **Virtual Machines**. Select the virtual machine from the list.
2. On the blade for the virtual machine, click **Connect**. This creates and downloads a Remote Desktop Protocol file (.rdp file) that is like a shortcut to connect to your machine. You might want to save the file to your desktop for easy access. **Open** this file to connect to your VM.



3. You get a warning that the .rdp is from an unknown publisher. This is normal. In the Remote Desktop window, click **Connect** to continue.



4. In the Windows Security window, type the username and password for the local account that you created when you created the VM. The username is entered as `vmname\username`, then click **OK**.



5. You get a warning that the certificate cannot be verified. This is normal. Click **Yes** to verify the identity of the virtual machine and finish logging on.

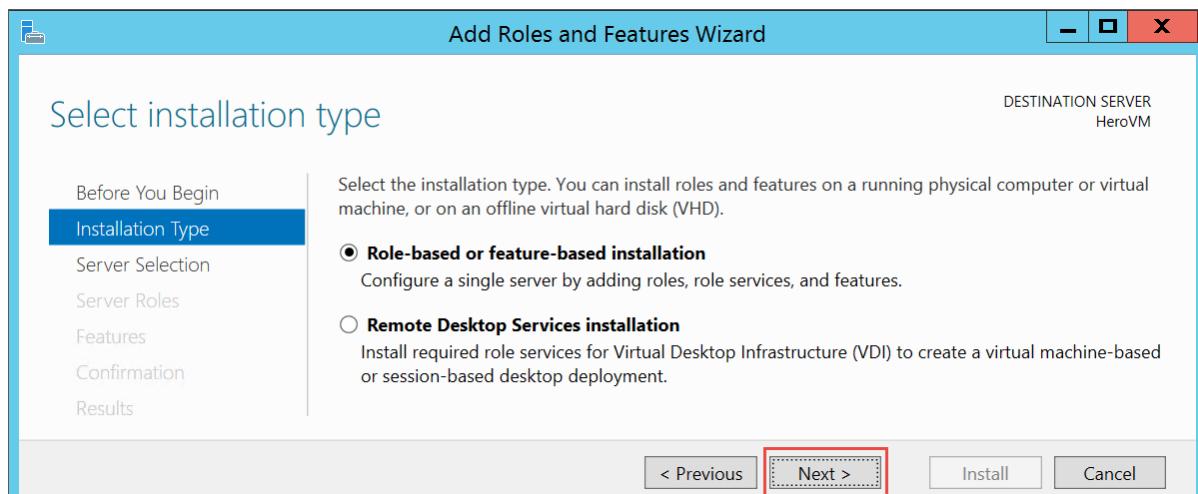


If you run in to trouble when you try to connect, see [Troubleshoot Remote Desktop connections to a Windows-based Azure Virtual Machine](#).

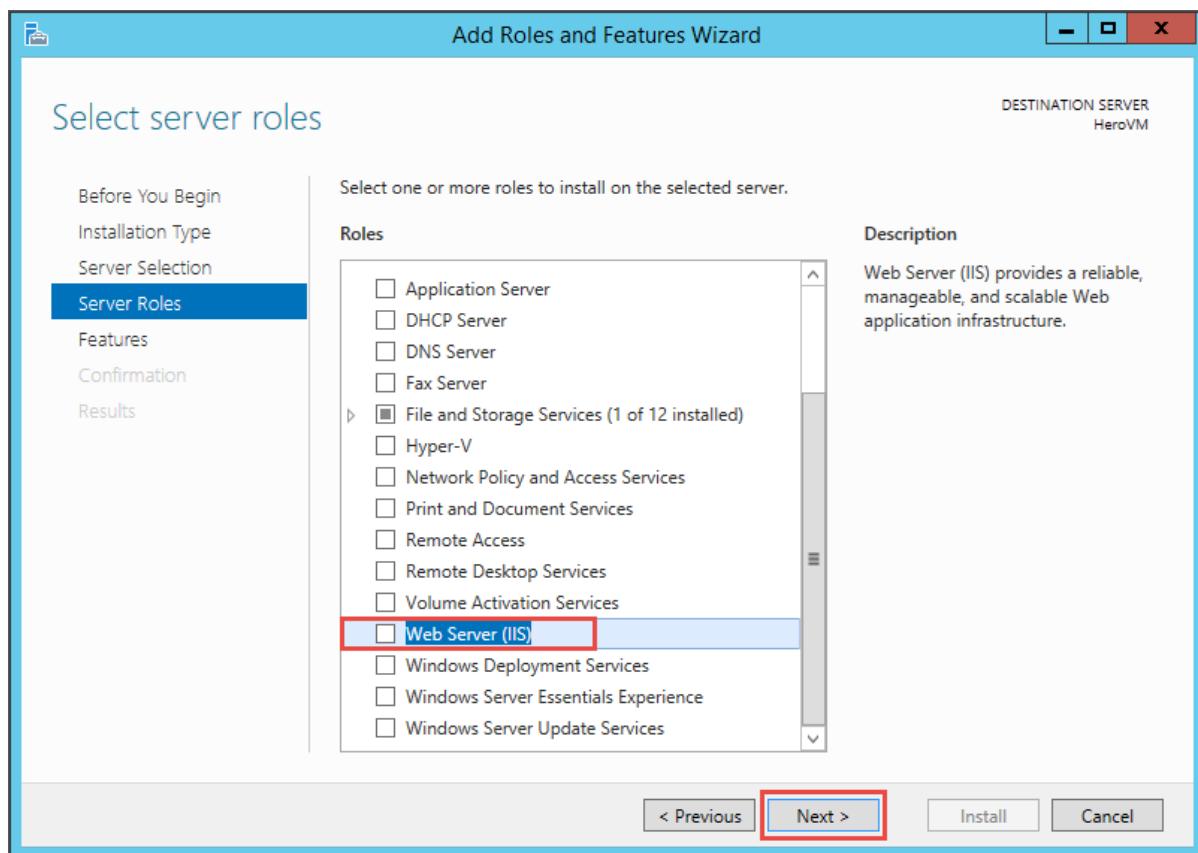
Install IIS on your VM

Now that you are logged in to the VM, we will install a server role so that you can experiment more.

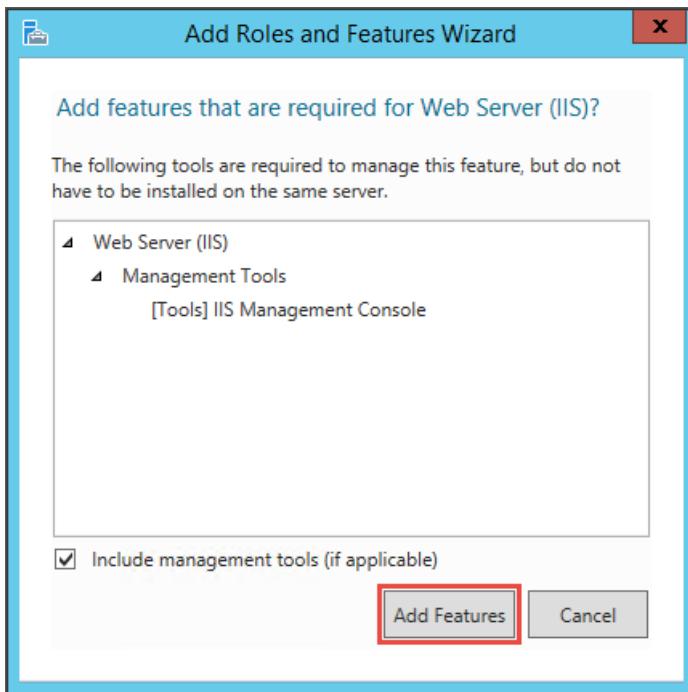
1. Open **Server Manager** if it isn't already open. Click the **Start** menu, and then click **Server Manager**.
2. In **Server Manager**, select **Local Server** from the left pane.
3. In the menu, select **Manage > Add Roles and Features**.
4. In the Add Roles and Features Wizard, on the **Installation Type** page, choose **Role-based or feature-based installation**, and then click **Next**.



5. Select the VM from the server pool and click **Next**.
6. On the **Server Roles** page, select **Web Server (IIS)**.



7. In the pop-up about adding features needed for IIS, make sure that **Include management tools** is selected and then click **Add Features**. When the pop-up closes, click **Next** in the wizard.

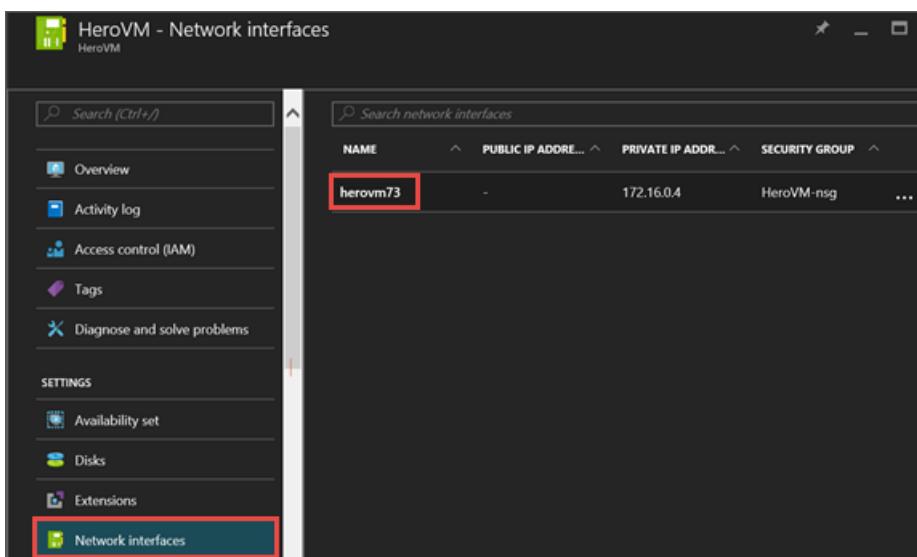


8. On the features page, click **Next**.
9. On the **Web Server Role (IIS)** page, click **Next**.
10. On the **Role Services** page, click **Next**.
11. On the **Confirmation** page, click **Install**.
12. When the installation is complete, click **Close** on the wizard.

Open port 80

In order for your VM to accept inbound traffic over port 80, you need to add an inbound rule to the network security group.

1. Open the [Azure portal](#).
2. In **Virtual machines** select the VM that you created.
3. In the virtual machines settings, select **Network interfaces** and then select the existing network interface.



4. In **Essentials** for the network interface, click the **Network security group**.

Essentials ^

Resource group NewHeroRG	Private IP address 172.16.0.4
Location West US	Virtual network/subnet HeroRG-vnet/default
Subscription name Microsoft Azure Internal Consumption	Public IP address HeroVM-ip
Subscription ID 7891b050-2d44-4a2c-8b96-3915868fe407	Network security group HeroVM-nsg
	Attached to HeroVM

5. In the **Essentials** blade for the NSG, you should have one existing default inbound rule for **default-allow-rdp** which allows you to log in to the VM. You will add another inbound rule to allow IIS traffic. Click **Inbound security rule**.

HeroVM-nsg
Network security group

Overview

Activity log

Access control (IAM)

Tags

SETTINGS

Inbound security rules (highlighted with a red box)

Outbound security rules

Network interfaces

Essentials ^

Resource group
NewHeroRG

Location
West US

Subscription name
Microsoft Azure Internal Consumption

Subscription ID
7891b050-2d44-4a2c-8b96-3915868fe407

Security rules
1 inbound, 0 outbound

Associated with
0 subnets, 1 network interfaces

1 Inbound security rule

PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
1000	default-allow-rdp	Any	Any	Custom (Any/3389)	Allow

6. In **Inbound security rules**, click **Add**.

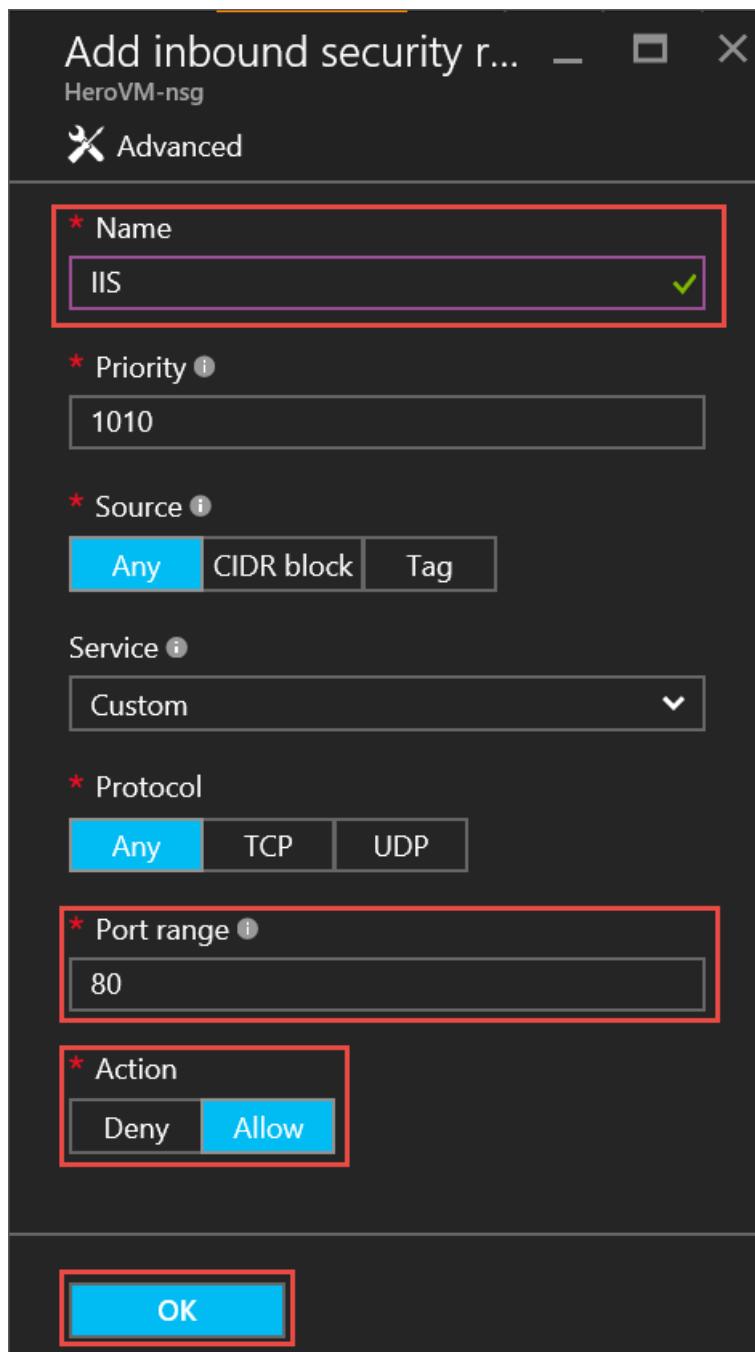
+ Add

Default rules

Inbound security rule

PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
1000	default-allow-rdp	Any	Any	Custom (Any/33...)	Allow

7. In **Inbound security rules**, click **Add**. Type **80** in the port range and make sure **Allow** is selected. When you are done, click **OK**.



For more information about NSGs, inbound and outbound rules, see [Allow external access to your VM using the Azure portal](#)

Connect to the default IIS website

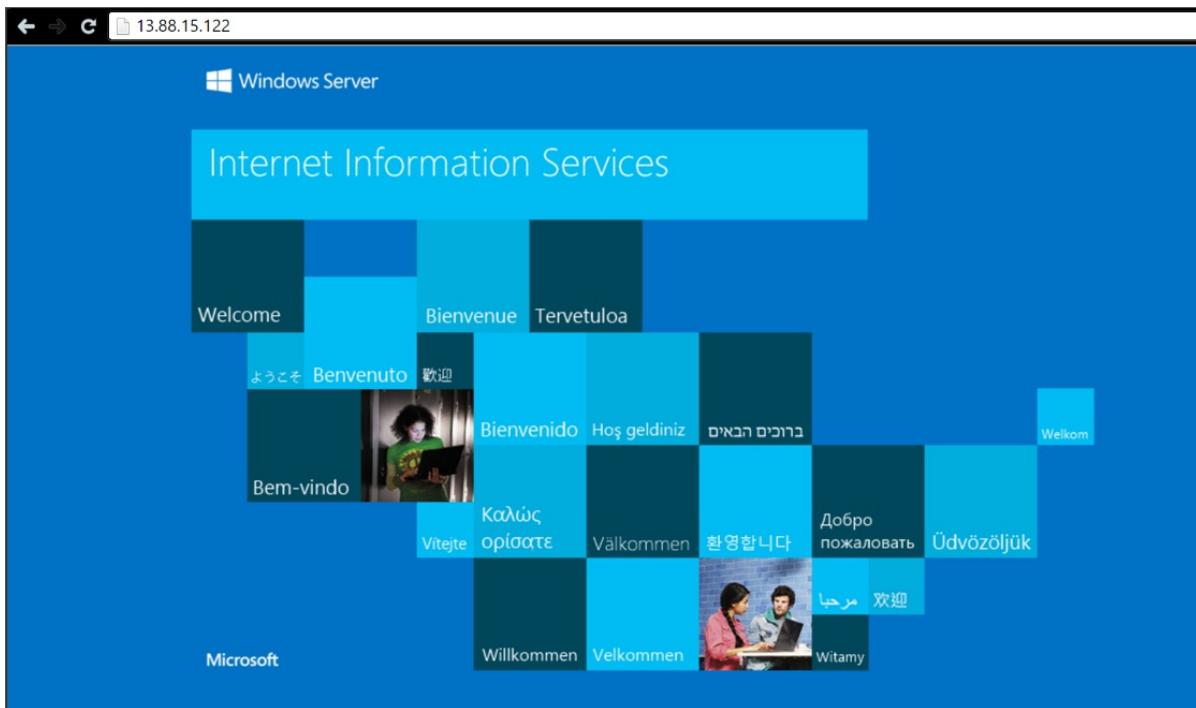
1. In the Azure portal, click **Virtual machines** and then select your VM.
2. In the **Essentials** blade, copy your **Public IP address**.

The Essentials blade displays the following information for the HeroVM virtual machine:

Resource group	NewHeroRG	Computer name	HeroVM
Status	Running	Operating system	Windows
Location	West US	Size	Standard DS1 v2 (1 core, 3.5 GB memory)
Subscription name	Microsoft Azure Internal Consumption	Public IP address/DNS name label	13.88.15.122<none>
Subscription ID		Virtual network/subnet	HeroRG-vnet/default

3. Open a browser and in the address bar, type in your public IP address like this: http:// and click **Enter** to go to that address.

4. Your browser should open the default IIS web page. It looks something like this:



Next steps

- You can also experiment with [attaching a data disk](#) to your virtual machine. Data disks provide more storage for your virtual machine.

Different ways to create a Windows VM

1/17/2017 • 1 min to read • [Edit on GitHub](#)

Azure offers different ways to create a virtual machine because virtual machines are suited for different users and purposes. This means that you need to make some choices about the virtual machine and how to create it. This article gives you a summary of these choices and links to instructions.

Azure portal

Using the Azure portal is a simple way to try out a virtual machine, especially if you're just starting out with Azure.

[Create a virtual machine running Windows using the portal](#)

Template

Virtual machines require a combination of resources (such as availability sets and storage accounts). Rather than deploying and managing each resource separately, you can create an Azure Resource Manager template that deploys and provisions all of the resources in a single, coordinated operation.

- [Create a Windows virtual machine with a Resource Manager template](#)

Azure PowerShell

If you prefer working in a command shell, you can use Azure PowerShell.

- [Create a Windows VM using PowerShell](#)

Visual Studio

Use Visual Studio to build, manage, and deploy VMs with the Azure Tools for Visual Studio and the Azure SDK.

[Azure Tools for Visual Studio](#)

Create a Windows VM using Resource Manager and PowerShell

1/17/2017 • 4 min to read • [Edit on GitHub](#)

This article shows you how to quickly create an Azure Virtual Machine running Windows Server and the resources it needs using [Resource Manager](#) and PowerShell.

All the steps in this article are required to create a virtual machine and it should take about 30 minutes to do the steps. Replace example parameter values in the commands with names that make sense for your environment.

Step 1: 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.

Step 2: Create a resource group

All resources must be contained in a resource group, so let's create that first.

1. Get a list of available locations where resources can be created.

```
Get-AzureRmLocation | sort Location | Select Location
```

2. Set the location for the resources. This command sets the location to **centralus**.

```
$location = "centralus"
```

3. Create a resource group. This command creates the resource group named **myResourceGroup** in the location that you set.

```
$myResourceGroup = "myResourceGroup"  
New-AzureRmResourceGroup -Name $myResourceGroup -Location $location
```

Step 3: Create a storage account

A [storage account](#) is needed to store the virtual hard disk that is used by the virtual machine that you create. Storage account names must be between 3 and 24 characters in length and may contain numbers and lowercase letters only.

1. Test the storage account name for uniqueness. This command tests the name **myStorageAccount**.

```
$myStorageAccountName = "mystorageaccount"  
Get-AzureRmStorageAccountNameAvailability $myStorageAccountName
```

If this command returns **True**, your proposed name is unique within Azure.

2. Now, create the storage account.

```
$myStorageAccount = New-AzureRmStorageAccount -ResourceGroupName $myResourceGroup `  
-Name $myStorageAccountName -SkuName "Standard_LRS" -Kind "Storage" -Location $location
```

Step 4: Create a virtual network

All virtual machines are part of a [virtual network](#).

1. Create a subnet for the virtual network. This command creates a subnet named **mySubnet** with an address prefix of 10.0.0.0/24.

```
$mySubnet = New-AzureRmVirtualNetworkSubnetConfig -Name "mySubnet" -AddressPrefix 10.0.0.0/24
```

2. Now, create the virtual network. This command creates a virtual network named **myVnet** using the subnet that you created and an address prefix of **10.0.0.0/16**.

```
$myVnet = New-AzureRmVirtualNetwork -Name "myVnet" -ResourceGroupName $myResourceGroup `  
-Location $location -AddressPrefix 10.0.0.0/16 -Subnet $mySubnet
```

Step 5: 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 the public IP address. This command creates a public IP address named **myPublicIp** with an allocation method of **Dynamic**.

```
$myPublicIp = New-AzureRmPublicIpAddress -Name "myPublicIp" -ResourceGroupName $myResourceGroup `  
-Location $location -AllocationMethod Dynamic
```

2. Create the network interface. This command creates a network interface named **myNIC**.

```
$myNIC = New-AzureRmNetworkInterface -Name "myNIC" -ResourceGroupName $myResourceGroup `  
-Location $location -SubnetId $myVnet.Subnets[0].Id -PublicIpAddressId $myPublicIp.Id
```

Step 6: Create a virtual machine

Now that you have all the pieces in place, it's time to create the virtual machine.

1. Run this command to set the administrator account name and password for the virtual machine.

```
$cred = Get-Credential -Message "Type the name and password of the local administrator account."
```

The password must be at 12-123 characters long and have at least one lower case character, one upper case character, one number, and one special character.

2. Create the configuration object for the virtual machine. This command creates a configuration object named **myVmConfig** that defines the name of the VM and the size of the VM.

```
$myVm = New-AzureRmVMConfig -VMName "myVM" -VMSize "Standard_DS1_v2"
```

See [Sizes for virtual machines in Azure](#) for a list of available sizes for a virtual machine.

3. Configure operating system settings for the VM. This command sets the computer name, operating system type, and account credentials for the VM.

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

4. Define the image to use to provision the VM. This command defines the Windows Server image to use for the VM.

```
$myVM = Set-AzureRmVMSourceImage -VM $myVM -PublisherName "MicrosoftWindowsServer" `  
-Offer "WindowsServer" -Skus "2012-R2-Datacenter" -Version "latest"
```

For more information about selecting images to use, see [Navigate and select Windows virtual machine images in Azure with PowerShell or the CLI](#).

5. Add the network interface that you created to the configuration.

```
$myVM = Add-AzureRmVMNetworkInterface -VM $myVM -Id $myNIC.Id
```

6. Define the name and location of the VM hard disk. The virtual hard disk file is stored in a container. This command creates the disk in a container named **vhds/myOsDisk1.vhd** in the storage account that you created.

```
$blobPath = "vhds/myOsDisk1.vhd"  
$osDiskUri = $myStorageAccount.PrimaryEndpoints.Blob.ToString() + $blobPath
```

7. Add the operating system disk information to the VM configuration. Replace The value of **\$diskName** with a name for the operating system disk. Create the variable and add the disk information to the configuration.

```
$myVM = Set-AzureRmVMOSDisk -VM $myVM -Name "myOsDisk1" -VhdUri $osDiskUri -CreateOption fromImage
```

8. Finally, create the virtual machine.

```
New-AzureRmVM -ResourceGroupName $myResourceGroup -Location $location -VM $myVM
```

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 manage the virtual machine that you created by reviewing [Manage virtual machines using Azure Resource Manager and PowerShell](#).
- Take advantage of using a template to create a virtual machine by using the information in [Create a Windows virtual machine with a Resource Manager template](#)

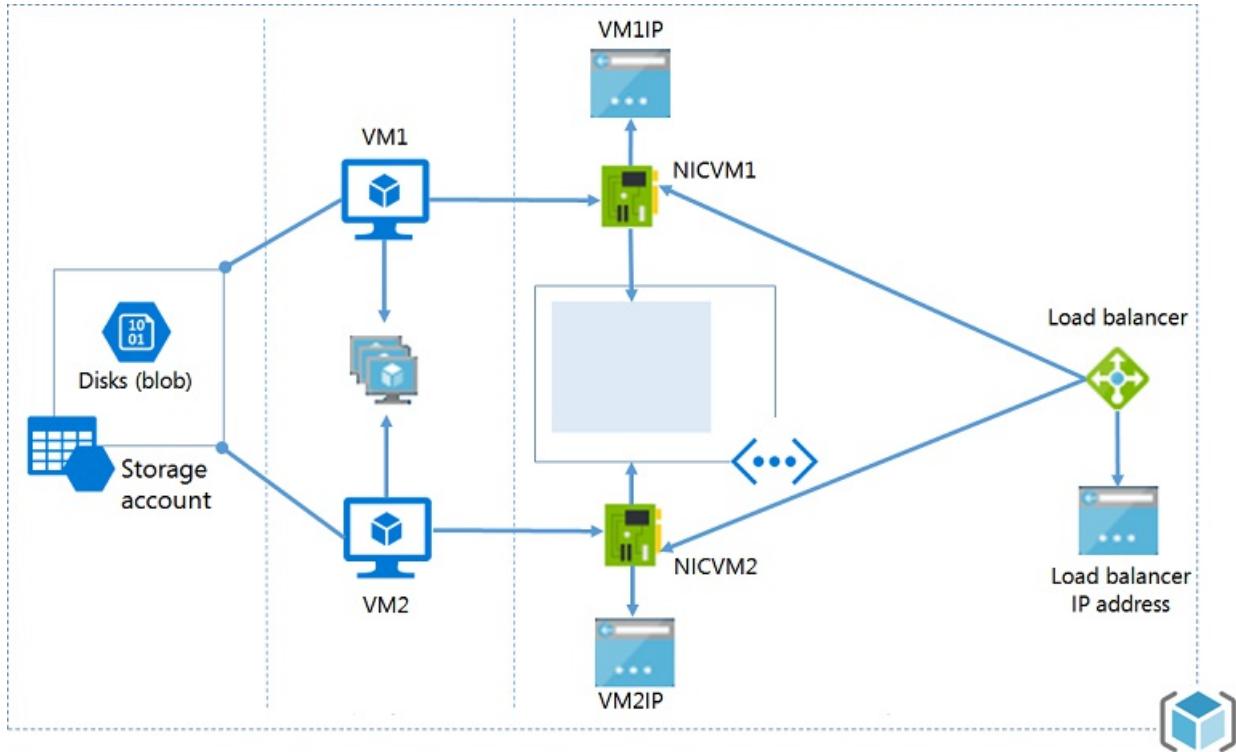
Resource Manager template walkthrough

1/17/2017 • 12 min to read • [Edit on GitHub](#)

One of the first questions when creating a template is "how to start?". One can start from a blank template, following the basic structure described in [Authoring Template article](#), and add the resources and appropriate parameters and variables. A good alternative would be to start by going through the [quickstart gallery](#) and look for similar scenarios to the one you are trying to create. You can merge several templates or edit an existing one to suit your own specific scenario.

Let's take a look at a common infrastructure:

- Two virtual machines that use the same storage account, are in the same availability set, and on the same subnet of a virtual network.
- A single NIC and VM IP address for each virtual machine.
- A load balancer with a load balancing rule on port 80



This topic walks you through the steps of creating a Resource Manager template for that infrastructure. The final template you create is based on a Quickstart template called [2 VMs in a Load Balancer and load balancing rules](#).

But, that's a lot to build all at once, so let's first create a storage account and deploy it. After you have mastered creating the storage account, you will add the other resources and re-deploy the template to complete the infrastructure.

NOTE

You can use any type of editor when creating the template. Visual Studio provides tools that simplify template development, but you do not need Visual Studio to complete this tutorial. For a tutorial on using Visual Studio to create a Web App and SQL Database deployment, see [Creating and deploying Azure resource groups through Visual Studio](#).

Create the Resource Manager template

The template is a JSON file that defines all of the resources you will deploy. It also permits you to define parameters that are specified during deployment, variables that are constructed from other values and expressions, and outputs from the deployment.

Let's start with the simplest template:

```
{  
  "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {},  
  "variables": {},  
  "resources": [],  
  "outputs": {}  
}
```

Save this file as **azureddeploy.json** (note that the template can have any name you want, just that it must be a json file).

Create a storage account

Within the **resources** section, add an object that defines the storage account, as shown below.

```
"resources": [  
  {  
    "type": "Microsoft.Storage/storageAccounts",  
    "name": "[parameters('storageAccountName')]",  
    "apiVersion": "2015-06-15",  

```

You may be wondering where these properties and values come from. The properties **type**, **name**, **apiVersion**, and **location** are standard elements that are available for all resource types. You can learn about the common elements at [Resources](#). **name** is set to a parameter value that you pass in during deployment and **location** as the location used by the resource group. We'll look at how you determine **type** and **apiVersion** in the sections below.

The **properties** section contains all of the properties that are unique to a particular resource type. The values you specify in this section exactly match the PUT operation in the REST API for creating that resource type. When creating a storage account, you must provide an **accountType**. Notice in the [REST API for creating a Storage account](#) that the properties section of the REST operation also contains an **accountType** property, and the permitted values are documented. In this example, the account type is set to **Standard_LRS**, but you could specify some other value or permit users to pass in the account type as a parameter.

Now let's jump back to the **parameters** section, and see how you define the name of the storage account. You can learn more about the use of parameters at [Parameters](#).

```
"parameters" : {  
  "storageAccountName": {  
    "type": "string",  
    "metadata": {  
      "description": "Storage Account Name"  
    }  
  }  
}
```

Here you defined a parameter of type string that will hold the name of the storage account. The value for this parameter will be provided during template deployment.

Deploying the template

We have a full template for creating a new storage account. As you recall, the template was saved in **azuredeploy.json** file:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {  
    "storageAccountName": {  
      "type": "string",  
      "metadata": {  
        "description": "Storage Account Name"  
      }  
    }  
  },  
  "resources": [  
    {  
      "type": "Microsoft.Storage/storageAccounts",  
      "name": "[parameters('storageAccountName')]",  
      "apiVersion": "2015-06-15",  
      "location": "[resourceGroup().location]",  
      "properties": {  
        "accountType": "Standard_LRS"  
      }  
    }  
  ]  
}
```

There are quite a few ways to deploy a template, as you can see in the [Resource Deployment article](#). To deploy the template using Azure PowerShell, use:

```
# create a new resource group  
New-AzureRmResourceGroup -Name ExampleResourceGroup -Location "West Europe"  
  
# deploy the template to the resource group  
New-AzureRmResourceGroupDeployment -Name ExampleDeployment -ResourceGroupName ExampleResourceGroup `  
  -TemplateFile azuredeploy.json
```

Or, to deploy the template using Azure CLI, use:

```
azure group create -n ExampleResourceGroup -l "West Europe"  
  
azure group deployment create -f azuredeploy.json -g ExampleResourceGroup -n ExampleDeployment
```

You are now the proud owner of a storage account!

The next steps will be to add all the resources required to deploy the architecture described in the start of this tutorial. You will add these resources in the same template you have been working on.

Availability Set

After the definition for the storage account, add an availability set for the virtual machines. In this case, there are no additional properties required, so its definition is fairly simple. See the [REST API for creating an Availability Set](#) for the full properties section, in case you want to define the update domain count and fault domain count values.

```
{  
  "type": "Microsoft.Compute/availabilitySets",  
  "name": "[variables('availabilitySetName')]",  
  "apiVersion": "2015-06-15",  
  "location": "[resourceGroup().location]",  
  "properties": {}  
}
```

Notice that the **name** is set to the value of a variable. For this template, the name of the availability set is needed in a few different places. You can more easily maintain your template by defining that value once and using it in multiple places.

The value you specify for **type** contains both the resource provider and the resource type. For availability sets, the resource provider is **Microsoft.Compute** and the resource type is **availabilitySets**. You can get the list of available resource providers by running the following PowerShell command:

```
Get-AzureRmResourceProvider -ListAvailable
```

Or, if you are using Azure CLI, you can run the following command:

```
azure provider list
```

Given that in this topic you are creating with storage accounts, virtual machines, and virtual networking, you will work with:

- Microsoft.Storage
- Microsoft.Compute
- Microsoft.Network

To see the resource types for a particular provider, run the following PowerShell command:

```
(Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Compute).ResourceTypes
```

Or, for Azure CLI, the following command will return the available types in JSON format and save it to a file.

```
azure provider show Microsoft.Compute --json > c:\temp.json
```

You should see **availabilitySets** as one of the types within **Microsoft.Compute**. The full name of the type is **Microsoft.Compute/availabilitySets**. You can determine the resource type name for any of the resources in your template.

Public IP

Define a public IP address. Again, look at the [REST API for public IP addresses](#) for the properties to set.

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/publicIPAddresses",
  "name": "[parameters('publicIPAddressName')]",
  "location": "[resourceGroup().location]",
  "properties": {
    "publicIPAllocationMethod": "Dynamic",
    "dnsSettings": {
      "domainNameLabel": "[parameters('dnsNameforLBIP')]"
    }
  }
}
```

The allocation method is set to **Dynamic** but you could set it to the value you need or set it to accept a parameter value. You have enabled users of your template to pass in a value for the domain name label.

Now, let's look at how you determine the **apiVersion**. The value you specify simply matches the version of the REST API that you want to use when creating the resource. So, you can look at the REST API documentation for that resource type. Or, you can run the following PowerShell command for a particular type.

```
((Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Network).ResourceTypes | Where-Object
ResourceTypeName -eq publicIPAddresses).ApiVersions
```

Which returns the following values:

```
2015-06-15
2015-05-01-preview
2014-12-01-preview
```

To see the API versions with Azure CLI, run the same **azure provider show** command shown previously.

When creating a new template, pick the most recent API version.

Virtual network and subnet

Create a virtual network with one subnet. Look at the [REST API for virtual networks](#) for all the properties to set.

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/virtualNetworks",
  "name": "[parameters('vnetName')]",
  "location": "[resourceGroup().location]",
  "properties": {
    "addressSpace": {
      "addressPrefixes": [
        "10.0.0.0/16"
      ]
    },
    "subnets": [
      {
        "name": "[variables('subnetName')]",
        "properties": {
          "addressPrefix": "10.0.0.0/24"
        }
      }
    ]
  }
}
```

Load balancer

Now you will create an external facing load balancer. Because this load balancer uses the public IP address, you must declare a dependency on the public IP address in the **dependsOn** section. This means the load balancer will not get deployed until the public IP address has finished deploying. Without defining this dependency, you will receive an error because Resource Manager will attempt to deploy the resources in parallel, and will try to set the load balancer to public IP address that doesn't exist yet.

You will also create a backend address pool, a couple of inbound NAT rules to RDP into the VMs, and a load balancing rule with a tcp probe on port 80 in this resource definition. Checkout the [REST API for load balancer](#) for all the properties.

```
{
  "apiVersion": "2015-06-15",
  "name": "[parameters('lbName')]",
  "type": "Microsoft.Network/loadBalancers",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Network/publicIPAddresses/', parameters('publicIPAddressName'))]"
  ],
  "properties": {
    "frontendIPConfigurations": [
      {
        "name": "LoadBalancerFrontEnd",
        "properties": {
          "publicIPAddress": {
            "id": "[variables('publicIPAddressID')]"
          }
        }
      }
    ],
    "backendAddressPools": [
      {
        "name": "BackendPool1"
      }
    ],
    "inboundNatRules": [
      {
        "name": "RDP-VM0",
        "properties": {
          "frontendIPConfiguration": {
            "id": "[variables('frontEndIPConfigID')]"
          },
          "protocol": "tcp",
          "frontendPort": 50001,
          "backendPort": 3389,
          "enableFloatingIP": false
        }
      },
      {
        "name": "RDP-VM1",
        "properties": {
          "frontendIPConfiguration": {
            "id": "[variables('frontEndIPConfigID')]"
          },
          "protocol": "tcp",
          "frontendPort": 50002,
          "backendPort": 3389,
          "enableFloatingIP": false
        }
      }
    ],
    "loadBalancingRules": [
      {
        "name": "LBRule",
        "properties": {
          "frontendIPConfiguration": {
            "id": "[variables('frontEndIPConfigID')]"
          }
        }
      }
    ]
  }
}
```

```

    },
    "backendAddressPool": {
      "id": "[variables('lbPoolID')]"
    },
    "protocol": "tcp",
    "frontendPort": 80,
    "backendPort": 80,
    "enableFloatingIP": false,
    "idleTimeoutInMinutes": 5,
    "probe": {
      "id": "[variables('lbProbeID')]"
    }
  }
],
"probes": [
  {
    "name": "tcpProbe",
    "properties": {
      "protocol": "tcp",
      "port": 80,
      "intervalInSeconds": 5,
      "numberOfProbes": 2
    }
  }
]
}
}

```

Network interface

You will create 2 network interfaces, one for each VM. Rather than having to include duplicate entries for the network interfaces, you can use the [copyIndex\(\) function](#) to iterate over the copy loop (referred to as nicLoop) and create the number network interfaces as defined in the `numberOfInstances` variables. The network interface depends on creation of the virtual network and the load balancer. It uses the subnet defined in the virtual network creation, and the load balancer id to configure the load balancer address pool and the inbound NAT rules. Look at the [REST API for network interfaces](#) for all the properties.

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/networkInterfaces",
  "name": "[concat(parameters('nicNamePrefix'), copyindex())]",
  "location": "[resourceGroup().location]",
  "copy": {
    "name": "nicLoop",
    "count": "[variables('numberOfInstances')]"
  },
  "dependsOn": [
    "[concat('Microsoft.Network/virtualNetworks/', parameters('vnetName'))]",
    "[concat('Microsoft.Network/loadBalancers/', parameters('lbName'))]"
  ],
  "properties": {
    "ipConfigurations": [
      {
        "name": "ipconfig1",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
          "subnet": {
            "id": "[variables('subnetRef')]"
          },
          "loadBalancerBackendAddressPools": [
            {
              "id": "[concat(variables('lbID'), '/backendAddressPools/BackendPool1')]"
            }
          ],
          "loadBalancerInboundNatRules": [
            {
              "id": "[concat(variables('lbID'), '/inboundNatRules/RDP-VM', copyindex())]"
            }
          ]
        }
      }
    ]
  }
}
```

Virtual machine

You will create 2 virtual machines, using `copyIndex()` function, as you did in creation of the [network interfaces](#). The VM creation depends on the storage account, network interface and availability set. This VM will be created from a marketplace image, as defined in the `storageProfile` property - `imageReference` is used to define the image publisher, offer, sku and version. Finally, a diagnostic profile is configured to enable diagnostics for the VM.

To find the relevant properties for a marketplace image, follow the [select Linux virtual machine images](#) or [select Windows virtual machine images](#) articles.

```
{
    "apiVersion": "2015-06-15",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[concat(parameters('vmNamePrefix'), copyindex())]",
    "copy": {
        "name": "virtualMachineLoop",
        "count": "[variables('numberOfInstances')]"
    },
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[concat('Microsoft.Storage/storageAccounts/', parameters('storageAccountName'))]",
        "[concat('Microsoft.Network/networkInterfaces/', parameters('nicNamePrefix'), copyindex())]",
        "[concat('Microsoft.Compute/availabilitySets/', variables('availabilitySetName'))]"
    ],
    "properties": {
        "availabilitySet": {
            "id": "[resourceId('Microsoft.Compute/availabilitySets', variables('availabilitySetName'))]"
        },
        "hardwareProfile": {
            "vmSize": "[parameters('vmSize')]"
        },
        "osProfile": {
            "computerName": "[concat(parameters('vmNamePrefix'), copyIndex())]",
            "adminUsername": "[parameters('adminUsername')]",
            "adminPassword": "[parameters('adminPassword')]"
        },
        "storageProfile": {
            "imageReference": {
                "publisher": "[parameters('imagePublisher')]",
                "offer": "[parameters('imageOffer')]",
                "sku": "[parameters('imageSKU')]",
                "version": "latest"
            },
            "osDisk": {
                "name": "osdisk",
                "vhd": {
                    "uri": "[concat('http://', parameters('storageAccountName'), '.blob.core.windows.net/vhds/', 'osdisk', copyindex(), '.vhd')]"
                },
                "caching": "ReadWrite",
                "createOption": "FromImage"
            }
        },
        "networkProfile": {
            "networkInterfaces": [
                {
                    "id": "[resourceId('Microsoft.Network/networkInterfaces', concat(parameters('nicNamePrefix'), copyindex()))]"
                }
            ]
        },
        "diagnosticsProfile": {
            "bootDiagnostics": {
                "enabled": "true",
                "storageUri": "[concat('http://', parameters('storageAccountName'), '.blob.core.windows.net')]"
            }
        }
    }
}
```

NOTE

For images published by **3rd party vendors**, you will need to specify another property named `plan`. An example can be found in [this template](#) from the quickstart gallery.

You have finished defining the resources for your template.

Parameters

In the parameters section, define the values that can be specified when deploying the template. Only define parameters for values that you think should be varied during deployment. You can provide a default value for a parameter that is used if one is not provided during deployment. You can also define the allowed values as shown for the **imageSKU** parameter.

```
"parameters": {  
    "storageAccountName": {  
        "type": "string",  
        "metadata": {  
            "description": "Name of storage account"  
        }  
    },  
    "adminUsername": {  
        "type": "string",  
        "metadata": {  
            "description": "Admin username"  
        }  
    },  
    "adminPassword": {  
        "type": "securestring",  
        "metadata": {  
            "description": "Admin password"  
        }  
    },  
    "dnsNameforLBIP": {  
        "type": "string",  
        "metadata": {  
            "description": "DNS for Load Balancer IP"  
        }  
    },  
    "vmNamePrefix": {  
        "type": "string",  
        "defaultValue": "myVM",  
        "metadata": {  
            "description": "Prefix to use for VM names"  
        }  
    },  
    "imagePublisher": {  
        "type": "string",  
        "defaultValue": "MicrosoftWindowsServer",  
        "metadata": {  
            "description": "Image Publisher"  
        }  
    },  
    "imageOffer": {  
        "type": "string",  
        "defaultValue": "WindowsServer",  
        "metadata": {  
            "description": "Image Offer"  
        }  
    },  
    "imageSKU": {  
        "type": "string",  
        "defaultValue": "2012-R2-Datacenter",  
        "allowedValues": [  
            "2008-R2-SP1",  
            "2012-Datacenter",  
            "2012-R2-Datacenter"  
        ],  
        "metadata": {  
            "description": "Image SKU"  
        }  
    },  
    "lbName": {  
        "type": "string".  
    }  
}
```

```

    "type": "string",
    "defaultValue": "myLB",
    "metadata": {
        "description": "Load Balancer name"
    }
},
"nicNamePrefix": {
    "type": "string",
    "defaultValue": "nic",
    "metadata": {
        "description": "Network Interface name prefix"
    }
},
"publicIPAddressName": {
    "type": "string",
    "defaultValue": "myPublicIP",
    "metadata": {
        "description": "Public IP Name"
    }
},
"vnetName": {
    "type": "string",
    "defaultValue": "myVNET",
    "metadata": {
        "description": "VNET name"
    }
},
"vmSize": {
    "type": "string",
    "defaultValue": "Standard_D1",
    "metadata": {
        "description": "Size of the VM"
    }
}
}
}

```

Variables

In the variables section, you can define values that are used in more than one place in your template, or values that are constructed from other expressions or variables. Variables are frequently used to simplify the syntax of your template.

```

"variables": {
    "availabilitySetName": "myAvSet",
    "subnetName": "Subnet-1",
    "vnetID": "[resourceId('Microsoft.Network/virtualNetworks',parameters('vnetName'))]",
    "subnetRef": "[concat(variables('vnetID'), '/subnets/', variables ('subnetName'))]",
    "publicIPAddressID": ""

[resourceId('Microsoft.Network/publicIPAddresses',parameters('publicIPAddressName'))]",
    "numberOfInstances": 2,
    "lbID": "[resourceId('Microsoft.Network/loadBalancers',parameters('lbName'))]",
    "frontEndIPConfigID": "[concat(variables('lbID'), '/frontendIPConfigurations/LoadBalancerFrontEnd')]",
    "lbPoolID": "[concat(variables('lbID'), '/backendAddressPools/BackendPool1')]",
    "lbProbeID": "[concat(variables('lbID'), '/probes/tcpProbe')]"
}

```

You have completed the template! You can compare your template against the full template in the [quickstart gallery](#) under [2 VMs with load balancer and load balancer rules template](#). Your template might be slightly different based on using different version numbers.

You can re-deploy the template by using the same commands you used when deploying the storage account. You do not need to delete the storage account before re-deploying because Resource Manager will skip re-creating resources that already exist and have not changed.

Next steps

- [Azure Resource Manager Template Visualizer](#) is a great tool to visualize Resource Manager templates, as they might become too large to understand just from reading the json file.
- To learn more about the structure of a template, see [Authoring Azure Resource Manager templates](#).
- To learn about deploying a template, see [Deploy a Resource Group with Azure Resource Manager template](#)
- For a four part series about automating deployment, see [Automating application deployments to Azure Virtual Machines](#). This series covers application architecture, access and security, availability and scale, and application deployment.

Automating application deployments to Azure Virtual Machines

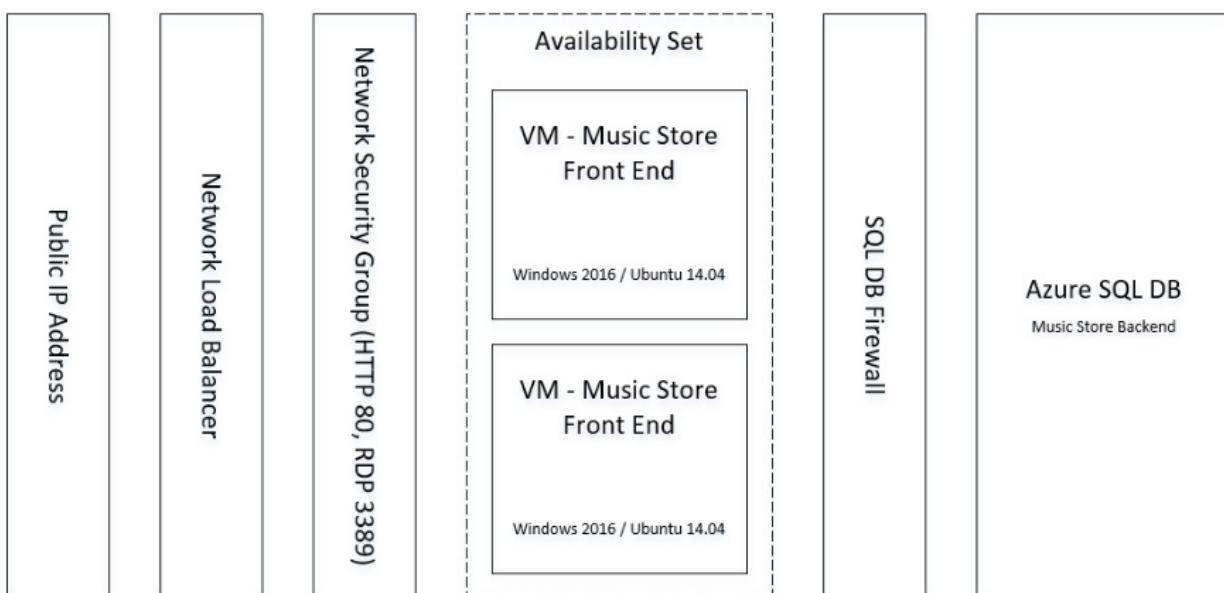
1/17/2017 • 2 min to read • [Edit on GitHub](#)

This four-part series details deploying and configuring Azure resource and applications using Azure Resource Manager templates. In this series, a sample template is deployed and the deployment template examined. The goal of this series is to educate on the relationship between Azure resources, and to provide hands on experience deploying fully integrated Azure Resource Manager templates. This document assumes a basic level of knowledge with Azure Resource Manager, before starting this tutorial familiarize yourself with basic Azure Resource Manager concepts.

Music store application

The sample used in this series is a .Net Core application simulating a Music Store shopping experience. This application can be deployed to either a Linux or Windows virtual system, sample deployments have been created for both. The application includes a web application and a SQL database. Before reading the articles in this series, deploy the application using the deployment button found on this page. When fully deployed, the application / Azure architecture looks like the following diagram.

The Music Store Resource Manager template can be found here, [Music Store Windows Template](#)



Each of these components, including the associate template JSON is examined in the following four articles.

- **Application Architecture** – Application components such as web sites and databases need to be hosted on Azure computer resources such as virtual machines and Azure SQL databases. This document walks through mapping compute need, to Azure resources, and deploying these resources with an Azure Resource Manager template.
- **Access and Security** – When hosting applications in Azure, it is necessary to consider how the application is accessed, and how different application components access each other. This document details providing and securing internet access to an application and access between application components.
- **Availability and Scale** – Availability and scale refer to the applications ability to stay running during infrastructure downtime, and the ability to scale compute resources to meet application demand. This document

details the components needed to deploy a load balanced and highly available application.

- **Application Deployment** - When deploying applications onto Azure Virtual Machines, the method by which the application binaries are installed on the Virtual Machine must be considered. This document details automating application installation using Azure Virtual Machine Custom Script Extensions.

The goal when developing Azure Resource Manager templates is to automate the deployment of Azure Infrastructure, and the installation and configuration of any applications being hosted on this Azure infrastructure. Working through these articles provides an example of this experience.

Deploy the music store application

The Music Store application can be deployed using this button.



The Azure Resource Manager template requires the following parameter values.

PARAMETER NAME	DESCRIPTION
ADMINUSERNAME	Admin user name that is used on the virtual machine and the Azure SQL Database.
ADMINPASSWORD	Password that is used on the Azure Virtual Machine and SQL Database.
NUMBEROFRINSTANCES	The number of virtual machines to be created. Each of these virtual machines host the Music Store web application, and all traffic is load balanced across them.
PUBLICIPADDRESSDNSNAME	Globally unique DNS name associated with the Public IP address.

When the template deployment has completed, browse to the public IP Address using any internet browser. The .Net Core Music site will be presented.

Next steps

[Step 1 - Application Architecture with Azure Resource Manager Templates](#)

[Step 2 - Access and Security in Azure Resource Manager Templates](#)

[Step 3 - Availability and Scale in Azure Resource Manager Templates](#)

[Step 4 - Application Deployment with Azure Resource Manager Templates](#)

Application architecture with Azure Resource Manager templates

1/17/2017 • 5 min to read • [Edit on GitHub](#)

When developing an Azure Resource Manager deployment, compute requirements need to be mapped to Azure resources and services. If an application consists of several http endpoints, a database, and a data caching service, the Azure resources that host each of these components needs to be rationalized. For instance, the sample Music Store application includes a web application that is hosted on a virtual machine, and a SQL database, which is hosted in Azure SQL database.

This document details how the Music Store compute resources are configured in the sample Azure Resource Manager template. All dependencies and unique configurations are highlighted. For the best experience, pre-deploy an instance of the solution to your Azure subscription and work along with the Azure Resource Manager template. The complete template can be found here – [Music Store Deployment on Windows](#).

Virtual Machine

The Music Store application includes a web application where customers can browse and purchase music. While there are several Azure services that can host web applications, for this example, a Virtual Machine is used. Using the sample Music Store template, a virtual machine is deployed, a web server install, and the Music Store website installed and configured. For the sake of this article, only the virtual machine deployment is detailed. The configuration of the web server and the application is detailed in a later article.

A virtual machine can be added to a template using the Visual Studio Add New Resource wizard, or by inserting valid JSON into the deployment template. When deploying a virtual machine, several related resources are also needed. If using Visual Studio to create the template, these resources are created for you. If manually constructing the template, these resources need to be inserted and configured.

Follow this link to see the JSON sample within the Resource Manager template – [Virtual Machine JSON](#).

```
{
  {
    "apiVersion": "2015-06-15",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[concat(variables('vmName'),copyindex())]",
    "location": "[resourceGroup().location]",
    "copy": [
      {
        "name": "virtualMachineLoop",
        "count": "[parameters('numberOfInstances')]"
      }
    ],
    "tags": {
      "displayName": "virtual-machine"
    },
    "dependsOn": [
      "[concat('Microsoft.Storage/storageAccounts/', variables('vhdStorageName'))]",
      "[concat('Microsoft.Compute/availabilitySets/', variables('availabilitySetName'))]",
      "nicLoop"
    ],
    "properties": {
      "availabilitySet": {
        "id": "[resourceId('Microsoft.Compute/availabilitySets', variables('availabilitySetName'))]"
      },
      .......<truncated>
    }
}
```

Once deployed, the virtual machine properties can be seen in the Azure portal.

Essentials ^	
Resource group win-music-store	Computer name MusicApp1
Status Running	Operating system Windows
Location West US	Size Standard D1 (1 core, 3.5 GB memory)
Subscription name Windows Azure MSDN - Visual Studio U...	Public IP address/DNS name label 23.99.9.45/winmusicstore.westus.cloudapp...
Subscription ID	Virtual network/subnet VirtualNetwork/MusicSubnet

Storage Account

Storage accounts have many storage options and capabilities. For the context of Azure Virtual machines, a storage account holds the virtual hard drives of the virtual machine and any additional data disks. The Music Store sample includes one storage account to hold the virtual hard drive of each virtual machine in the deployment.

Follow this link to see the JSON sample within the Resource Manager template – [Storage Account](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Storage/storageAccounts",
  "name": "[variables('vhdStorageName')]",
  "location": "[resourceGroup().location]",
  "tags": {
    "displayName": "storage-account"
  },
  "properties": {
    "accountType": "[variables('vhdStorageType')]"
  }
}
```

A storage account is associate with a virtual machine inside the Resource Manager template declaration of the virtual machine.

Follow this link to see the JSON sample within the Resource Manager template – [Virtual Machine and Storage Account association](#).

```
"osDisk": {
  "name": "osdisk",
  "vhd": {
    "uri": "[concat(reference(concat('Microsoft.Storage/storageAccounts/',variables('vhdStorageName'))), '2015-06-15').primaryEndpoints.blob,'vhds/osdisk', copyindex(), '.vhf'])"
  },
  "caching": "ReadWrite",
  "createOption": "FromImage"
}
```

After deployment, the storage account can be viewed in the Azure portal.

Essentials ^

Resource group **win-music-demo**

Status **Primary: Available**

Location **West US**

Subscription name **Windows Azure MSDN - Visual Studio U...**

Subscription ID

Performance **Standard**

Replication **Locally-redundant storage (LRS)**

Services

Blobs Files Tables Queues

Clicking into the storage account blob container, the virtual hard drive file for each virtual machine deployed with the template can be seen.

vhds

Container

Refresh Delete container Properties Access policy

Location: **vhds**

Search blobs by prefix (case-sensitive)

NAME	MODIFIED	BLOB TYPE	SIZE
MusicApp0.79e854f8-d2d1-477...	10/21/2016, 12:16:08 PM	Block blob	702 B
MusicApp1.743e099e-56d5-435...	10/21/2016, 12:16:07 PM	Block blob	702 B
osdisk0.vhd	10/21/2016, 12:16:11 PM	Page blob	136.37 GB
osdisk1.vhd	10/21/2016, 12:16:11 PM	Page blob	136.37 GB

For more information on Azure Storage, see [Azure Storage documentation](#).

Virtual Network

If a virtual machine requires internal networking such as the ability to communicate with other virtual machines and Azure resources, an Azure Virtual Network is required. A virtual network does not make the virtual machine accessible over the internet. Public connectivity requires a public IP address, which is detailed later in this series.

Follow this link to see the JSON sample within the Resource Manager template – [Virtual Network and Subnets](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/virtualNetworks",
  "name": "[variables('virtualNetworkName')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Network/networkSecurityGroups/', variables('networkSecurityGroup'))]"
  ],
  "tags": {
    "displayName": "virtual-network"
  },
  "properties": {
    "addressSpace": {
      "addressPrefixes": [
        "10.0.0.0/24"
      ]
    },
    "subnets": [
      {
        "name": "[variables('subnetName')]",
        "properties": {
          "addressPrefix": "10.0.0.0/24",
          "networkSecurityGroup": {
            "id": "[resourceId('Microsoft.Network/networkSecurityGroups', variables('networkSecurityGroup'))]"
          }
        }
      }
    ]
  }
}
```

From the Azure portal, the virtual network looks like the following image. Notice that all virtual machines deployed with the template are attached to the virtual network.

Essentials ▾

Resource group **win-music-store** Address space **10.0.0.0/24**
 Location **West US** DNS servers **Azure provided DNS service**
 Subscription name **Windows Azure MSDN - Visual Studio U...** **edit**
 Subscription ID

2 connected devices **edit**

DEVICE	TYPE	IP ADDRESS	SUBNET
NetworkInterface0	Network interface	10.0.0.4	MusicSubnet
NetworkInterface1	Network interface	10.0.0.5	MusicSubnet

Network Interface

A network interface connects a virtual machine to a virtual network, more specifically to a subnet that has been defined in the virtual network.

Follow this link to see the JSON sample within the Resource Manager template – [Network Interface](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/networkInterfaces",
  "name": "[concat(variables('networkInterfaceName'), copyIndex())]",
  "location": "[resourceGroup().location]",
  "tags": {
    "displayName": "network-interface"
  },
  "copy": {
    "name": "nicLoop",
    "count": "[parameters('numberOfInstances')]"
  },
  "dependsOn": [
    "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'))]",
    "[concat('Microsoft.Network/loadBalancers/', variables('loadBalancerName'))]",
    "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIpAddressName'))]",
    "[concat('Microsoft.Network/loadBalancers/', variables('loadBalancerName'), '/inboundNatRules/', 'RDP-VM',
copyIndex())]"
  ],
  "properties": {
    "ipConfigurations": [
      {
        "name": "ipconfig",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
          "subnet": {
            "id": "[variables('subnetRef')]"
          },
          "loadBalancerBackendAddressPools": [
            {
              "id": "[variables('lbPoolID')]"
            }
          ],
          "loadBalancerInboundNatRules": [
            {
              "id": "[concat(variables('lbID'), '/inboundNatRules/RDP-VM', copyIndex())]"
            }
          ]
        }
      }
    ]
  }
}
```

Each virtual machine resource includes a network profile. The network interface is associated with the virtual machine in this profile.

Follow this link to see the JSON sample within the Resource Manager template – [Virtual Machine Network Profile](#).

```
"networkProfile": {
  "networkInterfaces": [
    {
      "id": "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('networkInterfaceName'),
copyIndex()))]"
    }
  ]
}
```

From the Azure portal, the network interface looks like the following image. The internal IP address and the virtual machine association can be seen on the network interface resource.

Essentials ^

Resource group win-music-store	Private IP address 10.0.0.4
Location West US	Virtual network/subnet VirtualNetwork/MusicSubnet
Subscription name Windows Azure MSDN - Visual Studio U...	Public IP address -
Subscription ID	Network security group -
	Attached to MusicApp0

For more information on Azure Virtual Networks, see [Azure Virtual Network documentation](#).

Azure SQL Database

In addition to a virtual machine hosting the Music Store website, an Azure SQL Database is deployed to host the Music Store database. The advantage of using Azure SQL Database here is that a second set of virtual machines is not required, and scale and availability is built into the service.

An Azure SQL database can be added using the Visual Studio Add New Resource wizard, or by inserting valid JSON into a template. The SQL Server resource includes a user name and password that is granted administrative rights on the SQL instance. Also, a SQL firewall resource is added. By default, applications hosted in Azure are able to connect with the SQL instance. To allow external application such a SQL Server Management studio to connect to the SQL instance, the firewall needs to be configured. For the sake of the Music Store demo, the default configuration is fine.

Follow this link to see the JSON sample within the Resource Manager template – [Azure SQL DB](#).

```
{  
  "apiVersion": "2014-04-01-preview",  
  "type": "Microsoft.Sql/servers",  
  "name": "[variables('musicstoresqlName')]",  
  "location": "[resourceGroup().location]",  
  "dependsOn": [],  
  "tags": {  
    "displayName": "sql-music-store"  
  },  
  "properties": {  
    "administratorLogin": "[parameters('adminUsername')]",  
    "administratorLoginPassword": "[parameters('adminPassword')]"  
  },  
  "resources": [  
    {  
      "apiVersion": "2014-04-01-preview",  
      "type": "firewallrules",  
      "name": "firewall-allow-azure",  
      "location": "[resourceGroup().location]",  
      "dependsOn": [  
        "[concat('Microsoft.Sql/servers/', variables('musicstoresqlName'))]"  
      ],  
      "properties": {  
        "startIpAddress": "0.0.0.0",  
        "endIpAddress": "0.0.0.0"  
      }  
    }  
  ]  
}
```

A view of the SQL server and MusicStore database as seen in the Azure portal.

Essentials ^

Resource group win-music-demo	Server version V12
Status Available	Auditing Not configured
Location West US	Server admin neillocal
Subscription name Windows Azure MSDN - Visual Studio U...	Active Directory admin Not configured
Subscription ID	Firewall Show firewall settings

Databases

SQL databases		
1 Database	SQL V12	
<input type="checkbox"/> DATABASE	STATUS	PRICING TIER
MusicStore	Online	Standard: S0

For more information on deploying Azure SQL Database, see [Azure SQL Database documentation](#).

Next step

[Step 2 - Access and Security in Azure Resource Manager Templates](#)

Access and security in Azure Resource Manager templates

1/17/2017 • 3 min to read • [Edit on GitHub](#)

Applications hosted in Azure likely need to be accessed over the internet or a VPN / Express Route connection with Azure. With the Music Store application sample, the web site is made available on the internet with a public IP address. With access established, connections to the application and access to the virtual machine resources themselves should be secured. This access security is provided with a Network Security Group.

This document details how the Music Store application is secured in the sample Azure Resource Manager template. All dependencies and unique configurations are highlighted. For the best experience, pre-deploy an instance of the solution to your Azure subscription and work along with the Azure Resource Manager template. The complete template can be found here – [Music Store Deployment on Windows](#).

Public IP Address

To provide public access to an Azure resource, a public IP address resource can be used. Public IP address can be configured with a static or dynamic IP address. If a dynamic address is used, and the virtual machine is stopped and deallocated, the address is removed. When the machine is started again, it may be assigned a different public IP address. To prevent an IP address from changing, a reserved IP address can be used.

A Public IP Address can be added to an Azure Resource Manager template using the Visual Studio Add New Resource Wizard, or by inserting valid JSON into a template.

Follow this link to see the JSON sample within the Resource Manager template – [Public IP Address](#).

```
{  
  "apiVersion": "2015-06-15",  
  "type": "Microsoft.Network/publicIPAddresses",  
  "name": "[variables('publicIpAddressName')]",  
  "location": "[resourceGroup().location]",  
  "dependsOn": [],  
  "tags": {  
    "displayName": "public-ip"  
  },  
  "properties": {  
    "publicIPAllocationMethod": "Dynamic",  
    "dnsSettings": {  
      "domainNameLabel": "[parameters('publicipaddressDnsName')]"  
    }  
  }  
}
```

A Public IP Address can be associated with a Virtual Network Adapter, or a Load Balancer. In this example, because the Music Store website is load balanced across several virtual machines, the Public IP Address is attached to the Load Balancer.

Follow this link to see the JSON sample within the Resource Manager template – [Public IP Address association with Load Balancer](#).

```
"frontendIPConfigurations": [
  {
    "properties": {
      "publicIPAddress": {
        "id": "[resourceId('Microsoft.Network/publicIPAddresses', variables('publicIpAddressName'))]"
      }
    },
    "name": "LoadBalancerFrontend"
  }
]
```

The public IP Address as seen from the Azure portal. Notice that the public IP address is associated to a load balancer and not a virtual machine. Network load balancers are detailed in the next document of this series.

The screenshot shows the 'Essentials' section of the Azure portal for a Public IP Address. The resource group is 'win-music-demo', located in 'West US'. The IP address is 40.118.175.81, and the DNS name is winmusicdemo.westus.cloudapp.azure.com. It is associated with a 'LoadBalancer'. The subscription name is 'Windows Azure MSDN - Visual Studio U...', and the subscription ID is partially visible as 'Windows Azure MSDN - Visual Studio U...'. There are edit icons next to the IP address and DNS name.

For more information on Azure Public IP Addresses, see [IP addresses in Azure](#).

Network Security Group

Once access has been established to Azure resources, this access should be limited. For Azure virtual machines, secure access is accomplished using a network security group. With the Music Store application sample, all access to the virtual machine is restricted except for over port 80 for http access, and port 3389 for RDP access. A Network Security Group can be added to an Azure Resource Manager template using the Visual Studio Add New Resource Wizard, or by inserting valid JSON into a template.

Follow this link to see the JSON sample within the Resource Manager template – [Network Security Group](#).

```
{
  "apiVersion": "2016-03-30",
  "type": "Microsoft.Network/networkSecurityGroups",
  "name": "[variables('networkSecurityGroup')]",
  "location": "[resourceGroup().location]",
  "tags": {
    "displayName": "network-security-group"
  },
  "properties": {
    "securityRules": [
      {
        "name": "http",
        "properties": {
          "description": "http endpoint",
          "protocol": "Tcp",
          "sourcePortRange": "*",
          "destinationPortRange": "80",
          "sourceAddressPrefix": "*",
          "destinationAddressPrefix": "*",
          "access": "Allow",
          "priority": 100,
          "direction": "Inbound"
        }
      },
      .......<truncated>
    ]
  }
},
```

In this example, the network security group is associate with the subnet object declared in the Virtual Network resource.

Follow this link to see the JSON sample within the Resource Manager template – [Network Security Group association with Virtual Network](#).

```
"subnets": [
  {
    "name": "[variables('subnetName')]",
    "properties": {
      "addressPrefix": "10.0.0.0/24",
      "networkSecurityGroup": {
        "id": "[resourceId('Microsoft.Network/networkSecurityGroups', variables('networkSecurityGroup'))]"
      }
    }
  }
]
```

Here is what the network security group looks like from the Azure portal. Notice that an NSG can be associate with a subnet and / or network interface. In this case, the NSG is associated to a subnet. In this configuration, the inbound rules apply to all virtual machines connected to the subnet.

Essentials ^

Resource group **win-music-demo** Security rules **2 inbound, 0 outbound**
Location **West US** Associated with **1 subnets, 0 network interfaces**

Subscription name **Windows Azure MSDN - Visual Studio U...**

Subscription ID

2 Inbound security rules

PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
100	http	Any	Any	HTTP (TCP/80)	Allow
200	rdp	Any	Any	RDP (TCP/3389)	Allow

0 Outbound security rules

PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
No results.					

For in-depth information on Network Security Groups, see [What is a Network Security Group](#).

Next step

[Step 3 - Availability and Scale in Azure Resource Manager Templates](#)

Availability and scale in Azure Resource Manager templates

1/17/2017 • 6 min to read • [Edit on GitHub](#)

Availability and scale refer to uptime and the ability to meet demand. If an application must be up 99.9% of the time, it needs to have an architecture that allows for multiple concurrent compute resources. For instance, rather than having a single website, a configuration with a higher level of availability includes multiple instances of the same site, with balancing technology in front of them. In this configuration, one instance of the application can be taken down for maintenance, while the remaining continue to function. Scale on the other hand refers to an applications ability to serve demand. With a load balanced application, adding or removing instances from the pool allows an application to scale to meet demand.

This document details how the Music Store sample deployment is configured for availability and scale. All dependencies and unique configurations are highlighted. For the best experience, pre-deploy an instance of the solution to your Azure subscription and work along with the Azure Resource Manager template. The complete template can be found here – [Music Store Deployment on Windows](#).

Availability Set

An Availability Set logically spans Azure Virtual Machines across physical hosts and other infrastructural components such as power supplies and physical networking hardware. Availability sets ensure that during maintenance, device failure, or other down time, not all virtual machines are effected. An Availability Set can be added to an Azure Resource Manager template using the Visual Studio Add New Resource Wizard, or inserting valid JSON into a template.

Follow this link to see the JSON sample within the Resource Manager template – [Availability Set](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Compute/availabilitySets",
  "name": "[variables('availabilitySetName')]",
  "location": "[resourceGroup().location]",
  "tags": {
    "displayName": "availability-set"
  },
  "properties": {}
}
```

An Availability Set is declared as a property of a Virtual Machine resource.

Follow this link to see the JSON sample within the Resource Manager template – [Availability Set association with Virtual Machine](#).

```
"properties": {
  "availabilitySet": {
    "id": "[resourceId('Microsoft.Compute/availabilitySets', variables('availabilitySetName'))]"
  }
}
```

The availability set as seen from the Azure portal. Each virtual machine and details about the configuration are detailed here.

Essentials ^

Resource group win-music-demo	Fault domains 3
Location West US	Update domains 5
Subscription name Windows Azure MSDN - Visual Studio Ulti...	Virtual machines 2
Subscription ID	

Virtual machines

NAME	STATUS	FAULT DOMAIN	UPDATE DOMAIN
MusicApp0	Running	0	0
MusicApp1	Running	1	1

For in-depth information on Availability Sets, see [Manage availability of virtual machines](#).

Network Load Balancer

Whereas an availability set provides application fault tolerance, a load balancer makes many instances of the application available on a single network address. Multiple instances of an application can be hosted on many virtual machines, each one connected to a load balancer. As the application is accessed, the load balancer routes the incoming request across the attached members. A Load Balancer can be added using the Visual Studio Add New Resource Wizard, or by inserting properly formatted JSON resource into the Azure Resource Manager template.

Follow this link to see the JSON sample within the Resource Manager template – [Network Load Balancer](#).

```
{  
  "apiVersion": "2015-06-15",  
  "type": "Microsoft.Network/loadBalancers",  
  "name": "[variables('loadBalancerName')]",  
  "location": "[resourceGroup().location]",  
  "tags": {  
    "displayName": "load-balancer"  
  },  
  .......<truncated>  
}
```

Because the sample application is exposed to the internet with a public IP address, this address is associated with the load balancer.

Follow this link to see the JSON sample within the Resource Manager template – [Network Load Balancer association with Public IP Address](#).

```

"frontendIPConfigurations": [
  {
    "properties": {
      "publicIPAddress": {
        "id": "[resourceId('Microsoft.Network/publicIPAddresses', variables('publicIpAddressName'))]"
      }
    },
    "name": "LoadBalancerFrontend"
  }
]

```

From the Azure portal, the network load balancer overview shows the association with the public IP address.

Load Balancer Rule

When using a load balancer, rules are configured that control how traffic is balanced across the intended resources. With the sample Music Store application, traffic arrives on port 80 of the public IP address and is distributed across port 80 of all virtual machines.

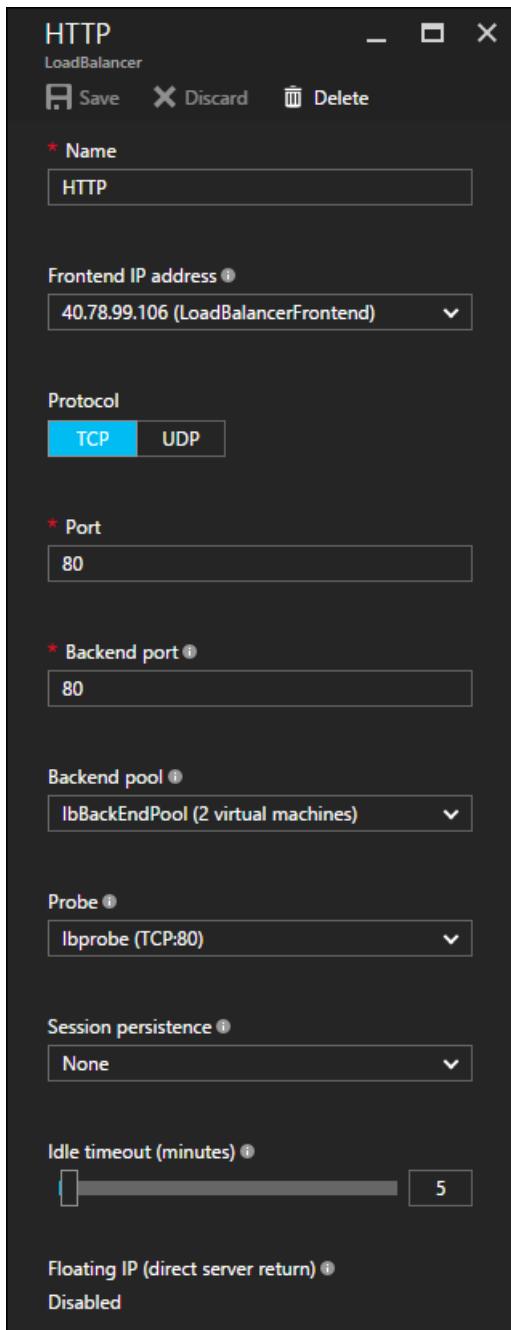
Follow this link to see the JSON sample within the Resource Manager template – [Load Balancer Rule](#).

```

"loadBalancingRules": [
  {
    "name": "[variables('loadBalencerRule')]",
    "properties": {
      "frontendIPConfiguration": {
        "id": "[concat(resourceId('Microsoft.Network/loadBalancers', variables('loadBalancerName')), '/frontendIPConfigurations/LoadBalancerFrontend')]"
      },
      "backendAddressPool": {
        "id": "[variables('lbPoolID')]"
      },
      "protocol": "Tcp",
      "frontendPort": 80,
      "backendPort": 80,
      "enableFloatingIP": false,
      "idleTimeoutInMinutes": 5,
      "probe": {
        "id": "[variables('lbProbeID')]"
      }
    }
  }
]

```

A view of the network load balancer rule from the portal.



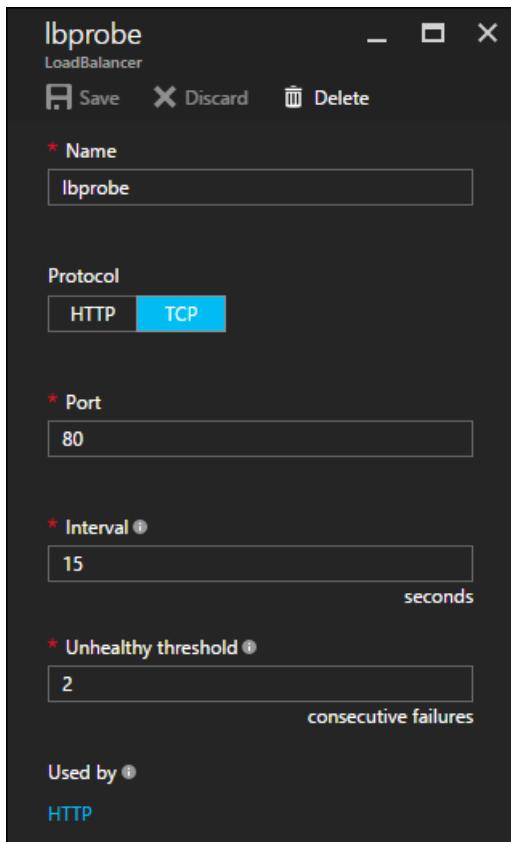
Load Balancer Probe

The load balancer also needs to monitor each virtual machine so that requests are served only to running systems. This monitoring takes place by constant probing of a pre-defined port. The Music Store deployment is configured to probe port 80 on all included virtual machines.

Follow this link to see the JSON sample within the Resource Manager template – [Load Balancer Probe](#).

```
"probes": [
  {
    "properties": {
      "protocol": "Tcp",
      "port": 80,
      "intervalInSeconds": 15,
      "numberOfProbes": 2
    },
    "name": "lprobe"
  }
]
```

The load balancer probe seen from the Azure portal.



Inbound NAT Rules

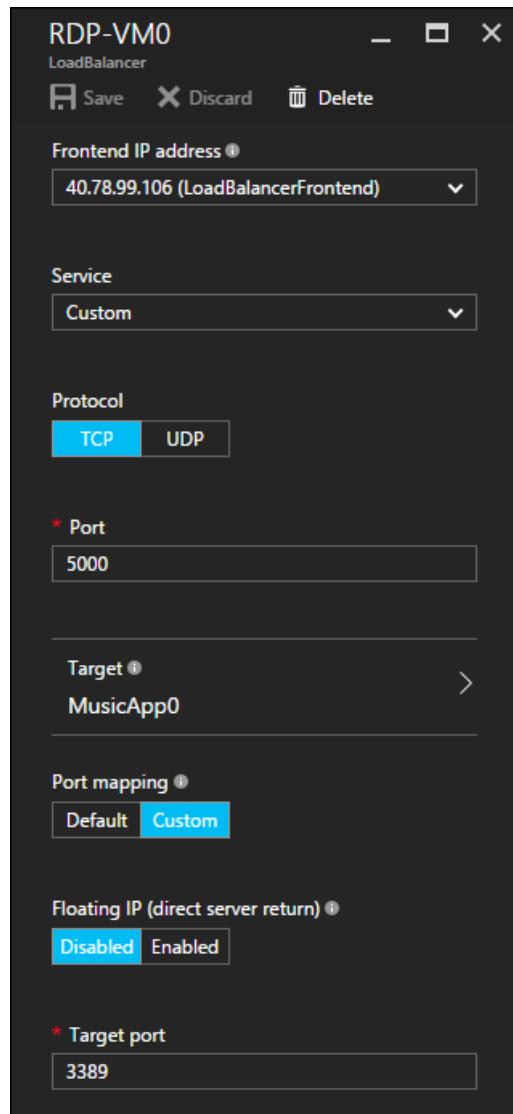
When using a Load Balancer, rules need to be put into place that provide non-load balanced access to each Virtual Machine. For instance, when creating an RDP connection with each virtual machine, this traffic should not be load balanced, rather a pre-determined path should be configured. Pre-determined paths are configured using an Inbound NAT Rule resource. Using this resource, inbound communication can be mapped to individual Virtual Machines.

With the Music Store application, a port starting at 5000 is mapped to port 3389 on each Virtual Machine for RDP access. The `copyindex()` function is used to increment the incoming port, such that the second Virtual Machine receives an incoming port of 5001, the third 5002, and so on.

Follow this link to see the JSON sample within the Resource Manager template – [Inbound NAT Rules](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/loadBalancers/inboundNatRules",
  "name": "[concat(variables('loadBalancerName'), '/', 'RDP-VM', copyIndex())]",
  "location": "[resourceGroup().location]",
  "tags": {
    "displayName": "load-balancer-nat-rule"
  },
  "copy": {
    "name": "lbNatLoop",
    "count": "[parameters('numberOfInstances')]"
  },
  "dependsOn": [
    "[concat('Microsoft.Network/loadBalancers/', variables('loadBalancerName'))]"
  ],
  "properties": {
    "frontendIPConfiguration": {
      "id": "[variables('ipConfigID')]"
    },
    "protocol": "tcp",
    "frontendPort": "[copyIndex(5000)]",
    "backendPort": 3389,
    "enableFloatingIP": false
  }
}
}
```

One example inbound NAT rule as seen in the Azure portal. An RDP NAT rule is created for each virtual machine in the deployment.



For in-depth information on the Azure Network Load Balancer, see [Load balancing for Azure infrastructure services](#).

Deploy multiple VMs

Finally, for an Availability Set or Load Balancer to effectively function, multiple virtual machines are required. Multiple VMs can be deployed using the Azure Resource Manager template copy function. Using the copy function, it is not necessary to define a finite number of Virtual Machines, rather this value can be dynamically provided at the time of deployment. The copy function consumes the number of instances to be created and handles deploying the proper number of virtual machines and associated resources.

In the Music Store Sample template, a parameter is defined that takes in an instance count. This number is used throughout the template when creating virtual machines and related resources.

```
"numberOfInstances": {  
    "type": "int",  
    "minValue": 1,  
    "defaultValue": 2,  
    "metadata": {  
        "description": "Number of VM instances to be created behind load balancer."  
    }  
},
```

On the Virtual Machine resource, the copy loop is given a name and the number of instances parameter used to control the number of resulting copies.

Follow this link to see the JSON sample within the Resource Manager template – [Virtual Machine Copy Function](#).

```
{  
    "apiVersion": "2015-06-15",  
    "type": "Microsoft.Compute/virtualMachines",  
    "name": "[concat(variables('vmName'),copyindex())]",  
    "location": "[resourceGroup().location]",  
    "copy": {  
        "name": "virtualMachineLoop",  
        "count": "[parameters('numberOfInstances')]"  
    }  
}
```

The current iteration of the copy function can be accessed with the `copyIndex()` function. The value of the copy index function can be used to name virtual machines and other resources. For instance, if two instances of a virtual machine are deployed, they need different names. The `copyIndex()` function can be used as part of the virtual machine name to create a unique name. An example of the `copyIndex()` function used for naming purposes can be seen in the Virtual Machine resource. Here, the computer name is a concatenation of the `vmName` parameter, and the `copyIndex()` function.

Follow this link to see the JSON sample within the Resource Manager template – [Copy Index Function](#).

```
"osProfile": {  
    "computerName": "[concat(variables('vmName'),copyindex())]",  
    "adminUsername": "[parameters('adminUsername')]",  
    "adminPassword": "[parameters('adminPassword')]"  
}
```

The `copyIndex` function is used several times in the Music Store sample template. Resources and functions utilizing `copyIndex` include anything specific to a single instance of the virtual machine such as network interface, load balancer rules, and any depends on functions.

For more information on the copy function, see [Create multiple instances of resources in Azure Resource Manager](#).

Next step

[Step 4 - Application Deployment with Azure Resource Manager Templates](#)

Application deployment with Azure Resource Manager templates

1/17/2017 • 3 min to read • [Edit on GitHub](#)

Once all Azure infrastructural requirements have been identified and translated into a deployment template, the actual application deployment needs to be addressed. Application deployment here is referring to installing the actual application binaries onto Azure resources. For the Music Store sample, .Net Core and IIS need to be installed and configured on each virtual machine. The Music Store binaries need to be installed onto the virtual machine, and the Music Store database pre-created.

This document details how Virtual Machine extensions can automate application deployment and configuration to Azure virtual machines. All dependencies and unique configurations are highlighted. For the best experience, pre-deploy an instance of the solution to your Azure subscription and work along with the Azure Resource Manager template. The complete template can be found here – [Music Store Deployment on Windows](#).

Configuration script

Virtual Machine extensions are specialized programs that execute against virtual machines to provide configuration automation. Extensions are available for many specific purposes such as anti-virus, logging configuration, and Docker configuration. The Custom Script extension can be used to run any script against a virtual machine. With the Music Store sample, it is up to the custom script extension to configure the Windows virtual machines and install the Music Store application.

Before detailing how virtual machine extensions are declared in an Azure Resource Manager template, examine the script that is run. This script configures the Windows virtual machine to host the Music Store application. When run, the script installs all needed software, install the Music store application from source control, and prepare the database.

This sample is for demonstration purposes.

```

<#
    .SYNOPSIS
        Downloads and configures .Net Core Music Store application sample across IIS and Azure SQL DB.
#>

Param (
    [string]$user,
    [string]$password,
    [string]$sqlserver
)

# Firewall
netsh advfirewall firewall add rule name="http" dir=in action=allow protocol=TCP localport=80

# Folders
New-Item -ItemType Directory c:\temp
New-Item -ItemType Directory c:\music

# Install iis
Install-WindowsFeature web-server -IncludeManagementTools

# Install dot.net core sdk
Invoke-WebRequest http://go.microsoft.com/fwlink/?LinkId=615460 -outfile c:\temp\vc_redistx64.exe
Start-Process c:\temp\vc_redistx64.exe -ArgumentList '/quiet' -Wait
Invoke-WebRequest https://go.microsoft.com/fwlink/?LinkId=809122 -outfile c:\temp\DotNetCore.1.0.0-SDK.Preview2-x64.exe
Start-Process c:\temp\DotNetCore.1.0.0-SDK.Preview2-x64.exe -ArgumentList '/quiet' -Wait
Invoke-WebRequest https://go.microsoft.com/fwlink/?LinkId=817246 -outfile c:\temp\DotNetCore.WindowsHosting.exe
Start-Process c:\temp\DotNetCore.WindowsHosting.exe -ArgumentList '/quiet' -Wait

# Download music app
Invoke-WebRequest https://github.com/Microsoft/dotnet-core-sample-templates/raw/master/dotnet-core-music-windows/music-app/music-store-azure-demo-pub.zip -OutFile c:\temp\musicstore.zip
Expand-Archive C:\temp\musicstore.zip c:\music

# Azure SQL connection string in environment variable
[Environment]::SetEnvironmentVariable("Data:DefaultConnectionString", "Server=$sqlserver;Database=MusicStore;Integrated Security=False;User Id=$user;Password=$password;MultipleActiveResultSets=True;Connect Timeout=30",
[EnvironmentVariableTarget]::Machine)

# Pre-create database
$env:Data:DefaultConnectionString = "Server=$sqlserver;Database=MusicStore;Integrated Security=False;User Id=$user;Password=$password;MultipleActiveResultSets=True;Connect Timeout=30"
Start-Process 'C:\Program Files\dotnet\dotnet.exe' -ArgumentList 'c:\music\MusicStore.dll'

# Configure iis
Remove-WebSite -Name "Default Web Site"
Set-ItemProperty IIS:\AppPools\DefaultAppPool\ managedRuntimeVersion ""
New-Website -Name "MusicStore" -Port 80 -PhysicalPath C:\music\ -ApplicationPool DefaultAppPool
& iisreset

```

VM Script Extension

VM Extensions can be run against a virtual machine at build time by including the extension resource in the Azure Resource Manager template. The extension can be added with the Visual Studio Add Resource wizard, or by inserting valid JSON into the template. The Script Extension resource is nested inside the Virtual Machine resource; this can be seen in the following example.

Follow this link to see the JSON sample within the Resource Manager template – [VM Script Extension](#).

Notice in the below JSON that the script is stored in GitHub. This script could also be stored in Azure Blob storage. Also, Azure Resource Manager templates allow the script execution string to be constructed such that template parameters values can be used as parameters for script execution. In this case data is provided when deploying the

templates, and these values can then be used when executing the script.

```
{  
    "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 on using the custom script extension, see [Custom script extensions with Resource Manager templates](#).

Next Step

[Explore More Azure Resource Manager Templates](#)

How to attach a data disk to a Windows VM in the Azure portal

1/17/2017 • 3 min to read • [Edit on GitHub](#)

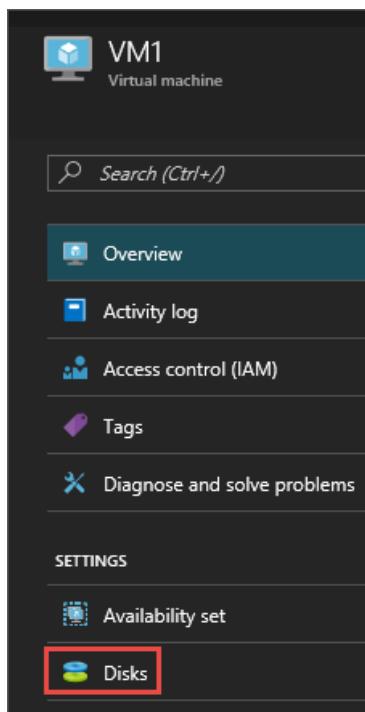
This article shows you how to attach both new and existing disks to a Windows virtual machine through the Azure portal. You can also [attach a data disk to a Linux VM in the Azure portal](#). Before you do this, review these tips:

- The size of the virtual machine controls how many data disks you can attach. For details, see [Sizes for virtual machines](#).
- To use Premium storage, you'll need a DS-series or GS-series virtual machine. You can use disks from both Premium and Standard storage accounts with these virtual machines. Premium storage is available in certain regions. For details, see [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#).
- For a new disk, you don't need to create it first because Azure creates it when you attach it.
- For an existing disk, the .vhd file must be available in an Azure storage account. You can use a .vhd that's already there, if it's not attached to another virtual machine, or upload your own .vhd file to the storage account.

You can also [attach a data disk using Powershell](#).

Find the virtual machine

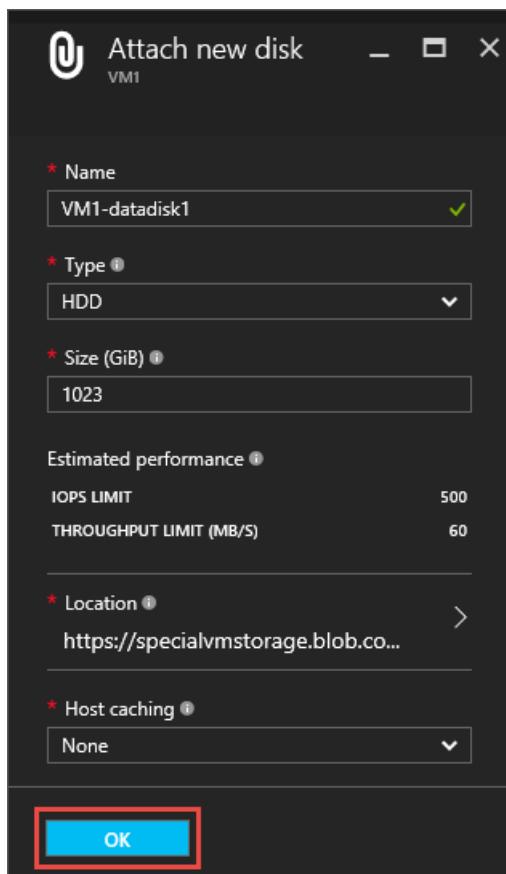
1. Sign in to the [Azure portal](#).
2. On the Hub menu, click **Virtual Machines**.
3. Select the virtual machine from the list.
4. To the Virtual machines blade, in **Essentials**, click **Disk**s.



Continue by following instructions for attaching either a [new disk](#) or an [existing disk](#).

Option 1: Attach and initialize a new disk

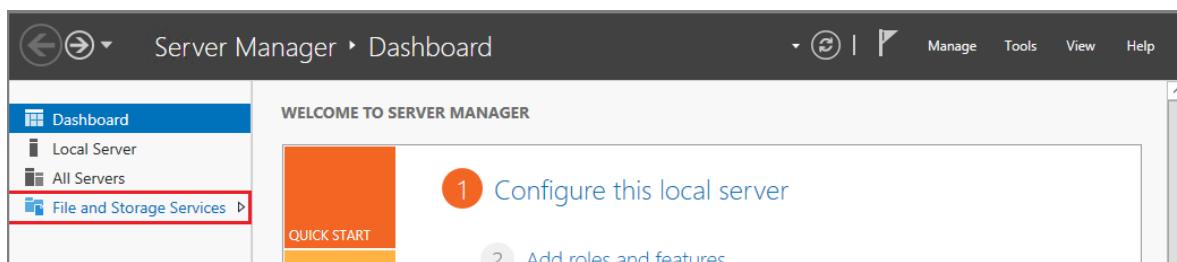
1. On the **Disks** blade, click **Attach new**.
2. Review the default settings, update as necessary, and then click **OK**.



3. 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 virtual machine. For instructions, see [How to connect and log on to an Azure virtual machine running Windows](#).
2. After you log on to the virtual machine, open **Server Manager**. In the left pane, select **File and Storage Services**.

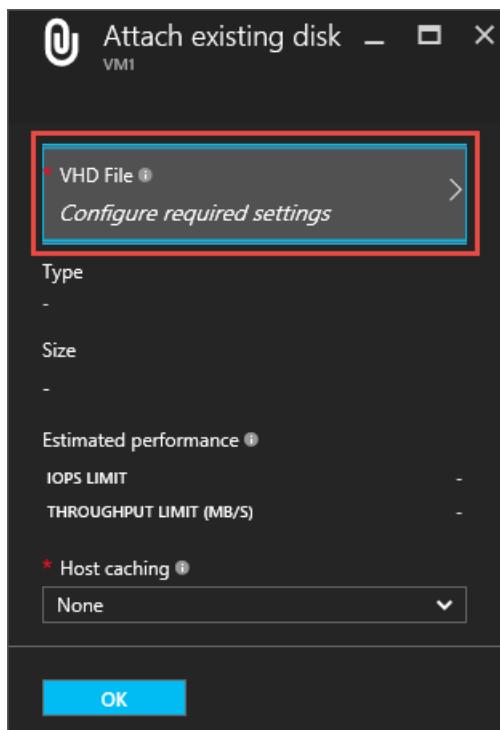


3. Expand the menu and select **Disks**.
4. The **Disks** section lists the disks. In most cases, it will have disk 0, disk 1, and disk 2. Disk 0 is the operating system disk, disk 1 is the temporary disk, and disk 2 is the data disk you just attached to the VM. The new data disk will list the Partition as **Unknown**. Right-click the disk and select **Initialize**.
5. You're notified that all data will be erased when the disk is initialized. Click **Yes** to acknowledge the warning and initialize the disk. Once complete, the partition will be listed as **GPT**. Right-click the disk again and select **New Volume**.
6. Complete the wizard using the default values. When the wizard is done, the **Volumes** section lists the new volume. The disk is now online and ready to store data.

The screenshot shows the 'File and Storage Services' interface. In the left navigation pane, 'Disks' is selected. The main area displays three sections: 'DISKS' (listing three disks: 1, 0, and 2), 'VOLUMES' (listing one volume: E), and 'STORAGE POOL' (noting 'No related storage pool exists'). The 'E' volume is highlighted with a red box.

Option 2: Attach an existing disk

1. On the **Disks** blade, click **Attach existing**.
2. Under **Attach existing disk**, click **VHD File**.



3. Under **Storage accounts**, select the account and container that holds the .vhdx file.

The 'Storage accounts' blade shows one account: 'specialvmstorage'. The 'Containers' blade shows one container: 'vhds'. Both items are highlighted with red boxes.

4. Select the .vhdx file.
5. Under **Attach existing disk**, the file you just selected is listed under **VHD File**. Click **OK**.

6. After Azure attaches the disk to the virtual machine, it's listed in the virtual machine's disk settings under **Data Disks**.

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
```

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

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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.

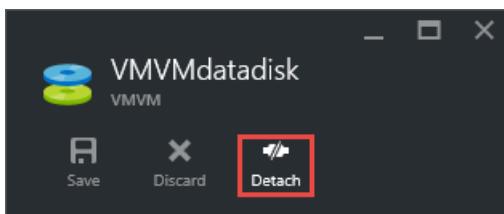
WARNING

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 portal hub, select **Virtual Machines**.
2. Select the virtual machine that has the data disk you want to detach and then click **All settings**.
3. In the **Settings** blade, select **Disks**.
4. In the **Disks** blade, select the data disk that you would like to detach.
5. In the blade for the data disk, click **Detach**.



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. The command stores the virtual machine in the **\$VirtualMachine** variable.

The second command removes the data disk named **DataDisk3** from the virtual machine.

The final command updates the state of the virtual machine 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" -Name "MyVM07" -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

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 127 GB. 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 1023 GB.

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](#).

Use the D: drive as a data drive on a Windows VM

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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](#)

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.

Attach the data disk

First, you'll need to attach the data disk to the virtual machine.

- To use the portal, see [How to attach a data disk in the Azure portal](#)
- To use the classic portal, see [How to attach a data disk to a Windows virtual machine](#).

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 drive **G** 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**.
7. Right-click on **G**, the Temporary Storage drive, and select **Change Drive Letter and Paths**.
8. Under Drive letter, select drive **E** and then click **OK**.

NOTE

If your VM has other disks or drives, use the same method to reassign the drive letters of the other disks and drives. You want the disk configuration to be:

- C: OS disk
- D: Data Disk
- E: Temporary disk

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 **E** 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](#).

Azure Disk Encryption for Windows and Linux IaaS VMs

1/17/2017 • 43 min to read • [Edit on GitHub](#)

Microsoft Azure is strongly committed to ensuring your data privacy, data sovereignty and enables you to control your Azure hosted data through a range of advanced technologies to encrypt, control and manage encryption keys, control & audit access of data. This provides Azure customers the flexibility to choose the solution that best meets their business needs. In this paper, we will introduce you to a new technology solution "Azure Disk Encryption for Windows and Linux IaaS VM's" to help protect and safeguard your data to meet your organizational security and compliance commitments. The paper provides detailed guidance on how to use the Azure disk encryption features including the supported scenarios and the user experiences.

NOTE

Certain recommendations contained herein may result in increased data, network, or compute resource usage resulting in additional license or subscription costs.

Overview

Azure Disk Encryption is a new capability that lets you encrypt your Windows and Linux IaaS virtual machine disks. Azure Disk Encryption leverages the industry standard [BitLocker](#) feature of Windows and the [DM-Crypt](#) feature of Linux to provide volume encryption for the OS and the data disks. The solution is integrated with [Azure Key Vault](#) to help you control and manage the disk encryption keys and secrets in your key vault subscription, while ensuring that all data in the virtual machine disks are encrypted at rest in your Azure storage.

Azure disk encryption for Windows and Linux IaaS VMs is now in **General Availability** in all Azure public regions for Standard VMs and VMs with premium storage.

Encryption Scenarios

The Azure Disk Encryption solution supports the following customer scenarios:

- Enable encryption on new IaaS VMs created from pre-encrypted VHD and encryption keys
- Enable encryption on new IaaS VMs created from the Azure Gallery images
- Enable encryption on existing IaaS VMs running in Azure
- Disable encryption on Windows IaaS VMs
- Disable encryption on data drives for Linux IaaS VMs

The solution supports the following for IaaS VMs when enabled in Microsoft Azure:

- Integration with Azure Key Vault
- Standard tier VMs - [A](#), [D](#), [DS](#), [G](#), [GS](#) etc series IaaS VMs
- Enable encryption on Windows and Linux IaaS VMs
- Disable encryption on OS and data drives for Windows IaaS VMs
- Disable encryption on data drives for Linux IaaS VMs
- Enable encryption on IaaS VMs running Windows Client OS
- Enable encryption on volumes with mount paths
- Enable encryption on Linux VMs configured with disk striping (RAID) using mdadm.
- Enable encryption on Linux VMs using LVM for data disks.

- Enable encryption on Windows VMs configured with Storage Spaces
- All Azure public regions are supported

The solution does not support the following scenarios, features and technology in the release:

- Basic tier IaaS VMs
- Disable encryption on OS drive for Linux IaaS VMs
- IaaS VMs created using classic VM creation method
- Integration with your on-premises Key Management Service
- Azure Files (Azure file share), Network file system (NFS), Dynamic volumes, Windows VMs configured with Software-based RAID systems

Encryption Features

When you enable and deploy Azure disk encryption for Azure IaaS VMs, the following capabilities are enabled, depending on the configuration provided:

- Encryption of OS volume to protect boot volume at rest in customer storage
- Encryption of Data volume/s to protect the data volumes at rest in customer storage
- Disable encryption on OS and data drives for Windows IaaS VMs
- Disable encryption on data drives for Linux IaaS VMs
- Safeguarding the encryption keys and secrets in customer Azure key vault subscription
- Reporting encryption status of the encrypted IaaS VM
- Removal of disk encryption configuration settings from the IaaS virtual machine
- Backup and Restore of encrypted VMs using Azure backup service

NOTE

Backup and restore of encrypted VMs is supported only for VMs encrypted using Key Encryption Key [KEK] configuration. It is not supported for VMs encrypted without KEK. KEK is an optional parameter to enable VM encryption.

The Azure disk encryption for IaaS VMS for Windows and Linux solution includes the disk encryption extension for Windows, disk encryption extension for Linux, disk encryption PowerShell cmdlets, disk encryption CLI cmdlets and disk encryption Azure Resource Manager templates. The Azure disk encryption solution is supported on IaaS VMs running Windows or Linux OS. For more details on the supported Operating Systems, see prerequisites section below.

NOTE

There is no additional charge for encrypting VM disks with Azure Disk Encryption.

Value Proposition

The Azure Disk Encryption Management solution enables the following business needs in the cloud:

- IaaS VM's are secured at rest using industry standard encryption technology to address organizational security and compliance requirements.
- IaaS VM's boot under customer controlled keys and policies, and they can audit their usage in Key Vault.

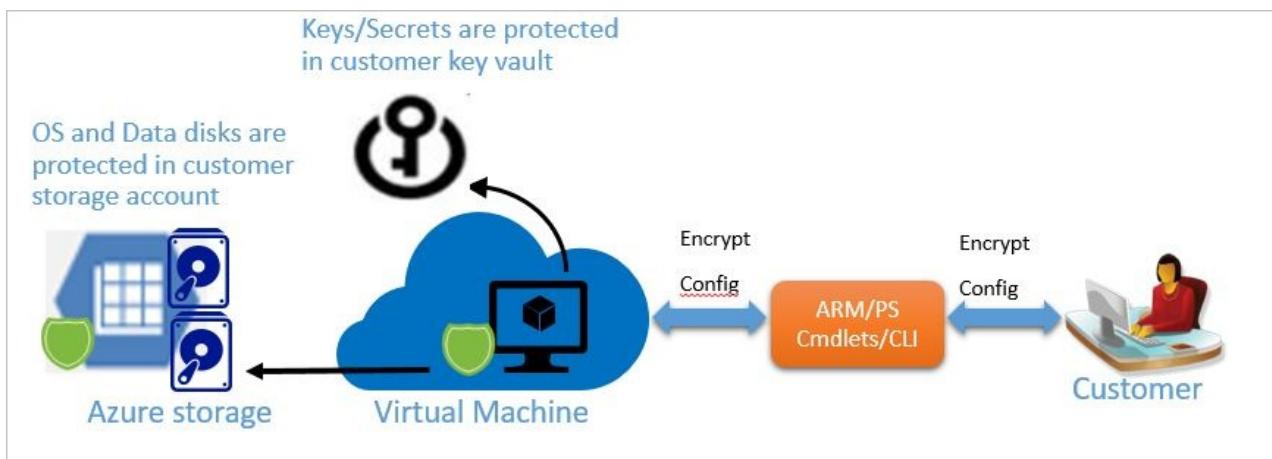
Encryption Workflow

The high level steps required to enable disk encryption for Windows and Linux VM's are:

1. Customer chooses a encryption scenario from the above encryption scenarios
2. Customer opts into enabling disk encryption via the Azure disk encryption Resource Manager template or

PS cmdlets or CLI command and specifies the encryption configuration

- For the customer encrypted VHD scenario, the customer uploads the encrypted VHD to their storage account and encryption key material to their key vault and provide the encryption configuration to enable encryption on a new IaaS VM
 - For the new VM's created from the Azure gallery and existing VM's already running in Azure, customer provide the encryption configuration to enable encryption on the IaaS VM
3. Customer grants access to Azure platform to read the encryption key material (BitLocker Encryption Keys for Windows systems and Passphrase for Linux) from their key vault to enable encryption on the IaaS VM
 4. Customer provide Azure AD application identity to write the encryption key material to their key vault to enable encryption on the IaaS VM for scenarios mentioned in #2 above
 5. Azure updates the VM service model with encryption and key vault configuration and provisions encrypted VM for the customer



Decryption Workflow

The high level steps required to disable disk encryption for IaaS VM's are:

1. Customer chooses to disable encryption (decryption) on a running IaaS VM in Azure via the Azure disk encryption Resource Manager template or PS cmdlets and specifies the decryption configuration.
2. The disable encryption step disables encryption of the OS or data volume or both on the running Windows IaaS VM. However disabling OS disk encryption for Linux is not supported as mentioned in the documentation above. The disable step is allowed only for data drives on Linux VMs.
3. Azure updates the VM service model and the IaaS VM is marked decrypted. The contents of the VM are not encrypted at rest anymore.
4. The disable encryption operation does not delete the customer key vault and the encryption key material, - BitLocker Encryption Keys for Windows or Passphrase for Linux.

Prerequisites

The following are prerequisites to enable Azure Disk Encryption on Azure IaaS VMs for the supported scenarios called out in the overview section

- User must have a valid active Azure subscription to create resources in Azure in the regions supported
- Azure Disk Encryption is supported on the following Windows server SKU's - Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2, and Windows Server 2016.
- Azure Disk Encryption is supported on the following Windows client SKU's - Windows 8 Client and Windows 10 Client.

NOTE

For Windows Server 2008 R2, .Net framework 4.5 MUST be installed before enabling encryption in Azure. You can install it from Windows update by installing the optional update "Microsoft .NET Framework 4.5.2 for Windows Server 2008 R2 x64-based Systems ([KB2901983](#))"

Azure Disk Encryption is supported on the following Linux server SKUs - Ubuntu, CentOS, SUSE and SUSE Linux Enterprise Server (SLES) and Red Hat Enterprise Linux.

NOTE

Linux OS disk encryption is currently supported on the following Linux distributions - RHEL 7.2, CentOS 7.2n, Ubuntu 16.04

All resources (Ex: Key Vault, Storage account, VM, etc.,) must belong to the same Azure region and subscription.

NOTE

Azure disk encryption requires that the Key Vault and the VMs reside in the same Azure region. Configuring them in separate region will cause failure in enabling Azure disk encryption feature.

- To set up and configure Azure Key Vault for Azure disk encryption usage, see section **Setting and Configuring Azure Key Vault for Azure disk encryption usage** in the *Prerequisites* section of this article.
- To set up and configure Azure AD application in Azure Active directory for Azure disk encryption usage, see section **Setup the Azure AD Application in Azure Active Directory** in the *Prerequisites* section of this article.
- To set up and configure Key Vault Access policy for the Azure AD Application, see section **Setting Key Vault Access policy for the Azure AD Application** in the *Prerequisites* section of this article.
- To prepare a pre-encrypted Windows VHD, see section **Preparing a pre-encrypted Windows VHD** in the Appendix of this article.
- To prepare a pre-encrypted Linux VHD, see section **Preparing a pre-encrypted Linux VHD** in the Appendix of this article.
- Azure platform needs access to the encryption keys or secrets in customer Azure Key Vault in order to make them available to the virtual machine to boot and decrypt the virtual machine OS volume. To grant permissions to Azure platform to access the customer Key Vault, **enabledForDiskEncryption** property must be set on the Key Vault for this requirement. Refer to section **Setting and Configuring Azure Key Vault for Azure disk encryption usage** in the Appendix of this article for more details.
- The Key Vault secret and key encryption key (KEK) URLs must be versioned. Azure enforces this restriction of versioning. See below examples for valid secret and KEK URL:
 - Example of valid secret URL:
<https://contosovault.vault.azure.net/secrets/BitLockerEncryptionSecretWithKek/xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - Example of valid KRK KEK:
<https://contosovault.vault.azure.net/keys/diskencryptionkek/xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
- Azure disk encryption does not support port numbers being specified as part of Key Vault secret and KEK URLs. See below examples for supported Key Vault URL:
 - Unaccepted Key Vault URL
<https://contosovault.vault.azure.net:443/secrets/contososecret/xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - Accepted Key Vault URL:
<https://contosovault.vault.azure.net/secrets/contososecret/xxxxxxxxxxxxxxxxxxxxxxxxxxxx>
- To enable Azure Disk Encryption feature, the IaaS VMs must meet the following network endpoint configuration requirements:
 - The IaaS VM must be able to connect to Azure Active Directory endpoint [Login.windows.net] to get a

- token to connect to Azure key vault
- The IaaS VM must be able to connect to Azure Key Vault endpoint to write the encryption keys to customer key vault
- The IaaS VM must be able to connect to Azure storage endpoint which hosts the Azure extension repository and Azure storage account which hosts the VHD files

NOTE

If your security policy limits access from Azure VMs to Internet, you can resolve the above URI to which you need connectivity and configure a specific rule to allow outbound connectivity to the IPs.

Use the latest version of Azure PowerShell SDK version to configure Azure Disk Encryption. Download the latest version of [Azure PowerShell release](#)

NOTE

Azure Disk Encryption is not supported on [Azure PowerShell SDK version 1.1.0](#). If you are receiving an error related to using Azure PowerShell 1.1.0, please see the article [Azure Disk Encryption Error Related to Azure PowerShell 1.1.0](#).

- To run any of the Azure CLI commands and associate it with your Azure subscription, you must first install Azure CLI version:
 - To install Azure CLI and associate it with your Azure subscription, see [How to install and configure Azure CLI](#)
 - Using the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager, see [here](#)
- Azure disk encryption solution use BitLocker external key protector for Windows IaaS VMs. If your VMs are domain joined, do not push any group policies that enforce TPM protectors. Refer to [this article](#) for details on the group policy for "Allow BitLocker without a compatible TPM".
- The Azure disk encryption prerequisite PowerShell script to create Azure AD application, create new key vault or setup existing key vault and enable encryption is located [here](#).
- To use Azure backup service to backup and restore encrypted VMs, when encryption is enabled using Azure disk encryption, you must encrypt your VMs using Azure disk encryption key encryption key configuration. The backup service supports VMs encrypted using key encryption key [KEK] configuration only. It does not support VMs encrypted without KEK. See disk encryption deployment scenarios below to enable VM encryption using KEK option.

Setup the Azure AD Application in Azure Active Directory

When encryption needs to be enabled on a running VM in Azure, Azure disk encryption generates and writes the encryption keys to your Key Vault. Managing encryption keys in Key Vault needs Azure AD authentication.

For this purpose, an Azure AD application should be created. Detailed steps for registering an application can be found here, in the section "Get an Identity for the Application" section in this [blog post](#). This post also contains a number of helpful examples on provisioning and configuring your Key Vault. For authentication purposes, either client secret based authentication or client certificate-based Azure AD authentication can be used.

Client secret based authentication for Azure AD

The sections that follow have the necessary steps to configure a client secret based authentication for Azure AD.

Create a new Azure AD app using Azure PowerShell

Use the PowerShell cmdlet below to create a new Azure AD app:

```
$aadClientSecret = "yourSecret"  
$azureAdApplication = New-AzureRmADApplication -DisplayName "<Your Application Display Name>" -HomePage "  
<https://YourApplicationHomePage>" -IdentifierUris "<https://YourApplicationUri>" -Password $aadClientSecret  
$servicePrincipal = New-AzureRmADServicePrincipal -ApplicationId $azureAdApplication.ApplicationId
```

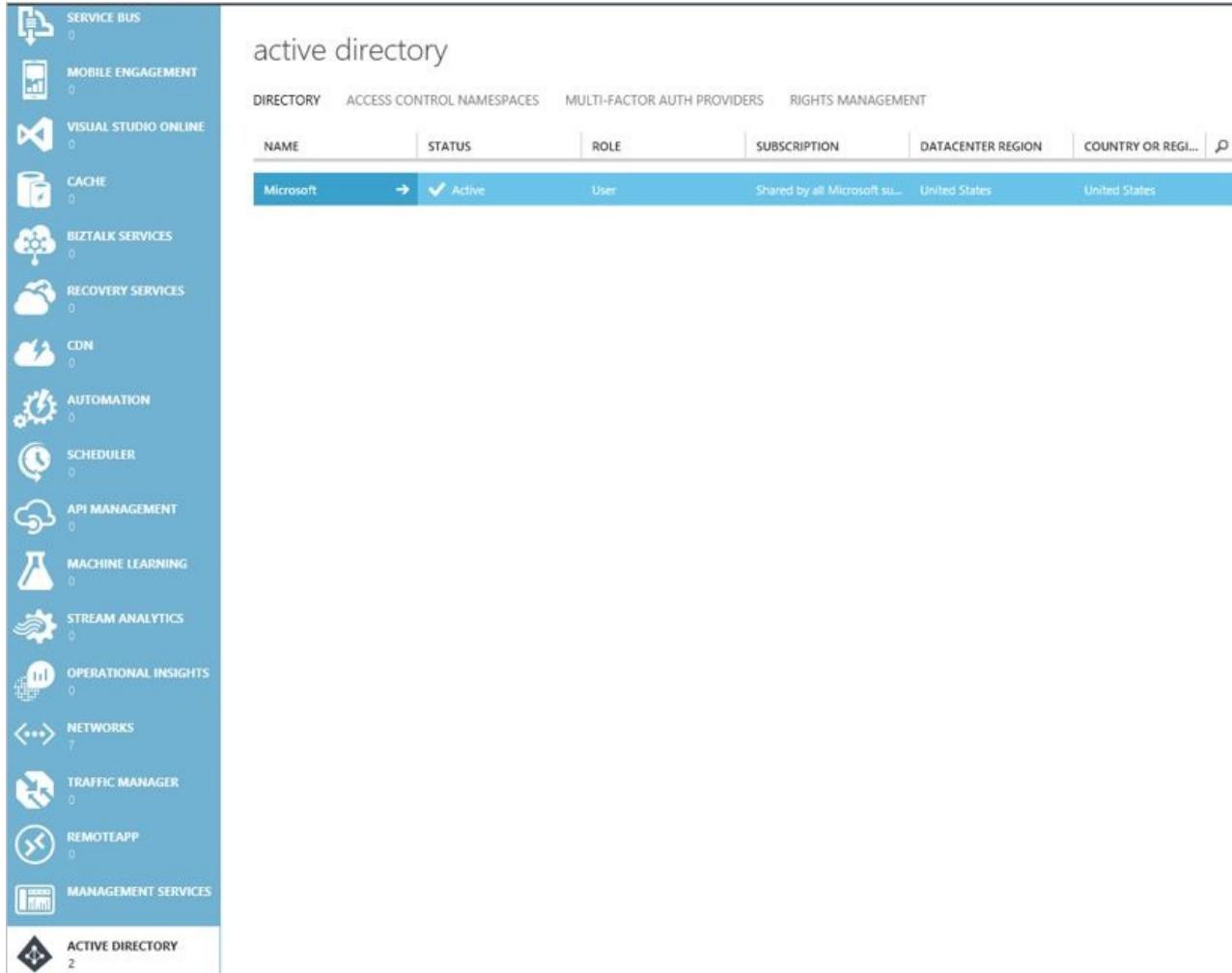
NOTE

\$azureAdApplication.ApplicationId is the Azure AD ClientID and \$aadClientSecret is the client Secret that you should use later to enable ADE. You should safeguard the Azure AD client secret appropriately.

Provisioning the Azure AD client ID and secret from the Azure Classic deployment model Portal

Azure AD Client ID and secret can also be provisioned using the Azure Classic deployment model Portal at <https://manage.windowsazure.com>, follow the steps below to perform this task:

1. Click the Active Directory tab as shown in Figure below:



The screenshot shows the Azure Classic deployment model Portal interface. On the left, there is a sidebar with various service icons and names: SERVICE BUS, MOBILE ENGAGEMENT, VISUAL STUDIO ONLINE, CACHE, BIZTALK SERVICES, RECOVERY SERVICES, CDN, AUTOMATION, SCHEDULER, API MANAGEMENT, MACHINE LEARNING, STREAM ANALYTICS, OPERATIONAL INSIGHTS, NETWORKS, TRAFFIC MANAGER, REMOTEAPP, MANAGEMENT SERVICES, and ACTIVE DIRECTORY. The ACTIVE DIRECTORY icon is highlighted with a red box. The main area is titled "active directory". It has tabs for DIRECTORY, ACCESS CONTROL NAMESPACES, MULTI-FACTOR AUTH PROVIDERS, and RIGHTS MANAGEMENT. Below these tabs is a table with columns: NAME, STATUS, ROLE, SUBSCRIPTION, DATACENTER REGION, and COUNTRY OR REGI... (partially visible). One row is visible in the table, showing "Microsoft" as the name, "Active" as the status, "User" as the role, "Shared by all Microsoft su..." as the subscription, "United States" as the datacenter region, and "United States" as the country or region. The entire screenshot is enclosed in a blue border.

2. Click Add Application and type the application name as shown below:

ADD APPLICATION ×

Tell us about your application

NAME ×

testkv1

Type 2

WEB APPLICATION AND/OR WEB API ?

NATIVE CLIENT APPLICATION ?

→ 2

3.Click the arrow button and configure the app's properties as shown below:

ADD APPLICATION ×

App properties

SIGN-ON URL ?

https://testkv1.azure.com ✓

APP ID URI ?

https://testkv1.azure.com ✓

1

← ✓

4.Click the check mark in the lower left corner to finish. The app's configuration page appears. Notice the Azure AD Client ID is located in the bottom of the page as shown in figure below.

testkv1

DASHBOARD USERS CONFIGURE OWNERS

properties

NAME	<input type="text" value="testkv1"/>	
SIGN-ON URL	<input type="text" value="https://testkv1.azure.com"/>	
LOGO		
APPLICATION IS MULTI-TENANT	<input type="radio"/> YES <input checked="" type="radio"/> NO	
CLIENT ID	<input type="text" value="2fcdee7-f4ee-47c8-9469-72bcfc639f"/>	
USER ASSIGNMENT REQUIRED TO ACCESS APP	<input type="radio"/> YES <input checked="" type="radio"/> NO	

5. Save the Azure AD client secret by click in the Save button. Click the save button and note the secret from the keys text box, this is the Azure AD client secret. You should safeguard the Azure AD client secret appropriately.

The screenshot shows the Azure portal interface for creating a new Azure AD application. In the 'keys' section, a key value is being generated with a 2-year expiration. In the 'single sign-on' section, the app ID URI is set to <https://testkv1.azure.com>. In the 'permissions to other applications' section, there are application and delegated permissions listed.

NOTE

The flow above is not supported in the Portal.

Use an existing app

In order to execute the commands below you need the Azure AD PowerShell module, which can be obtained from [here](#).

NOTE

The commands below must be executed from a new PowerShell window. Do NOT use Azure PowerShell or the Azure Resource Manager window to execute these commands. The reason for this recommendation is because these cmdlets are in the MSOnline module or Azure AD PowerShell.

```
$clientSecret = '<yourAadClientSecret>'  
$aadClientID = '<Client ID of your AAD app>'  
connect-msolservice  
New-MsolServicePrincipalCredential -AppPrincipalId $aadClientID -Type password -Value $clientSecret
```

Certificate based authentication for Azure AD

NOTE

AAD certificate based authentication is currently not supported on Linux VMs.

The sections that follow have the necessary steps to configure a certificate based authentication for Azure AD.

Create a new Azure AD app

Execute the PowerShell cmdlets below to create a new Azure AD app:

NOTE

Replace `yourpassword` string below with your secure password and safeguard the password.

```
$cert = New-Object  
System.Security.Cryptography.X509Certificates.X509Certificate("C:\certificates\examplecert.pfx", "yourpassword")  
$keyValue = [System.Convert]::ToBase64String($cert.GetRawCertData())  
$azureAdApplication = New-AzureRmADApplication -DisplayName "<Your Application Display Name>" -HomePage "  
<https://YourApplicationHomePage>" -IdentifierUris "<https://YouApplicationUri>" -KeyValue $keyValue -KeyType  
AsymmetricX509Cert  
$servicePrincipal = New-AzureRmADServicePrincipal -ApplicationId $azureAdApplication.ApplicationId
```

Once you finish this step, upload a .pfx file to Key Vault and enable the access policy needed to deploy that certificate to a VM.

Use an existing Azure AD app

If you are configuring certificate based authentication for an existing app, use the PowerShell cmdlets below. Make sure to execute them from a new PowerShell window.

```
$certLocalPath = 'C:\certs\myaadapp.cer'  
$aadClientID = '<Client ID of your AAD app>'  
connect-msolservice  
$cer = New-Object System.Security.Cryptography.X509Certificates.X509Certificate  
$cer.Import($certLocalPath)  
$binCert = $cer.GetRawCertData()  
$credValue = [System.Convert]::ToBase64String($binCert);  
New-MsolServicePrincipalCredential -AppPrincipalId $aadClientID -Type asymmetric -Value $credValue -Usage verify
```

Once you finish this step, upload a .pfx file to Key Vault and enable the access policy needed to deploy that certificate to a VM.

Upload a PFX file to Key Vault

You can read this [blog post](#) for detail explanation on how this process works. However, the PowerShell cmdlets below are all you need for this task. Make sure to execute them from Azure PowerShell console:

NOTE

Replace `yourpassword` string below with your secure password and safeguard the password.

```

$certLocalPath = 'C:\certs\myaadapp.pfx'
$certPassword = "yourpassword"
$resourceGroupName = 'yourResourceGroup'
$keyVaultName = 'yourKeyVaultName'
$keyVaultSecretName = 'yourAadCertSecretName'

$fileContentBytes = get-content $certLocalPath -Encoding Byte
$fileContentEncoded = [System.Convert]::ToBase64String($fileContentBytes)

$jsonObject = @"
{
  "data": "$fileContentEncoded",
  "dataType" : "pfx",
  "password": "$certPassword"
}
"@

$jsonObjectBytes = [System.Text.Encoding]::UTF8.GetBytes($jsonObject)
$jsonEncoded = [System.Convert]::ToBase64String($jsonObjectBytes)

Switch-AzureMode -Name AzureResourceManager
$secret = ConvertTo-SecureString -String $jsonEncoded -AsPlainText -Force
Set-AzureKeyVaultSecret -VaultName $keyVaultName -Name $keyVaultSecretName -SecretValue $secret
Set-AzureRmKeyVaultAccessPolicy -VaultName $keyVaultName -ResourceGroupName $resourceGroupName -
EnabledForDeployment

```

Deploy a certificate in Key Vault to an existing VM

After finishing uploading the PFX, use the steps below to deploy a certificate in Key Vault to an existing VM:

```

$resourceGroupName = 'yourResourceGroup'
$keyVaultName = 'yourKeyVaultName'
$keyVaultSecretName = 'yourAadCertSecretName'
$vmName = 'yourVMName'
$certUrl = (Get-AzureKeyVaultSecret -VaultName $keyVaultName -Name $keyVaultSecretName).Id
$sourceVaultId = (Get-AzureRmKeyVault -VaultName $keyVaultName -ResourceGroupName $resourceGroupName).ResourceId
$vm = Get-AzureRmVM -ResourceGroupName $resourceGroupName -Name $vmName
$vm = Add-AzureRmVMSSecret -VM $vm -SourceVaultId $sourceVaultId -CertificateStore "My" -CertificateUrl $certUrl
Update-AzureRmVM -VM $vm -ResourceGroupName $resourceGroupName

```

Setting Key Vault Access policy for the Azure AD Application

Your Azure AD application needs rights to access the keys or secrets in the vault. Use the [Set-AzureKeyVaultAccessPolicy](#) cmdlet to grant permissions to the application, using the Client Id (which was generated when the application was registered) as the –ServicePrincipalName parameter value. You can read [this blog post](#) for some examples on that. Below you also have an example of how to perform this task via PowerShell:

```

$keyVaultName = '<yourKeyVaultName>'
$aadClientID = '<yourAadAppClientID>'
$rgname = '<yourResourceGroup>'
Set-AzureRmKeyVaultAccessPolicy -VaultName $keyVaultName -ServicePrincipalName $aadClientID -PermissionsToKeys
'WrapKey' -PermissionsToSecrets 'Set' -ResourceGroupName $rgname

```

NOTE

Azure disk encryption requires you to configure the following access policies to your AAD Client Application - 'WrapKey' and 'Set' permissions

Terminology

Use the terminology table as reference to understand some of the common terms used by this technology:

TERMINOLOGY	DEFINITION
Azure AD	Azure AD is Azure Active Directory . Azure AD account is a prerequisite for authenticating, storing, and retrieving secrets from the Key Vault.
Azure Key Vault [AKV]	Azure Key Vault is a cryptographic key management service based on FIPS-validated Hardware Security Modules to safeguard your cryptographic keys and sensitive secrets securely.,Refer to Key Vault documentation for more details.
ARM	Azure Resource Manager
BitLocker	BitLocker is an industry recognized Windows volume encryption technology used to enable disk encryption on Windows IaaS VMs
BEK	BitLocker Encryption Keys are used to encrypt the OS boot volume and data volumes. The BitLocker keys are safeguard in customer's Azure key vault as secrets.
CLI	Azure Command-Line Interface
DM-Crypt	DM-Crypt is the Linux-based transparent disk encryption subsystem used to enable disk encryption on Linux IaaS VMs
KEK	Key Encryption Key is the asymmetric key (RSA 2048) used to protect or wrap the secret if desired. You can provide an HSM-protected key or software-protected key. For more details, refer to Azure Key Vault documentation for more details
PS cmdlets	Azure PowerShell cmdlets

Setting and Configuring Azure Key Vault for Azure disk encryption usage

Azure disk encryption safeguards the disk encryption keys and secrets in your Azure Key Vault. Follow the steps on each one of the sections below to setup Key Vault for Azure disk encryption usage.

Create a New Key Vault

To create a new Key Vault, use one of the options listed below:

- Use the "101-Create-KeyVault" Resource Manager template located [here](#)
- Use the Azure PowerShell [Key Vault cmdlets](#).
- Use the Azure resource manager portal.

NOTE

If you already have a Key Vault setup for your subscription, please proceed to next section.

Everything

Filter

Key Vault

Results

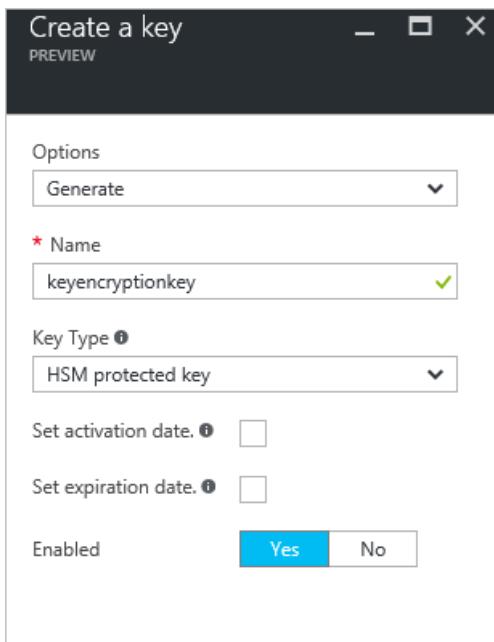
NAME	PUBLISHER	CATEGORY
Key Vault (preview)	Microsoft	Management

Provisioning a Key Encryption Key (optional)

If you wish to use a Key Encryption Key (KEK) for an additional layer of security to wrap the BitLocker encryption keys, you should add a KEK to your Key Vault for use in the provisioning process. Use the [Add-AzureKeyVaultKey](#) cmdlet to create a new Key Encryption Key in Key Vault. You can also import KEK from your on-premises key management HSM. For more details, see [Key Vault documentation](#).

```
Add-AzureKeyVaultKey [-VaultName] <string> [-Name] <string> -Destination <string> {HSM | Software}
```

The KEK can be added from Azure Resource Manager portal as well using Azure Key Vault UX.



Set Key Vault permissions to allow the Azure platform access to the keys and secrets

The Azure platform needs access to the encryption keys or secrets in your Azure Key Vault in order to make them available to the VM to boot and decrypt the volumes. To grant permissions to the Azure platform so that it can access the Key Vault, the *enabledForDiskEncryption* property must be set on the Key Vault. You can set the *enabledForDiskEncryption* property on your key vault using the key vault PS cmdlet:

```
Set-AzureRmKeyVaultAccessPolicy -VaultName <yourVaultName> -ResourceGroupName <yourResourceGroup> -  
EnabledForDiskEncryption
```

You can also set the *enabledForDiskEncryption* property by visiting <https://resources.azure.com>. You must set the *enabledForDiskEncryption* property on your Key Vault as mentioned before. Otherwise the deployment will fail.

You can setup access policies for your AAD application from the Key Vault UX:

Add new permissions

Add a new access policy - PREVIEW

* Select principal
vmencrypt >

Configure from template (optional)

Key permissions
1 selected >

Secret permissions
1 selected >

Authorized application ⓘ
None selected

Key permissions

Configure permitted operations - PREV...

All Key Operations
 All

Key Management Operations

Get

List

Update

Create

Import

Delete

Backup

Restore

Cryptographic Operations

Decrypt

Encrypt

UnwrapKey

WrapKey

Verify

Sign

Add new permissions

Add a new access policy - PREVIEW

* Select principal
vmencrypt >

Configure from template (optional)

Key permissions
1 selected >

Secret permissions
1 selected >

Authorized application ⓘ
None selected

Secret permissions

Configure permitted operations - PREV...

All Secret Operations
 All

Secret Management Operations

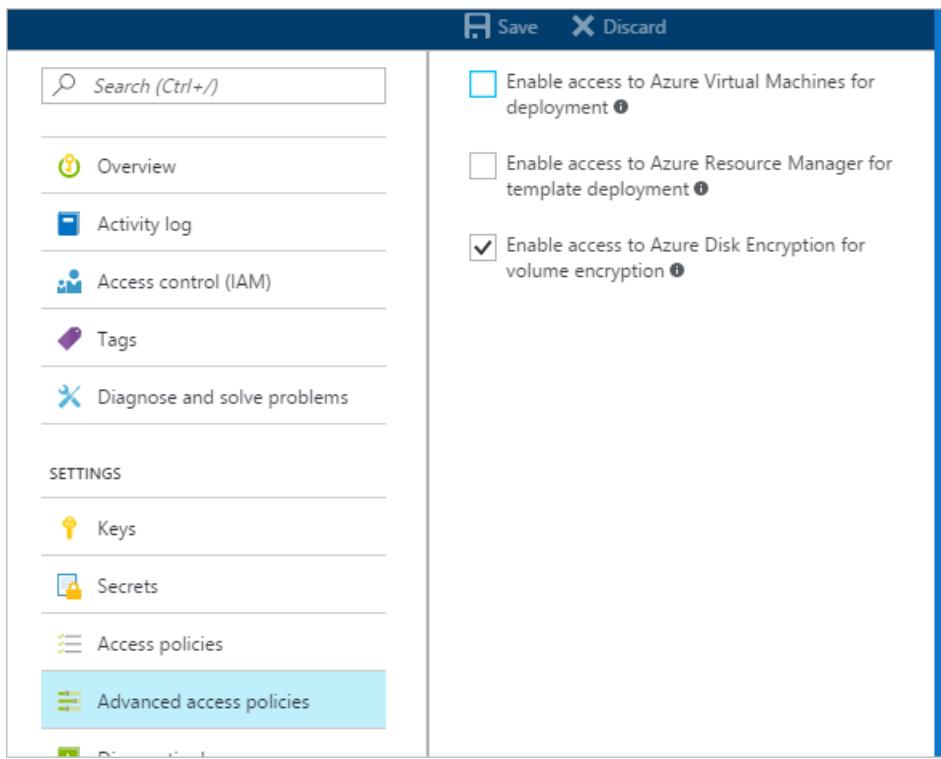
Get

List

Set

Delete

Make sure that Key Vault is enabled for Disk Encryption in "Advanced Access Policies":



Disk Encryption deployment scenarios and user experiences

There are many scenarios that you can enable disk encryption and the steps may vary according to the scenario. The sections that follow will cover in more details these scenarios.

Enable encryption on new IaaS VM's created from the Azure Gallery

Disk encryption can be enabled on new IaaS Windows VM from Azure gallery in Azure using the Resource Manager template published [here](#). Click on "Deploy to Azure" button on the Azure quickstart template, input encryption configuration in the parameters blade and click OK. Select the subscription, resource group, resource group location, legal terms and agreement and click Create button to enable encryption on a new IaaS VM.

NOTE

This template creates a new encrypted Windows VM using the Windows Server 2012 gallery image.

Disk encryption can be enabled on a new IaaS RedHat Linux 7.2 VM with a 200 GB RAID-0 array using [this](#) resource manager template. After the template is deployed, verify the VM encryption status using the `Get-AzureRmVmDiskEncryptionStatus` cmdlet as described in the section "[Encrypting OS drive on a running Linux VM](#)". When the machine returns status `VMRestartPending`, restart the VM.

You can see the Resource Manager template parameters details for new VM from Azure gallery scenario using Azure AD Client ID in the table below:

PARAMETER	DESCRIPTION
adminUserName	Admin user name for the virtual machine
adminPassword	Admin user password for the virtual machine
newStorageAccountName	Name of the storage account to store OS and data VHDS

PARAMETER	DESCRIPTION
vmSize	Size of the VM. Currently, only Standard A, D and G series are supported
virtualNetworkName	Name of the VNet to which the VM NIC should belong to.
subnetName	Name of the subnet in the vNet to which the VM NIC should belong to
AADClientID	Client ID of the Azure AD app that has permissions to write secrets to Key Vault
AADClientSecret	Client Secret of the Azure AD app that has permissions to write secrets to Key Vault
keyVaultURL	URL of the Key Vault to which BitLocker key should be uploaded to. You can get it using the cmdlet: (Get-AzureRmKeyVault -VaultName,-ResourceGroupName).VaultURI
keyEncryptionKeyURL	URL of the Key Encryption Key that's used to encrypt the generated BitLocker key. This is optional.
keyVaultResourceGroup	Resource Group of the key vault
vmName	Name of the VM on which encryption operation is to be performed

NOTE

KeyEncryptionKeyURL is an optional parameter. You can bring your own KEK to further safeguard the data encryption key (Passphrase secret) in Key Vault.

Enable encryption on new IaaS VM's created from Customer Encrypted VHD and encryption keys

In this scenario you can enable encrypting by using the Resource Manager template, PowerShell cmdlets or CLI commands. The sections below will explain in more details the Resource Manager template and CLI commands.

Follow the instructions from one of these sections for preparing pre-encrypted images that can be used in Azure. Once the image is created, the steps in the next section can be used for creating an encrypted Azure VM.

- [Preparing a pre-encrypted Windows VHD](#)
- [Preparing a pre-encrypted Linux VHD](#)

Using Resource Manager template

Disk encryption can be enabled on customer encrypted VHD using the Resource Manager template published [here](#). Click on “Deploy to Azure” button on the Azure quickstart template, input encryption configuration in the parameters blade and click OK. Select the subscription, resource group, resource group location, legal terms and agreement and click Create button to enable encryption on new IaaS VM.

The Resource Manager template parameters details for customer encrypted VHD scenario are described in the table below:

PARAMETER	DESCRIPTION
newStorageAccountName	Name of the storage account to store encrypted OS vhd. This storage account should have already been created in the same resource group and same location as the VM
osVhdUri	URI of OS vhd from storage account
osType	OS product type (Windows/Linux)
virtualNetworkName	Name of the VNet to which the VM NIC should belong to. This should have been already created in the same resource group and same location as the VM
subnetName	Name of the subnet in the vNet to which the VM NIC should belong to
vmSize	Size of the VM. Currently, only Standard A, D and G series are supported
keyVaultResourceId	ResourceId identifying the key vault resource in ARM. You can get it using the PowerShell cmdlet: (Get-AzureRmKeyVault -VaultName <yourKeyVaultName> -ResourceGroupName <yourResourceGroupName>).ResourceId
keyVaultSecretUrl	URL of the disk encryption key provisioned in key vault
keyVaultKekUrl	URL of the Key Encryption Key that's to encrypt the generated disk encryption key
vmName	Name of the IaaS VM

Using PowerShell cmdlets

Disk encryption can be enabled on customer encrypted VHD using the PS cmdlets published [here](#).

Using CLI Commands

Follow the steps below to enable disk encryption for this scenario using CLI commands:

1. Set access policies on Key Vault:

- Set 'EnabledForDiskEncryption' flag:

```
azure keyvault set-policy --vault-name <keyVaultName> --enabled-for-disk-encryption true
```

- Set permissions to Azure AD app to write secrets to KeyVault:

```
azure keyvault set-policy --vault-name <keyVaultName> --spn <aadClientID> --perms-to-keys
['wrapKey'] --perms-to-secrets '['set']'
```

2. To enable encryption on an existing/running VM, type:

```
azure vm enable-disk-encryption --resource-group <resourceGroupName> --name <vmName> --aad-client-id
<aadClientId> --aad-client-secret <aadClientSecret> --disk-encryption-key-vault-url <keyVaultURL> --disk-
encryption-key-vault-id <keyVaultResourceId> --volume-type [All|OS|Data]
```

3. Get encryption status:

```
azure vm show-disk-encryption-status --resource-group <resourceGroupName> --name <vmName> --json
```

4. To enable encryption on a new VM from customer encrypted VHD, use the below parameters with "azure vm create" command:

- disk-encryption-key-vault-id
- disk-encryption-key-url
- key-encryption-key-vault-id

- key-encryption-key-url

Enable encryption on existing or running IaaS Windows VM in Azure

In this scenario you can enable encrypting by using the Resource Manager template, PowerShell cmdlets or CLI commands. The sections below will explain in more details how to enable it using Resource Manager template and CLI commands.

Using Resource Manager template

Disk encryption can be enabled on existing/running IaaS Windows VM in Azure using the Resource Manager template published [here](#). Click on “Deploy to Azure” button on the Azure quickstart template, input encryption configuration in the parameters blade and click OK. Select the subscription, resource group, resource group location, legal terms and agreement and click Create button to enable encryption on existing/running IaaS VM.

The Resource Manager template parameters details for existing/running VM scenario using Azure AD Client ID are available in the table below:

PARAMETER	DESCRIPTION
AADClientID	Client ID of the Azure AD app that has permissions to write secrets to Key Vault
AADClientSecret	Client Secret of the Azure AD app that has permissions to write secrets to Key Vault
keyVaultName	Name of the Key Vault to which BitLocker key should be uploaded to. You can get it using the cmdlet: (Get-AzureRmKeyVault -ResourceGroupName). Vaultname
keyEncryptionKeyURL	URL of the Key Encryption Key that's used to encrypt the generated BitLocker key. This is optional if you select <code>nokek</code> in the UseExistingKek dropdown. If you select <code>kek</code> in the UseExistingKek dropdown, you must input the keyEncryptionKeyURL value
volumeType	Type of the volume on which encryption operation is performed. Valid values are "OS", "Data", "All"
sequenceVersion	Sequence version of the BitLocker operation. Increment this version number every time a disk encryption operation is performed on the same VM
vmName	Name of the VM on which encryption operation is to be performed

NOTE

KeyEncryptionKeyURL is an optional parameter. You can bring your own KEK to further safeguard the data encryption key (BitLocker encryption secret) in Key Vault.

Using PowerShell cmdlets

Refer to the **Explore Azure disk encryption with Azure PowerShell** blog post [part 1](#) and [part 2](#) for details on how to enable encryption using Azure Disk Encryption using PS cmdlets.

Using CLI Commands

Follow the steps below to enable encryption on existing/running IaaS Windows VM in Azure using CLI commands:

1. Set access policies on Key Vault:

- Set 'EnabledForDiskEncryption' flag:

```
azure keyvault set-policy --vault-name <keyVaultName> --enabled-for-disk-encryption true
```

- Set permissions to Azure AD app to write secrets to KeyVault:

```
azure keyvault set-policy --vault-name <keyVaultName> --spn <aadClientID> --perms-to-keys  
['wrapKey'] --perms-to-secrets '['set']'
```

2. To enable encryption on an existing/running VM:

```
azure vm enable-disk-encryption --resource-group <resourceGroupName> --name <vmName> --aad-client-id  
<aadClientId> --aad-client-secret <aadClientSecret> --disk-encryption-key-vault-url <keyVaultURL> --disk-  
encryption-key-vault-id <keyVaultResourceId> --volume-type [All|OS|Data]
```

3. Get encryption status:

```
azure vm show-disk-encryption-status --resource-group <resourceGroupName> --name <vmName> --json
```

4. To enable encryption on a new VM from customer encrypted VHD, use the below parameters with "azure vm create" command:

- disk-encryption-key-vault-id
- disk-encryption-key-url
- key-encryption-key-vault-id
- key-encryption-key-url

Enable encryption on existing or running IaaS Linux VM in Azure

Disk encryption can be enabled on existing/running IaaS Linux VM in Azure using the Resource Manager template published [here](#). Click on "Deploy to Azure" button on the Azure quickstart template, input encryption configuration in the parameters blade and click OK. Select the subscription, resource group, resource group location, legal terms and agreement and click Create button to enable encryption on existing/running IaaS VM.

The Resource Manager template parameters details for existing/running VM scenario using Azure AD Client ID are described in the table below:

PARAMETER	DESCRIPTION
AADClientID	Client ID of the Azure AD app that has permissions to write secrets to Key Vault
AADClientSecret	Client Secret of the Azure AD app that has permissions to write secrets to Key Vault
keyVaultName	Name of the Key Vault to which BitLocker key should be uploaded to. You can get it using the cmdlet: (Get-AzureRmKeyVault -ResourceGroupName). Vaultname
keyEncryptionKeyURL	URL of the Key Encryption Key that's used to encrypt the generated BitLocker key. This is optional if you select "nokek" in the UseExistingKek dropdown. If you select "kek" in the UseExistingKek dropdown, you must input the keyEncryptionKeyURL value
volumeType	Type of the volume on which encryption operation is performed. Valid supported values are "OS"/"All" (for RHEL 7.2, CentOS 7.2 & Ubuntu 16.04) and "Data" for all other distros.
sequenceVersion	Sequence version of the BitLocker operation. Increment this version number every time a disk encryption operation is performed on the same VM

PARAMETER	DESCRIPTION
vmName	Name of the VM on which encryption operation is to be performed
passPhrase	Type a strong passphrase as the data encryption key

NOTE

KeyEncryptionKeyURL is an optional parameter. You can bring your own KEK to further safeguard the data encryption key (Passphrase secret) in Key Vault.

CLI Commands

Disk encryption can be enabled on customer encrypted VHD using the CLI command installed from [here](#). Follow the steps below to enable encryption on existing/running IaaS Linux VM in Azure using CLI commands:

1. Set access policies on Key Vault:

- Set 'EnabledForDiskEncryption' flag:

```
azure keyvault set-policy --vault-name <keyVaultName> --enabled-for-disk-encryption true
```

- Set permissions to Azure AD app to write secrets to KeyVault:

```
azure keyvault set-policy --vault-name <keyVaultName> --spn <aadClientID> --perms-to-keys
['wrapKey'] --perms-to-secrets '['set']'
```

2. To enable encryption on an existing/running VM:

```
azure vm enable-disk-encryption --resource-group <resourceGroupName> --name <vmName> --aad-client-id
<aadClientId> --aad-client-secret <aadClientSecret> --disk-encryption-key-vault-url <keyVaultURL> --disk-
encryption-key-vault-id <keyVaultResourceId> --volume-type [All|OS|Data]
```

3. Get encryption status:

```
azure vm show-disk-encryption-status --resource-group <resourceGroupName> --name <vmName> --json
```

4. To enable encryption on a new VM from customer encrypted VHD, use the below parameters with "azure vm create" command.

- disk-encryption-key-vault-id*
- disk-encryption-key-url*
- key-encryption-key-vault-id*
- key-encryption-key-url*

Get encryption status of an encrypted IaaS VM

You can get encryption status using Azure Resource Manager portal, [PowerShell cmdlets](#) or CLI commands. The sections below will explain how to use the Azure portal and CLI commands to get the encryption status.

Get encryption status of an encrypted Windows VM using Azure Resource Manager portal

You can get the encryption status of the IaaS VM from Azure Resource Manager portal. Logon to Azure portal at <https://portal.azure.com/>, click on virtual machines link in the left menu to see summary view of the virtual machines in your subscription. You can filter the virtual machines view by selecting the subscription name from the subscription dropdown. Click on columns located at the top of the virtual machines page menu. Select Disk Encryption column from the choose column blade and click update. You should see the disk encryption column showing the encryption state "Enabled" or "Not Enabled" for each VM as shown in the figure below.

Name	Status	Location	Size	Disk Encryption
ADEDemoCAT	Running	Australia East	Standard_D1	Enabled
ADEPreDemoCAT	Running	Australia East	Standard_D1	Enabled
at-east	Running	East US	Standard_A1	Enabled
at-prevm10	Running	Australia East	Standard_D2	Enabled

Get encryption status of an encrypted (Windows/Linux) IaaS VM using disk encryption PS Cmdlet

You can get the encryption status of the IaaS VM from disk encryption PS cmdlet "Get-AzureRmVmDiskEncryptionStatus". To get the encryption settings for your VM, type in your Azure PowerShell session:

```
C:\> Get-AzureRmVmDiskEncryptionStatus -ResourceGroupName $ResourceGroupName -VMName $VMName
-ExtensionName $ExtensionName

OsVolumeEncrypted      : NotEncrypted
DataVolumesEncrypted    : Encrypted
OsVolumeEncryptionSettings : Microsoft.Azure.Management.Compute.Models.DiskEncryptionSettings
ProgressMessage         : https://rheltest1keyvault.vault.azure.net/secrets/bdb6bfb1-5431-4c28-af46-
b18d0025ef2a/abebacb83d864a5fa729508315020f8a
```

The output of Get-AzureRmVmDiskEncryptionStatus can be inspected for encryption key URLs.

```
C:\> $status = Get-AzureRmVmDiskEncryptionStatus -ResourceGroupName $ResourceGroupName -VMName $VMName -ExtensionName $ExtensionName
C:\> $status.OsVolumeEncryptionSettings

DiskEncryptionKey          KeyEncryptionKey
Enabled                   -----
-----
```

Microsoft.Azure.Management.Compute.Models.KeyVaultSecretReference	-----
Microsoft.Azure.Management.Compute.Models.KeyVaultKeyReference	True

```
C:\> $status.OsVolumeEncryptionSettings.DiskEncryptionKey.SecretUrl
https://rheltest1keyvault.vault.azure.net/secrets/bdb6bfb1-5431-4c28-af46-
b18d0025ef2a/abebacb83d864a5fa729508315020f8a
C:\> $status.OsVolumeEncryptionSettings.DiskEncryptionKey

SecretUrl
SourceVault
-----
```

```
https://rheltest1keyvault.vault.azure.net/secrets/bdb6bfb1-5431-4c28-af46-
b18d0025ef2a/abebacb83d864a5fa729508315020f8a Microsoft.Azure.Management....
```

The OsVolumeEncrypted and DataVolumesEncrypted settings value are set to "Encrypted" showing that both the volumes are encrypted using Azure disk encryption. Refer to the [Explore Azure disk encryption with Azure PowerShell](#) blog post [part 1](#) and [part 2](#) for details on how to enable encryption using Azure Disk Encryption using PS cmdlets.

NOTE

On Linux VMs, the `Get-AzureRmVMDiskEncryptionStatus` cmdlet takes 3-4 minutes to report the encryption status.

Get encryption status of the IaaS VM from disk encryption CLI command

You can get the encryption status of the IaaS VM from disk encryption CLI command `azure vm show-disk-encryption-status`. To get the encryption settings for your VM, type in your Azure CLI session:

```
azure vm show-disk-encryption-status --resource-group <yourResourceGroupName> --name <yourVMName> --json
```

Disable Encryption on running Windows IaaS VM

You can disable encryption on a running Windows or Linux IaaS VM via the Azure disk encryption Resource Manager template or PS cmdlets and specifies the decryption configuration.

Windows VM

The disable encryption step disables encryption of the OS or data volume or both on the running Windows IaaS VM. You cannot disable the OS volume and leave the data volume encrypted. When the disable encryption step is performed, Azure classic deployment model updates the VM service model and the Windows IaaS VM is marked decrypted. The contents of the VM are not encrypted at rest anymore. The disable encryption does not delete the customer key vault and the encryption key material, which is BitLocker Encryption Keys for Windows and Passphrase for Linux.

Linux VM

The disable encryption step disables encryption of the data volume on the running Linux IaaS VM

NOTE

Disabling encryption on OS disk is not allowed on Linux VMs.

Disable encryption on existing/running IaaS VM in Azure using Resource Manager template

Disk encryption can be disabled on running Windows IaaS VM using the Resource Manager template published [here](#). Click on “Deploy to Azure” button on the Azure quickstart template, input decryption configuration in the parameters blade and click OK. Select the subscription, resource group, resource group location, legal terms and agreement and click Create button to enable encryption on a new IaaS VM.

For Linux VM, [this](#) template can be used to disable encryption.

Resource Manager template parameters details for disabling encryption on running IaaS VM:

VMNAME	NAME OF THE VM ON WHICH ENCRYPTION OPERATION IS TO BE PERFORMED
volumeType	Type of the volume on which decryption operation is performed. Valid values are "OS", "Data", "All". Note: You cannot disable encryption on running Windows IaaS VM OS/boot volume without disabling encryption on "Data" volume. Note: Disabling encryption on OS disk is not allowed on Linux VMs.
sequenceVersion	Sequence version of the BitLocker operation. Increment this version number every time a disk decryption operation is performed on the same VM

Disable encryption on existing/running IaaS VM in Azure using PS cmdlet

To disable using the PS cmdlet, [Disable-AzureRmVMDiskEncryption](#) cmdlet disables encryption on an infrastructure as a service (IaaS) virtual machine. This cmdlet supports both Windows and Linux VMs. This cmdlet installs an extension on the virtual machine to disable encryption. If the Name parameter is not specified, an

extension with the default name "AzureDiskEncryption for Windows VMs" is created.

On Linux VMs, the "AzureDiskEncryptionForLinux" extension is used.

NOTE

This cmdlet reboots the virtual machine.

Appendix

Connect to your subscription

Make sure to review the *Prerequisites* section in this document before proceeding. After ensuring that all prerequisites were fulfilled, follow the steps below to connect to your subscription:

1. Start an Azure PowerShell session and sign in to your Azure account with the following command:

```
Login-AzureRmAccount
```

2. If you have multiple subscriptions and want to specify a specific one to use, type the following to see the subscriptions for your account:

```
Get-AzureRmSubscription
```

3. To specify the subscription you want to use, type:

```
Select-AzureRmSubscription -SubscriptionName <Yoursubscriptionname>
```

4. To verify the subscription configured is correct, type:

```
Get-AzureRmSubscription
```

5. To confirm the Azure Disk Encryption cmdlets are installed, type:

```
Get-command *diskencryption*
```

6. You should see the below output confirming Azure Disk Encryption PowerShell installation:

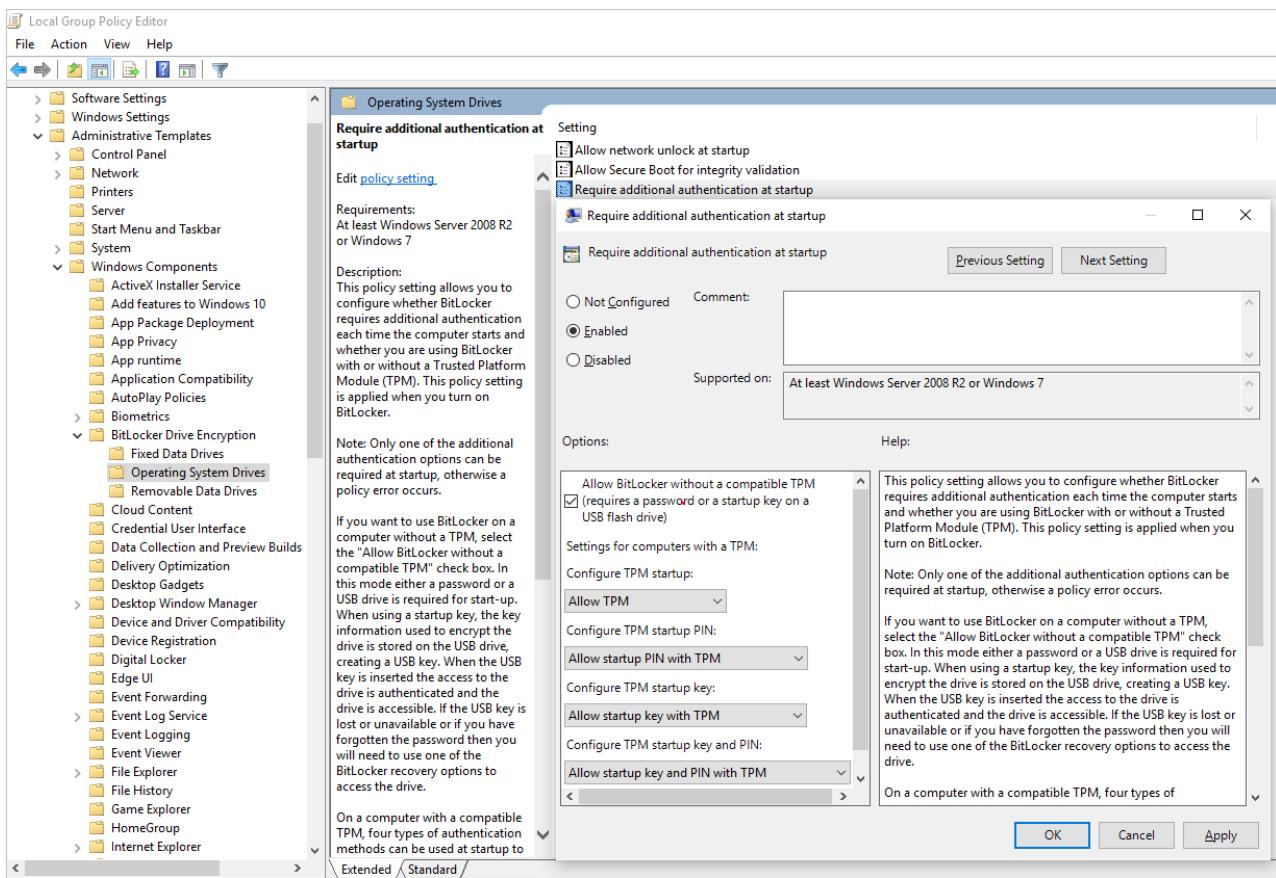
```
PS C:\Windows\System32\WindowsPowerShell\v1.0> get-command *diskencryption*
 CommandType     Name                                               Source
 Cmdlet        Get-AzureRmVMDiskEncryptionStatus                 AzureRM.Compute
 Cmdlet        Disable-AzureRmVMDiskEncryption                AzureRM.Compute
 Cmdlet        Set-AzureRmVMDiskEncryptionExtension            AzureRM.Compute
```

Preparing a pre-encrypted Windows VHD

The sections that follow are necessary in order to prepare a pre-encrypted Windows VHD for deployment as an encrypted VHD in Azure IaaS. The steps are used to prepare and boot a fresh windows VM (vhd) on Hyper-V or Azure.

Update group policy to allow non-TPM for OS protection

You need to configure the BitLocker Group Policy setting called BitLocker Drive Encryption, located under Local Computer Policy \Computer Configuration\Administrative Templates\Windows Components. Change this setting to: *Operating System Drives - Require additional authentication at startup - Allow BitLocker without a compatible TPM* as shown in the figure below:



Install BitLocker feature components

For Windows Server 2012 and above use the below command:

```
dism /online /Enable-Feature /all /FeatureName:Bitlocker /quiet /norestart
```

For Windows Server 2008 R2 use the below command:

```
ServerManagerCmd -install BitLockers
```

Prepare OS volume for BitLocker using [bdehcpl](#)

Execute the command below to compress the OS partition and prepare the machine for BitLocker.

```
bdehcfc -target c: shrink -quiet
```

Using BitLocker to protect the OS volume

Use the [manage-bde](#) command to enable encryption on the boot volume using an external key protector and place the external key (.bek file) on the external drive or volume. Encryption will be enabled on the system/boot volume after the next reboot.

```
manage-bde -on %systemdrive% -sk [ExternalDriveOrVolume]
reboot
```

NOTE

The VM needs to be prepared with a separate data/resource vhd for getting the external key using BitLocker.

Encrypting OS drive on a running Linux VM

Encryption of OS drive on a running Linux VM is supported on the following distros:

- RHEL 7.2

- CentOS 7.2
- Ubuntu 16.04

Prerequisites for OS disk encryption:

- VM must be created from Azure Gallery image in Azure Resource Manager portal.
- Azure VM with at least 4 GB of RAM (recommended size is 7 GB).
- (For RHEL and CentOS) SELinux must be [disabled](#) on the VM. The VM must be rebooted at least once after disabling SELinux.

Steps

1.Create a VM using one of the distros specified above.

For CentOS 7.2, OS disk encryption is supported via a special image. To use this image, specify "7.2n" as the Sku when creating the VM:

```
Set-AzureRmVMSourceImage -VM $VirtualMachine -PublisherName "OpenLogic" -Offer "CentOS" -Skus "7.2n" -Version "latest"
```

2.Configure the VM according to your needs. If you are going to encrypt all the (OS + data) drives the data drives need to be specified and mountable from /etc/fstab.

NOTE

You must use UUID=... to specify data drives in /etc/fstab instead of specifying the block device name, e.g., /dev/sdb1. During encryption the order of drives will change on the VM. If your VM relies on a specific order of block devices it will fail to mount them after encryption.

3.Logout SSH sessions.

4.To encrypt the OS, specify volumeType as "All" or "OS" when [enabling encryption](#).

NOTE

All user-space processes that are not running as `systemd` services shall be killed with a `SIGKILL`. The VM shall be rebooted. Please plan on downtime of the VM when enabling OS disk encryption on a running VM.

5.Periodically monitor the progress of encryption using instructions in the [next section](#).

6.Once Get-AzureRmVmDiskEncryptionStatus shows "VMRestartPending", restart your VM by either logging on to it or via Portal/PowerShell/CLI.

```
C:\> Get-AzureRmVmDiskEncryptionStatus -ResourceGroupName $ResourceGroupName -VMName $VMName
-ExtensionName $ExtensionName

OsVolumeEncrypted      : VMRestartPending
DataVolumesEncrypted    : NotMounted
OsVolumeEncryptionSettings : Microsoft.Azure.Management.Compute.Models.DiskEncryptionSettings
ProgressMessage         : OS disk successfully encrypted, please reboot the VM
```

It is recommended to save [boot diagnostics](#) of the VM *before* rebooting.

Monitoring OS encryption progress

There are three ways to monitor OS encryption progress.

1.Use the Get-AzureRmVmDiskEncryptionStatus cmdlet and inspect the ProgressMessage field:

```

OsVolumeEncrypted      : EncryptionInProgress
DataVolumesEncrypted   : NotMounted
OsVolumeEncryptionSettings : Microsoft.Azure.Management.Compute.Models.DiskEncryptionSettings
ProgressMessage        : OS disk encryption started

```

Once the VM reaches "OS disk encryption started" it will take roughly 40-50 minutes on a Premium-storage backed VM.

Due to [issue #388](#) in WALinuxAgent, `OsVolumeEncrypted` and `DataVolumesEncrypted` show up as `Unknown` in some distros. With WALinuxAgent version 2.1.5 and above this will be fixed automatically. In case you see `Unknown` in the output, you can verify disk encryption status by using Azure Resource Viewer.

Go to [Azure Resource Viewer](#), then expand this hierarchy in the selection panel on left:

```

|-- subscriptions
  |-- [Your subscription]
    |-- resourceGroups
      |-- [Your resource group]
        |-- providers
          |-- Microsoft.Compute
            |-- virtualMachines
              |-- [Your virtual machine]
                |-- InstanceView

```

In the InstanceView, scroll down to see the encryption status of your drives.

```

{
  "code": "ProvisioningState/succeeded",
  "level": "Info",
  "displayStatus": "Provisioning succeeded",
  "time": "2016-09-22T02:19:41.4646766+00:00"
}
],
"extensions": [
{
  "name": "AzureDiskEncryptionForLinux",
  "type": "Microsoft.Azure.Security.AzureDiskEncryptionForLinux",
  "typeHandlerVersion": "0.1.0.999190",
  "substatuses": [
    {
      "code": "ComponentStatus/Microsoft.Azure.Security.AzureDiskEncryptionForLinux",
      "level": "Info",
      "displayStatus": "Provisioning succeeded",
      "message": "{\"os\": \"NotEncrypted\\\", \"data\": \"EncryptionInProgress\\\"}"
    }
  ]
}
]
}

```

2. Look at [boot diagnostics](#). Messages from ADE extension shall be prefixed with `[AzureDiskEncryption]`.

3. Logon on to the VM via SSH and getting the extension log from

```
/var/log/azure/Microsoft.Azure.Security.AzureDiskEncryptionForLinux
```

It is not recommended to log on to the VM while OS encryption is in progress. Therefore, the logs should be copied only when other two methods have failed.

Preparing a pre-encrypted Linux VHD

```

Ubuntu 16
Configure encryption during distro install

```

1. Select "Configure encrypted volumes" when partitioning disks.

```
[!!] Partition disks

This is an overview of your currently configured partitions and mount points. Select a
partition to modify its settings (file system, mount point, etc.), a free space to create
partitions, or a device to initialize its partition table.

        Guided partitioning
        Configure software RAID
        Configure the Logical Volume Manager
Configure encrypted volumes
        Configure iSCSI volumes

SCSI3 (0,0,0) (sda) - 3.2 GB Msft Virtual Disk
    #1 primary 199.2 MB  B  f  ext4  /boot
    #2 primary   3.0 GB   f  ext4  /

        Undo changes to partitions
        Finish partitioning and write changes to disk

<Go Back>
```

<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons

2.Create a separate boot drive which must not be encrypted. Encrypt your root drive.

```
[!!] Partition disks

Please select the devices to be encrypted.

You can select one or more devices.

Devices to encrypt:

[ ] /dev/sda1          (199MB; ext4)
[*] /dev/sda2          (3019MB; ext4)

<Go Back>             <Continue>
```

<Tab> moves; <Space> selects; <Enter> activates buttons

3.Provide a passphrase. This is the passphrase that you will upload into KeyVault.

[!!] Partition disks

You need to choose a passphrase to encrypt SCSI3 (0,0,0), partition #2 (sda).

The overall strength of the encryption depends strongly on this passphrase, so you should take care to choose a passphrase that is not easy to guess. It should not be a word or sentence found in dictionaries, or a phrase that could be easily associated with you.

A good passphrase will contain a mixture of letters, numbers and punctuation. Passphrases are recommended to have a length of 20 or more characters.

There is no way to recover this passphrase if you lose it. To avoid losing data, you should normally write down the passphrase and keep it in a safe place separate from this computer.

Encryption passphrase:

contoso-password

[*] Show Password in Clear

<Go Back>

<Continue>

<Tab> moves; <Space> selects; <Enter> activates buttons

4.Finish partitioning.

[!!] Partition disks

This is an overview of your currently configured partitions and mount points. Select a partition to modify its settings (file system, mount point, etc.), a free space to create partitions, or a device to initialize its partition table.

Guided partitioning
Configure software RAID
Configure the Logical Volume Manager
Configure encrypted volumes
Configure iSCSI volumes

Encrypted volume (sda2_crypt) - 3.0 GB Linux device-mapper (crypt)
#1 3.0 GB f ext4 /
SCSI3 (0,0,0) (sda) - 3.2 GB Msft Virtual Disk
#1 primary 199.2 MB B F ext4 /boot
#2 primary 3.0 GB K crypto (sda2_crypt)

Undo changes to partitions
Finish partitioning and write changes to disk

<Go Back>

<F1> for help; <Tab> moves; <Space> selects; <Enter> activates buttons

5.When booting the VM, you will be asked for a passphrase. Use the passphrase you provided in step 3.

```

[ 1.129797] input: Microsoft Vmbus HID-compliant Mouse as /devices/0006:045E:0621.0001/input/input4
[ 1.132206] sda: sda1 sda2
[ 1.133217] hid 0006:045E:0621.0001: input: <UNKNOWN> HID v0.01 Mouse [Microsoft Vmbus HID-compliant Mouse] on
[ 1.134340] hv_netvsc: hv_netvsc channel opened successfully
[ 1.138418] sd 2:0:0:0: [sda] Attached SCSI disk
[ 1.265049] hv_netvsc vmbus_15: Send section size: 6144, Section count:2560
[ 1.266137] hv_netvsc vmbus_15: Device MAC 00:15:5d:05:34:01 link state up
[ 1.272596] scsi host3: storvsc_host_t
[ 1.436076] psmouse serio1: trackpoint: failed to get extended button data
Begin: Loading essential drivers ... [ 2.401782] md: linear personality registered for level -1
[ 2.404316] md: multipath personality registered for level -4
[ 2.407122] md: raid0 personality registered for level 0
[ 2.410610] md: raid1 personality registered for level 1
[ 2.480009] raid6: sse2x1 gen() 10995 MB/s
[ 2.548012] raid6: sse2x1 xor() 8467 MB/s
[ 2.616010] raid6: sse2x2 gen() 14312 MB/s
[ 2.684013] raid6: sse2x2 xor() 9555 MB/s
[ 2.752011] raid6: sse2x4 gen() 16205 MB/s
[ 2.820010] raid6: sse2x4 xor() 11594 MB/s
[ 2.888007] raid6: avx2x1 gen() 21995 MB/s
[ 2.956007] raid6: avx2x2 gen() 25959 MB/s
[ 3.024011] raid6: avx2x4 gen() 29505 MB/s
[ 3.024735] raid6: using algorithm avx2x4 gen() 29505 MB/s
[ 3.025038] raid6: using avx2x2 recovery algorithm
[ 3.027102] xor: automatically using best checksumming function:
[ 3.064003]     avx      : 35013.000 MB/sec
[ 3.065688] async_tx: api initialized (async)
[ 3.074685] md: raid6 personality registered for level 6
[ 3.075435] md: raid5 personality registered for level 5
[ 3.075746] md: raid4 personality registered for level 4
[ 3.079565] md: raid10 personality registered for level 10
done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... Begin: Running /scripts/local-top ... Please unlock disk sda2_crypt: -

```

6. Prepare VM for uploading into Azure using [these instructions](#). Do not run the last step (deprovisioning the VM) yet.

Configure encryption to work with Azure

1. Create a file under /usr/local/sbin/azure_crypt_key.sh, with the content in the script below. Pay attention to the KeyFileName, because it is the passphrase file name put by Azure.

```

#!/bin/sh
MountPoint=/tmp-keydisk-mount
KeyFileName=LinuxPassPhraseFileName
echo "Trying to get the key from disks ..." >&2
mkdir -p $MountPoint
modprobe vfat >/dev/null 2>&1
modprobe ntfs >/dev/null 2>&1
sleep 2
OPENED=0
cd /sys/block
for DEV in sd*; do
    echo "> Trying device: $DEV ..." >&2
    mount -t vfat -r /dev/${DEV}1 $MountPoint >/dev/null ||
    mount -t ntfs -r /dev/${DEV}1 $MountPoint >/dev/null
    if [ -f $MountPoint/$KeyFileName ]; then
        cat $MountPoint/$KeyFileName
        umount $MountPoint 2>/dev/null
        OPENED=1
        break
    fi
done

if [ $OPENED -eq 0 ]; then
    echo "FAILED to find suitable passphrase file ..." >&2
    echo -n "Try to enter your password: " >&2
    read -s -r A </dev/console
    echo -n "$A"
else
    echo "Success loading keyfile!" >&2
fi

```

2.Change the crypt config in */etc/crypttab*. It should look like this:

```
xxx_crypt uuid=xxxxxxxxxxxxxxxxxxxx none luks,discard,keyscript=/usr/local/sbin/azure_crypt_key.sh
```

3.If you are editing the *azure_crypt_key.sh* in Windows and copied it to Linux, do not forget to run *dos2unix /usr/local/sbin/azure_crypt_key.sh*.

4.Add executable permissions to the script:

```
chmod +x /usr/local/sbin/azure_crypt_key.sh
```

5.Edit */etc/initramfs-tools/modules* by appending lines:

```

vfat
ntfs
nls_cp437
nls_utf8
nls_iso8859-1

```

6.Run `update-initramfs -u -k all` to update the initramfs to make the `keyscript` take effect. 7.Now you can deprovision the VM.

```

root@ubuntu-preencrypted:~# ls -l /usr/local/sbin/azure_crypt_key.sh
-rwxr-xr-x 1 root root 860 Sep 18 16:57 /usr/local/sbin/azure_crypt_key.sh
root@ubuntu-preencrypted:~# cat /etc/crypttab
sda2_crypt UUID=b0dee704-1f2a-4f02-9a13-289c6c99dbb8 none luks,discard,keyscheme=/usr/local/sbin/azure_crypt_key.sh
root@ubuntu-preencrypted:~# cat /etc/initramfs-tools/modules
# List of modules that you want to include in your initramfs.
# They will be loaded at boot time in the order below.
#
# Syntax: module_name [args ...]
#
# You must run update-initramfs(8) to effect this change.
#
# Examples:
#
# raid1
# sd_mod
# vfat
# ntfs
# nls_cp437
# nls_utf8
# nls_iso8859-1
root@ubuntu-preencrypted:~# update-initramfs -u -k all
update-initramfs: Generating /boot/initrd.img-4.4.0-36-generic
W: plymouth: The plugin label.so is missing, the selected theme might not work as expected.
W: plymouth: You might want to install the plymouth-themes and plymouth-label package to fix this.
W: mdadm: /etc/mdadm/mdadm.conf defines no arrays.
[ 6289.960173] blk_update_request: I/O error, dev fd0, sector 0
update-initramfs: Generating /boot/initrd.img-4.4.0-21-generic
W: plymouth: The plugin label.so is missing, the selected theme might not work as expected.
W: plymouth: You might want to install the plymouth-themes and plymouth-label package to fix this.
W: mdadm: /etc/mdadm/mdadm.conf defines no arrays.
[ 6297.592236] blk_update_request: I/O error, dev fd0, sector 0
root@ubuntu-preencrypted:~# waagent -force -deprovision
WARNING! The waagent service will be stopped.
WARNING! Cached DHCP leases will be deleted.
WARNING! root password will be disabled. You will not be able to login as root.
WARNING! Nameserver configuration in /etc/resolvconf/resolv.conf.d/{tail,original} will be deleted.
2016/09/18 17:06:38.572398 INFO resolvconf is enabled: leaving /etc/resolv.conf intact
2016/09/18 17:06:38.572398 INFO resolvconf is enabled: leaving /etc/resolv.conf intact
root@ubuntu-preencrypted:~# export HISTSIZE=0
root@ubuntu-preencrypted:~# logout

```

8. Continue to next step and [upload your VHD](#) into Azure.

```

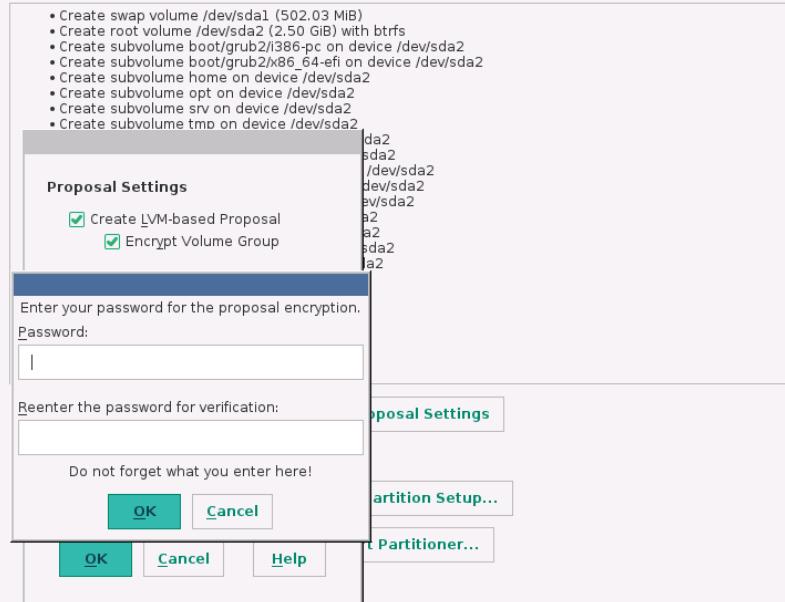
openSUSE 13.2
Configure encryption during distro install

```

1. Select "Encrypt Volume Group" when partitioning disks. Provide a passphrase. This is the passphrase that you will upload into KeyVault.



Suggested Partitioning



[Help](#) [Release Notes...](#)

[Abort](#) [Back](#) [Next](#)

2. Boot the VM using your passphrase.

```
[ 0.000000] tsc: Fast TSC calibration failed
[ OK ] Found device Virtual_Disk.
[ OK ] Found device Virtual_Disk.
      Starting Cryptography Setup for cr_scsi-14d53465420202020fd10f64360...278fd6327ec-part2...
      Starting Setup Virtual Console...
[ OK ] Started Setup Virtual Console.
      Starting Dispatch Password Requests to Console...
[ OK ] Started Dispatch Password Requests to Console.
Please enter passphrase for disk Virtual_Disk (cr_scsi-14d53465420202020fd10f64360f5f14797052278fd6327ec-part2)! *****
```

3. Prepare VM for uploading into Azure using [these instructions](#). Do not run the last step (deprovisioning the VM)

yet.

Configure encryption to work with Azure

1.Edit the /etc/dracut.conf and add the following line:

```
add_drivers+=" vfat ntfs nls_cp437 nls_iso8859-1"
```

2.Comment out these lines by the end of the file "/usr/lib/dracut/modules.d/90crypt/module-setup.sh":

```
# inst_multiple -o \
# $systemdutildir/system-generators/systemd-cryptsetup-generator \
# $systemdutildir/systemd-cryptsetup \
# $systemdsystemunitdir/systemd-ask-password-console.path \
# $systemdsystemunitdir/systemd-ask-password-console.service \
# $systemdsystemunitdir/cryptsetup.target \
# $systemdsystemunitdir/sysinit.target.wants/cryptsetup.target \
# systemd-ask-password systemd-tty-ask-password-agent \
# inst_script "$moddir"/crypt-run-generator.sh /sbin/crypt-run-generator
```

3.Append the following line at the beginning of the file "/usr/lib/dracut/modules.d/90crypt/parse-crypt.sh"

```
DRACUT_SYSTEMD=0
```

and change all occurrences of

```
if [ -z "$DRACUT_SYSTEMD" ]; then
```

to

```
if [ 1 ]; then
```

4.Edit /usr/lib/dracut/modules.d/90crypt/cryptroot-ask.sh and append this after the "# Open LUKS device"

```
MountPoint=/tmp-keydisk-mount
KeyFileName=LinuxPassPhraseFileName
echo "Trying to get the key from disks ..." >&2
mkdir -p $MountPoint >&2
modprobe vfat >/dev/null >&2
modprobe ntfs >/dev/null >&2
for SFS in /dev/sd*; do
echo "> Trying device:$SFS..." >&2
mount ${SFS}1 $MountPoint -t vfat -r >&2 ||
mount ${SFS}1 $MountPoint -t ntfs -r >&2
if [ -f $MountPoint/$KeyFileName ]; then
    echo "> keyfile got..." >&2
    cp $MountPoint/$KeyFileName /tmp-keyfile >&2
    luksfile=/tmp-keyfile
    umount $MountPoint >&2
    break
fi
done
```

5.Run the "/usr/sbin/dracut -f -v" to update the initrd.

6.Now you can deprovision the VM and [upload your VHD](#) into Azure.

CentOS 7
Configure encryption during distro install

1.Select "Encrypt my data" when partitioning disks.

INSTALLATION DESTINATION

CENTOS 7 INSTALLATION

Done **Help!**

Device Selection

Select the device(s) you'd like to install to. They will be left untouched until you click on the main menu's "Begin Installation" button.

Local Standard Disks

3072 MiB  Msft Virtual Disk sda / 3072 MiB free
--

Disks left unselected here will not be touched.

Specialized & Network Disks

 Add a disk...

Disks left unselected here will not be touched.

Other Storage Options

Partitioning

Automatically configure partitioning. I will configure partitioning.

I would like to make additional space available.

Encryption

Encrypt my data. You'll set a passphrase next.

[Full disk summary and boot loader...](#)

1 disk selected; 3072 MiB capacity; 3072 MiB free

2. Make sure "Encrypt" is selected for root partition.

MANUAL PARTITIONING

CENTOS 7 INSTALLATION

Done **Help!**

New CentOS 7 Installation

SYSTEM

/boot	190 MiB
sda1	
/ luks-sda2	2879 MiB >

luks-sda2

Mount Point: / **Device(s):** Msft Virtual Disk (sda)

Desired Capacity: 2879 MiB

Modify...

Device Type: Standard Partition Encrypt

File System: xfs Reformat

Label: **Name:** sda2

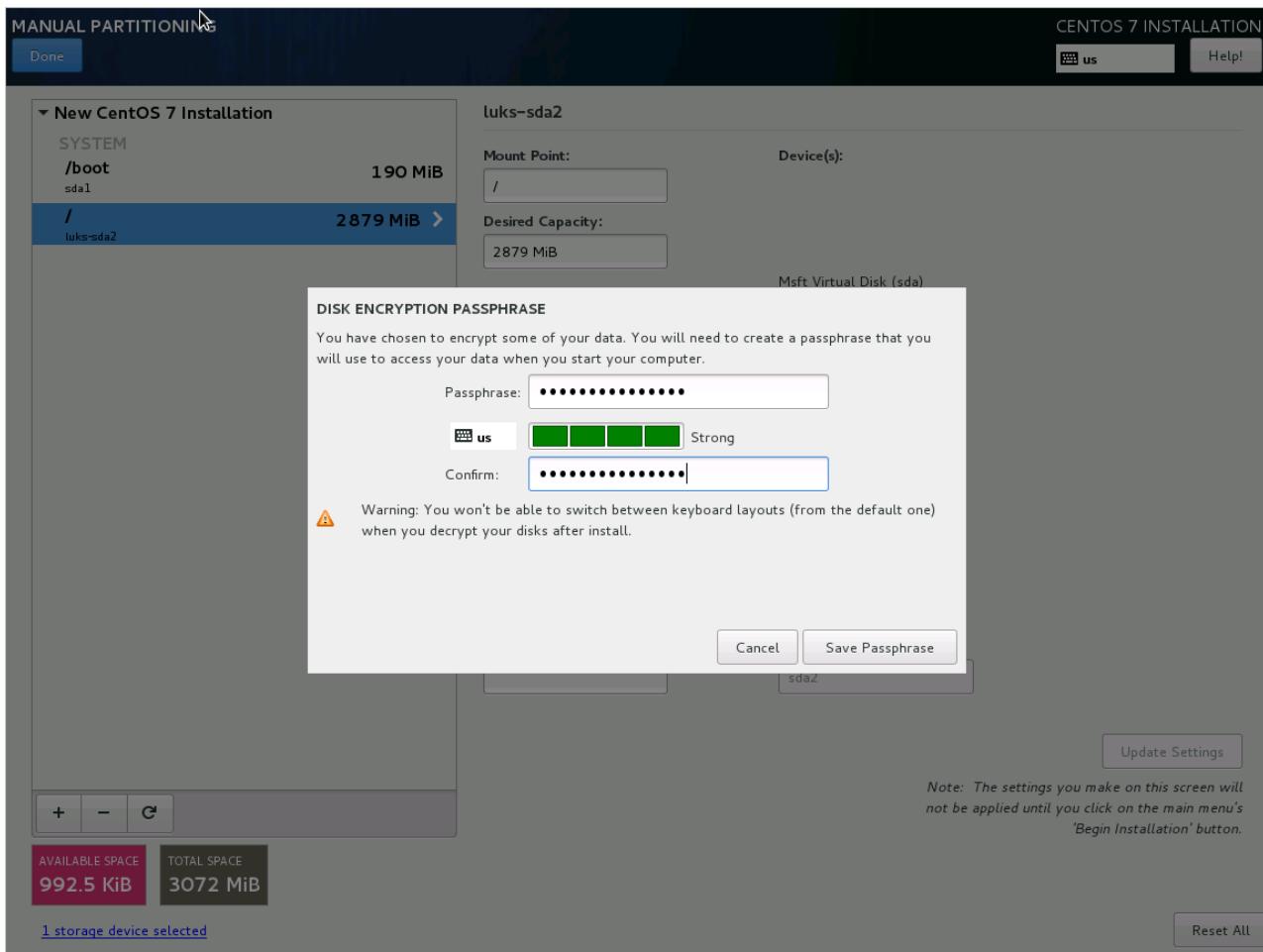
Update Settings

Note: The settings you make on this screen will not be applied until you click on the main menu's 'Begin Installation' button.

AVAILABLE SPACE 992.5 KiB **TOTAL SPACE** 3072 MiB

[1 storage device selected](#) **Reset All**

3. Provide a passphrase. This is the passphrase that you will upload into KeyVault.



4. When booting the VM, you will be asked for a passphrase. Use the passphrase you provided in step 3.



5. Prepare VM for uploading into Azure using [these instructions](#). Do not run the last step (deprovisioning the VM)

yet.

6.Now you can deprovision the VM and [upload your VHD](#) into Azure.

Configure encryption to work with Azure

1.Edit the /etc/dracut.conf and add the following line:

```
add_drivers+=" vfat ntfs nls_cp437 nls_iso8859-1"
```

2.Comment out these lines by the end of the file "/usr/lib/dracut/modules.d/90crypt/module-setup.sh":

```
#      inst_multiple -o \
#      $systemdutildir/system-generators/systemd-cryptsetup-generator \
#      $systemdutildir/systemd-cryptsetup \
#      $systemdsystemunitdir/systemd-ask-password-console.path \
#      $systemdsystemunitdir/systemd-ask-password-console.service \
#      $systemdsystemunitdir/cryptsetup.target \
#      $systemdsystemunitdir/sysinit.target.wants/cryptsetup.target \
#      systemd-ask-password systemd-tty-ask-password-agent
#      inst_script "$moddir"/crypt-run-generator.sh /sbin/crypt-run-generator
```

3.Append the following line at the beginning of the file "/usr/lib/dracut/modules.d/90crypt/parse-crypt.sh"

```
DRACUT_SYSTEMD=0
```

and change all occurrences of

```
if [ -z "$DRACUT_SYSTEMD" ]; then
```

to

```
if [ 1 ]; then
```

4.Edit /usr/lib/dracut/modules.d/90crypt/cryptroot-ask.sh and append this after the "# Open LUKS device"

```
MountPoint=/tmp-keydisk-mount
KeyFileName=LinuxPassPhraseFileName
echo "Trying to get the key from disks ..." >&2
mkdir -p $MountPoint >&2
modprobe vfat >/dev/null >&2
modprobe ntfs >/dev/null >&2
for SFS in /dev/sd*; do
echo "> Trying device:$SFS..." >&2
mount ${SFS}1 $MountPoint -t vfat -r >&2 ||
mount ${SFS}1 $MountPoint -t ntfs -r >&2
if [ -f $MountPoint/$KeyFileName ]; then
    echo "> keyfile got..." >&2
    cp $MountPoint/$KeyFileName /tmp-keyfile >&2
    luksfile=/tmp-keyfile
    umount $MountPoint >&2
    break
fi
done
```

5.Run the "/usr/sbin/dracut -f -v" to update the initrd.

```
[root@centos-preencrypted ~]# cat /etc/dracut.conf | grep add_drivers
add_drivers+=" vfat ntfs nls_cp437 nls_iso8859-1"
[root@centos-preencrypted ~]# cat /usr/lib/dracut/modules.d/90crypt/cryptroot-ask.sh | grep LinuxPassPhrasefileName -A 15 -B 1
MountPoint=/tmp-keydisk-mount
KeyFileName=LinuxPassPhraseFileName
echo "Trying to get the key from disks ..." >&2
mkdir -p $MountPoint >&2
modprobe vfat >/dev/null >&2
modprobe ntfs >/dev/null >&2
for SFS in /dev/sd*; do
echo "> Trying device:$SFS..." >&2
mount ${SFS}1 $MountPoint -t vfat -r >&2 ||
mount ${SFS}1 $MountPoint -t ntfs -r >&2
if [ ! -f $MountPoint/$KeyFileName ]; then
    echo "> keyfile got..." >&2
    cp $MountPoint/$KeyFileName /tmp-keyfile >&2
    luksfile=/tmp-keyfile
    umount $MountPoint >&2
    break
fi
[root@centos-preencrypted ~]# dracut -f -v_
```

Upload encrypted VHD to an Azure storage account

Once BitLocker encryption or DM-Crypt encryption is enabled, the local encrypted VHD needs to be uploaded to your storage account.

```
Add-AzureRmVhd [-Destination] <Uri> [-LocalFilePath] <FileInfo> [[-NumberOfUploaderThreads] <Int32> ] [[-BaseImageUriToPatch] <Uri> ] [[-OverWrite]] [ <CommonParameters>]
```

Upload disk encryption secret for the pre-encrypted VM to Key Vault

The disk encryption secret obtained previously needs to be uploaded as a secret in Key Vault. The Key Vault needs to have permissions enabled for your AAD client as well as disk encryption.

```
$AadClientId = "YourAADClientId"
$AadClientSecret = "YourAADClientSecret"

$keyVault = New-AzureRmKeyVault -VaultName $keyVaultName -ResourceGroupName $resourceGroupName -Location
$location

Set-AzureRmKeyVaultAccessPolicy -VaultName $keyVaultName -ResourceGroupName $resourceGroupName -
ServicePrincipalName $aadClientId -PermissionsToKeys all -PermissionsToSecrets all
Set-AzureRmKeyVaultAccessPolicy -VaultName $keyVaultName -ResourceGroupName $resourceGroupName -
EnabledForDiskEncryption
```

Disk encryption secret not encrypted with a KEK

Use [Set-AzureKeyVaultSecret](#) to provision the secret in key vault. In case of a Windows virtual machine, the bek file is encoded as a base64 string and then uploaded to key vault using the Set-AzureKeyVaultSecret cmdlet. For Linux, the passphrase is encoded as a base64 string and then uploaded to Key Vault. In addition, make sure that the following tags are set while creating the secret in key vault.

```

# This is the passphrase that was provided for encryption during distro install
$passphrase = "contoso-password"

$tags = @{"DiskEncryptionKeyEncryptionAlgorithm" = "RSA-OAEP"; "DiskEncryptionKeyFileName" =
"LinuxPassPhraseFileName"}
$secretName = [guid]::NewGuid().ToString()
$secretValue = [Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes($passphrase))
$secureSecretValue = ConvertTo-SecureString $secretValue -AsPlainText -Force

$secret = Set-AzureKeyVaultSecret -VaultName $KeyVaultName -Name $secretName -SecretValue $secureSecretValue -
tags $tags
$secretUrl = $secret.Id

```

The `$secretUrl` shall be used in the next step for [attaching the OS disk without using KEK](#).

Disk encryption secret encrypted with a KEK

The secret can optionally be encrypted with a Key Encryption Key before uploading to Key vault. Use the [wrap API](#) to first encrypt the secret using the Key Encryption Key. The output of this wrap operation is a base64 URL encoded string which is then uploaded as a secret using the [Set-AzureKeyVaultSecret](#) cmdlet.

```

# This is the passphrase that was provided for encryption during distro install
$passphrase = "contoso-password"

Add-AzureKeyVaultKey -VaultName $KeyVaultName -Name "keyencryptionkey" -Destination Software
$keyEncryptionKey = Get-AzureKeyVaultKey -VaultName $KeyVault.OriginalVault.Name -Name "keyencryptionkey"

$apiversion = "2015-06-01"

#####
# Get Auth URI
#####

$uri = $KeyVault.VaultUri + "/keys"
$headers = @{}

$response = try { Invoke-RestMethod -Method GET -Uri $uri -Headers $headers } catch { $_.Exception.Response }

$authHeader = $response.Headers["www-authenticate"]
$authUri = [regex]::match($authHeader, 'authorization="(.*)"').Groups[1].Value

Write-Host "Got Auth URI successfully"

#####
# Get Auth Token
#####

$uri = $authUri + "/oauth2/token"
$body = "grant_type=client_credentials"
$body += "&client_id=" + $AadClientId
$body += "&client_secret=" + [Uri]::EscapeDataString($AadClientSecret)
$body += "&resource=" + [Uri]::EscapeDataString("https://vault.azure.net")
$headers = @{}

$response = Invoke-RestMethod -Method POST -Uri $uri -Headers $headers -Body $body

$access_token = $response.access_token

Write-Host "Got Auth Token successfully"

#####
# Get KEK info
#####

$uri = $KeyEncryptionKey.Id + "?api-version=" + $apiversion
$headers = @{"Authorization" = "Bearer " + $access_token}

```

```

$response = Invoke-RestMethod -Method GET -Uri $uri -Headers $headers

$keyid = $response.key.kid

Write-Host "Got KEK info successfully"

#####
# Encrypt passphrase using KEK
#####

$passphraseB64 = [Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes($Passphrase))
$uri = $keyid + "/encrypt?api-version=" + $apiversion
$headers = @{"Authorization" = "Bearer " + $access_token; "Content-Type" = "application/json"}
$bodyObj = @{"alg" = "RSA-OAEP"; "value" = $passphraseB64}
$body = $bodyObj | ConvertTo-Json

$response = Invoke-RestMethod -Method POST -Uri $uri -Headers $headers -Body $body

$wrappedSecret = $response.value

Write-Host "Encrypted passphrase successfully"

#####
# Store secret
#####

$secretName = [guid]::NewGuid().ToString()
$uri = $KeyVault.VaultUri + "/secrets/" + $secretName + "?api-version=" + $apiversion
$secretAttributes = @{"enabled" = $true}
$secretTags = @{"DiskEncryptionKeyEncryptionAlgorithm" = "RSA-OAEP"; "DiskEncryptionKeyFileName" =
"LinuxPassPhraseFileName"}
$headers = @{"Authorization" = "Bearer " + $access_token; "Content-Type" = "application/json"}
$bodyObj = @{"value" = $wrappedSecret; "attributes" = $secretAttributes; "tags" = $secretTags}
$body = $bodyObj | ConvertTo-Json

$response = Invoke-RestMethod -Method PUT -Uri $uri -Headers $headers -Body $body

Write-Host "Stored secret successfully"

$secretUrl = $response.id

```

The `$keyEncryptionKey` and `$secretUrl` shall be used in the next step for [attaching the OS disk using KEK](#).

Specify secret URL when attaching OS Disk

Without using a KEK

While attaching the OS disk, `$secretUrl` needs to be passed. The URL was generated in the section "[disk encryption secret not encrypted with a KEK](#)".

```

Set-AzureRmVMOSDisk ` 
    -VM $VirtualMachine ` 
    -Name $OSDiskName ` 
    -SourceImageUri $VhdUri ` 
    -VhdUri $OSDiskUri ` 
    -Linux ` 
    -CreateOption FromImage ` 
    -DiskEncryptionKeyVaultId $KeyVault.ResourceId ` 
    -DiskEncryptionKeyUrl $SecretUrl

```

Using a KEK

While attaching the OS disk, `$keyEncryptionKey` and `$secretUrl` need to be passed. The URL was generated in the section "[disk encryption secret encrypted with a KEK](#)".

```
Set-AzureRmVMOSDisk ` 
    -VM $VirtualMachine ` 
    -Name $OSDiskName ` 
    -SourceImageUri $CopiedTemplateBlobUri ` 
    -VhdUri $OSDiskUri ` 
    -Linux ` 
    -CreateOption FromImage ` 
    -DiskEncryptionKeyVaultId $KeyVault.ResourceId ` 
    -DiskEncryptionKeyUrl $SecretUrl ` 
    -KeyEncryptionKeyVaultId $KeyVault.ResourceId ` 
    -KeyEncryptionKeyURL $KeyEncryptionKey.Id
```

Download this Guide

You can download this guide from the [TechNet Gallery](#).

For more information

[Explore Azure Disk Encryption with Azure PowerShell](#)

[Explore Azure Disk Encryption with Azure PowerShell - Part 2](#)

Common network Azure PowerShell commands for VMs

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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.

Create network resources

TASK	COMMAND
Create subnet configurations	\$subnet1 = New-AzureRmVirtualNetworkSubnetConfig -Name "subnet_name" -AddressPrefix XX.X.X.X/XX \$subnet2 = New-AzureRmVirtualNetworkSubnetConfig -Name "subnet_name" -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 "virtual_network_name" -ResourceGroupName "resource_group_name" -Location "location_name" -AddressPrefix XX.X.X.X/XX -Subnet \$subnet1, \$subnet2
Test for a unique domain name	Test-AzureRmDnsAvailability -DomainQualifiedNames "domain_name" -Location "location_name" 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 "ip_address_name" -ResourceGroupName "resource_group_name" -DomainNameLabel "domain_name" -Location "location_name" -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	\$frontendIP = New-AzureRmLoadBalancerFrontendIpConfig -Name "frontend_ip_name" -PublicIpAddress \$pip The frontend configuration includes the public IP address that you previously created for incoming network traffic.
Create a backend address pool	\$beAddressPool = New-AzureRmLoadBalancerBackendAddressPoolConfig -Name "backend_pool_name" Provides internal addresses for the backend of the load balancer that are accessed through a network interface.
Create a probe	\$healthProbe = New-AzureRmLoadBalancerProbeConfig -Name "probe_name" -RequestPath 'HealthProbe.aspx' -Protocol http -Port 80 -IntervalInSeconds 15 -ProbeCount 2 Contains health probes used to check availability of virtual machines instances in the backend address pool.
Create a load balancing rule	\$lbRule = New-AzureRmLoadBalancerRuleConfig -Name HTTP -FrontendIpConfiguration \$frontendIP -BackendAddressPool \$beAddressPool -Probe \$healthProbe -Protocol Tcp -FrontendPort 80 -BackendPort 80 Contains rules that assign a public port on the load balancer to a port in the backend address pool.
Create an inbound NAT rule	\$inboundNATRule = New-AzureRmLoadBalancerInboundNatRuleConfig -Name "rule_name" -FrontendIpConfiguration \$frontendIP -Protocol TCP -FrontendPort 3441 -BackendPort 3389 Contains rules mapping a public port on the load balancer to a port for a specific virtual machine in the backend address pool.
Create a load balancer	\$loadBalancer = New-AzureRmLoadBalancer -ResourceGroupName "resource_group_name" -Name "load_balancer_name" -Location "location_name" -FrontendIpConfiguration \$frontendIP -InboundNatRule \$inboundNATRule -LoadBalancingRule \$lbRule -BackendAddressPool \$beAddressPool -Probe \$healthProbe
Create a network interface	\$nic1 = New-AzureRmNetworkInterface -ResourceGroupName "resource_group_name" -Name "network_interface_name" -Location "location_name" -PrivateIpAddress XX.X.X.X -Subnet subnet2 -LoadBalancerBackendAddressPool \$loadBalancer.BackendAddressPools[0] -LoadBalancerInboundNatRule \$loadBalancer.InboundNatRules[0] Create a network interface using the public IP address and virtual network subnet that you previously created.

Get information about network resources

TASK	COMMAND
List virtual networks	<code>Get-AzureRmVirtualNetwork -ResourceGroupName "resource_group_name"</code> Lists all the virtual networks in the resource group.
Get information about a virtual network	<code>Get-AzureRmVirtualNetwork -Name "virtual_network_name" -ResourceGroupName "resource_group_name"</code>
List subnets in a virtual network	<code>Get-AzureRmVirtualNetwork -Name "virtual_network_name" -ResourceGroupName "resource_group_name" Select Subnets</code>
Get information about a subnet	<code>Get-AzureRmVirtualNetworkSubnetConfig -Name "subnet_name" -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 "resource_group_name"</code> Lists the public IP addresses in the resource group.
List load balancers	<code>Get-AzureRmLoadBalancer -ResourceGroupName "resource_group_name"</code> Lists all the load balancers in the resource group.
List network interfaces	<code>Get-AzureRmNetworkInterface -ResourceGroupName "resource_group_name"</code> Lists all the network interfaces in the resource group.
Get information about a network interface	<code>Get-AzureRmNetworkInterface -Name "network_interface_name" -ResourceGroupName "resource_group_name"</code> Gets information about a specific network interface.
Get the IP configuration of a network interface	<code>Get-AzureRmNetworkInterfaceIPConfig -Name "ipconfiguration_name" -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 "subnet_name" -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 "virtual_network_name" -ResourceGroupName "resource_group_name"</pre> <p>Removes the specified virtual network from the resource group.</p>
Delete a network interface	<pre>Remove-AzureRmNetworkInterface -Name "network_interface_name" -ResourceGroupName "resource_group_name"</pre> <p>Removes the specified network interface from the resource group.</p>
Delete a load balancer	<pre>Remove-AzureRmLoadBalancer -Name "load_balancer_name" -ResourceGroupName "resource_group_name"</pre> <p>Removes the specified load balancer from the resource group.</p>
Delete a public IP address	<pre>Remove-AzureRmPublicIpAddress -Name "ip_address_name" -ResourceGroupName "resource_group_name"</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](#).

Opening ports to a VM in Azure using the Azure portal

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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, click **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's a 'Subscription name' field with 'No deployments' and a 'Last deployment' status. Below that is a 'Filter items...' input field. A table at the bottom lists resources by Name, Type, and Location. The first row in the table is for a 'Network security group'. To the right of the table is a 'Results' pane with a 'NAME' section containing a single item: 'Network security group'. Both the 'Add' button and the 'Network security group' entry in the results pane are highlighted with red boxes.

Enter a name for your Network Security Group, select or create a resource group, and select a location. Click **Create** when finished:

Create network securit...

*** Name**
myNetworkSecurityGroup 

*** Subscription**
Visual Studio Ultimate with MSDN

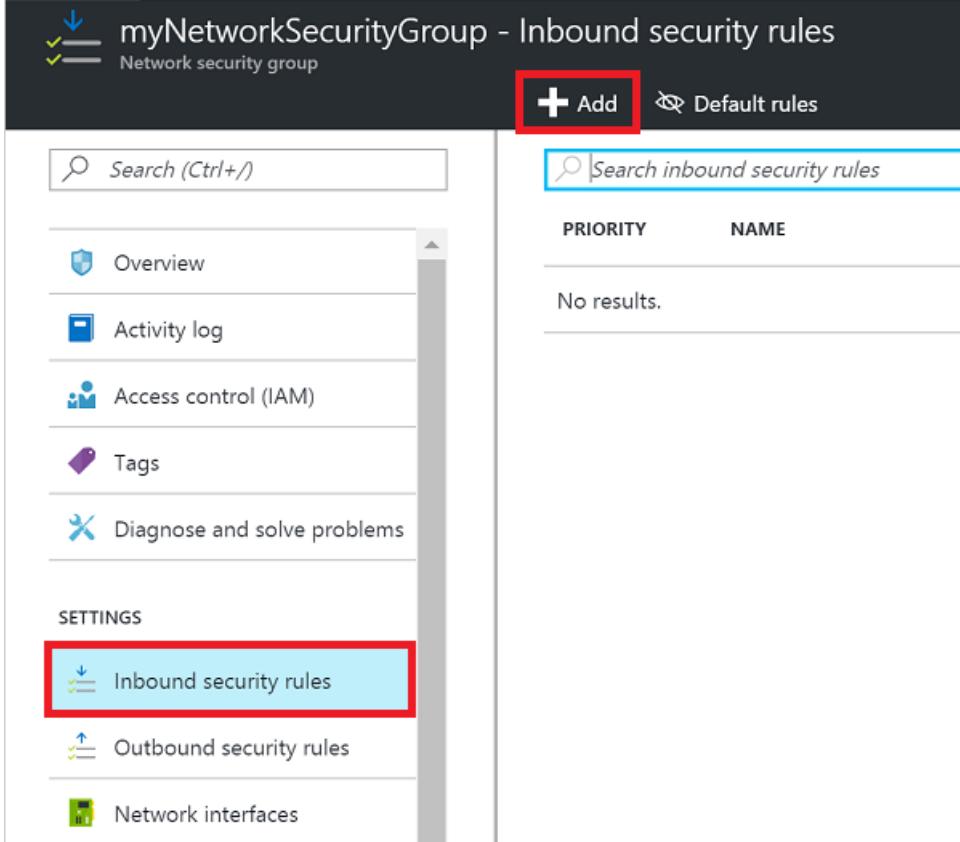
*** Resource group** 
 Create new Use existing
myResourceGroup

*** Location**
West US

Pin to dashboard

Create [Automation options](#)

Select your new Network Security Group. Select 'Inbound security rules', then click the **Add** button to create a rule:



PRIORITY	NAME
No results.	

Provide a name for your new rule. Port 80 is already entered by default. This blade is where you would change the source, protocol, and destination when adding additional rules to your Network Security Group. Click **OK** to create the rule:

Add inbound security r... — X

myNetworkSecurityGroup

Advanced

* Name
AllowHTTP ✓

* Priority ⓘ
100

* Source ⓘ
 Any CIDR block Tag

Service ⓘ
Custom

* Protocol
 Any TCP UDP

* Port range ⓘ
80

* Action
 Deny Allow

OK

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 click **Associate**:

The screenshot shows the 'Subnets' section of the 'myNetworkSecurityGroup' page in the Azure portal. The left sidebar includes links for 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 area has a search bar and a table with columns for NAME and ADDRESS RANGE, showing 'No results.' The '+ Associate' button is also highlighted with a red box.

Select your virtual network, and then select the appropriate subnet:

The screenshot shows the 'Associate subnet' dialog. Step 1 shows a list of virtual networks, with 'myVnet' selected. Step 2 shows a list of subnets under 'myVnet', with 'mySubnet' selected. Both steps are highlighted with red boxes.

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](#).

You can define Network Security Groups and ACL rules as part of Azure Resource Manager templates. Read more about [creating Network Security Groups with templates](#).

If you need to use port-forwarding to map a unique external port to an internal port on your VM, use a load balancer and Network Address Translation (NAT) rules. For example, you may want to expose TCP port 8080 externally and have traffic directed to TCP port 80 on a VM. You can learn about [creating an Internet-facing load balancer](#).

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](#)

Opening ports and endpoints to a VM in Azure using PowerShell

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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 example parameter names with your own values. Example parameter names included `myResourceGroup`, `myNetworkSecurityGroup`, and `myVnet`.

Create a rule. 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 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 "WestUS" -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`:

```
$vnet = Get-AzureRmVirtualNetwork -ResourceGroupName "myResourceGroup" `  
-Name "myVnet"
```

Associate your Network Security Group with your subnet. 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 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](#).

You can define Network Security Groups and ACL rules as part of Azure Resource Manager templates. Read more about [creating Network Security Groups with templates](#).

If you need to use port-forwarding to map a unique external port to an internal port on your VM, use a load balancer and Network Address Translation (NAT) rules. For example, you may want to expose TCP port 8080 externally and have traffic directed to TCP port 80 on a VM. You can learn about [creating an Internet-facing load balancer](#).

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 Fully Qualified Domain Name in the Azure portal

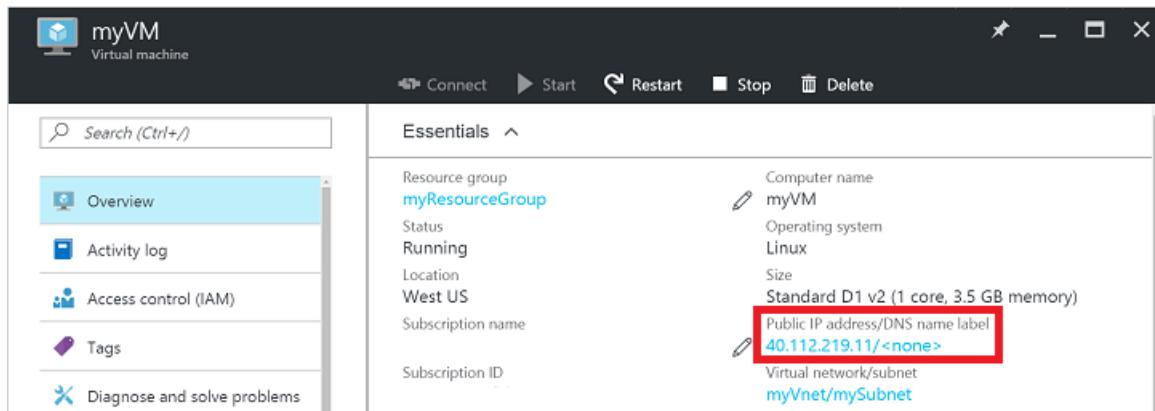
1/17/2017 • 1 min to read • [Edit on GitHub](#)

When you create a virtual machine (VM) in the [Azure portal](#) using the Resource Manager deployment model, 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, by default, you can create one once the VM is created. This article demonstrates the steps to create a DNS name or FQDN.

Quick steps

The article assumes that you have logged in to your subscription in the portal, and created a virtual machine with the available images using the Resource Manager deployment model. Follow these steps once your virtual machine starts running.

1. Select your virtual machine in the portal. The DNS name is blank. Click **Public IP address**:



2. Enter the desired DNS name label and then click **Save**.

The screenshot shows the 'myPublicIP - Configuration' blade in the Azure portal. On the left, there's a navigation menu with 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Configuration' (which is selected and highlighted in blue), 'Properties', 'Locks', and 'Automation script'. On the right, under 'Assignment', the 'Dynamic' tab is selected. It shows the 'IP address' as 40.112.219.11 and the 'Idle timeout (minutes)' set to 4. Below that, the 'DNS name label (optional)' field contains 'mydns' with a green checkmark, and the full FQDN '.westus.cloudapp.azure.com' is displayed.

The Public IP resource now shows this new DNS label on its blade.

3. Close the Public IP blades and go back to the VM overview blade in the portal. After a few seconds, the portal should update your settings. Verify that the DNS name/FQDN appears next to the IP address for the **Public IP address** resource.

The screenshot shows the 'myVM' blade in the Azure portal. The left sidebar includes 'Overview' (selected and highlighted in blue), 'Activity log', 'Access control (IAM)', 'Tags', and 'Diagnose and solve problems'. The main area displays 'Essentials' information for the VM, including 'Resource group: myResourceGroup', 'Status: Running', 'Location: ---', 'Subscription name: ---', and 'Subscription ID: ---'. To the right, detailed VM properties are listed: 'Computer name: myVM', 'Operating system: Linux', 'Size: Standard D1 v2 (1 core, 3.5 GB memory)', 'Public IP address/DNS name label: 40.112.219.11/mydns.westus.cloudapp.azure.com', and 'Virtual network/subnet: myVnet/mySubnet'. The 'Public IP address/DNS name label' field is highlighted with a red box.

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.

Creating a Windows VM with multiple NICs

1/17/2017 • 3 min to read • [Edit on GitHub](#)

You can create a virtual machine (VM) in Azure that has multiple virtual network interfaces (NICs) attached to it. A common scenario would be to have different subnets for front-end and back-end connectivity, or a network dedicated to a monitoring or backup solution. This article provides quick commands to create a VM with multiple NICs attached to it. For detailed information, including how to create multiple NICs within your own PowerShell scripts, read more about [deploying multi-NIC VMs](#). Different [VM sizes](#) support a varying number of NICs, so size your VM accordingly.

WARNING

You must attach multiple NICs when you create a VM - you cannot add NICs to an existing VM. You can [create a VM based on the original virtual disk\(s\)](#) and create multiple NICs as you deploy the VM.

Create core resources

Make sure that you have the [latest Azure PowerShell installed and configured](#). Log in to your Azure account:

```
Login-AzureRmAccount
```

In the following examples, replace example parameter names with your own values. Example parameter names included `myResourceGroup`, `mystorageaccount`, and `myVM`.

First, create a resource group. The following example creates a resource group named `myResourceGroup` in the `WestUS` location:

```
New-AzureRmResourceGroup -Name "myResourceGroup" -Location "WestUS"
```

Create a storage account to hold your VMs. The following example creates a storage account named `mystorageaccount`:

```
$storageAcc = New-AzureRmStorageAccount -ResourceGroupName "myResourceGroup" `  
-Location "WestUS" -Name "mystorageaccount" `  
-Kind "Storage" -SkuName "Premium_LRS"
```

Create virtual network and subnets

Define two virtual network subnets - one for front-end traffic and one for back-end traffic. The following example defines two subnets, named `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"
```

Create your virtual network and subnets. The following example creates a virtual network named `myVnet`:

```
$myVnet = New-AzureRmVirtualNetwork -ResourceGroupName "myResourceGroup" `  
    -Location "WestUS" -Name "myVnet" -AddressPrefix "192.168.0.0/16" `  
    -Subnet $mySubnetFrontEnd,$mySubnetBackEnd
```

Create multiple NICs

Create two NICs, attaching one NIC to the front-end subnet and one NIC to the back-end subnet. The following example creates two NICs, named `myNic1` and `myNic2`:

```
$frontEnd = $myVnet.Subnets|?{$_ .Name -eq 'mySubnetFrontEnd'}  
$myNic1 = New-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" `  
    -Location "WestUS" -Name "myNic1" -SubnetId $frontEnd.Id  
  
$backEnd = $myVnet.Subnets|?{$_ .Name -eq 'mySubnetBackEnd'}  
$myNic2 = New-AzureRmNetworkInterface -ResourceGroupName "myResourceGroup" `  
    -Location "WestUS" -Name "myNic2" -SubnetId $backEnd.Id
```

Typically you also create a [network security group](#) or [load balancer](#) to help manage and distribute traffic across your VMs. The [more detailed multi-NIC VM](#) article guides you through creating a Network Security Group and assigning NICs.

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. Read more about [Windows VM sizes](#).

First, set your VM credentials to the `$cred` variable as follows:

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

The following example defines a VM named `myVM` and uses a VM size that supports up to two NICs (`Standard_DS2_v2`):

```
$vmConfig = New-AzureRmVMConfig -VMName "myVM" -VMSize "Standard_DS2_v2"
```

Create the rest of your VM config. The following example creates a Windows Server 2012 R2 VM:

```
$vmConfig = Set-AzureRmVMOperatingSystem -VM $vmConfig -Windows -ComputerName "myVM" `  
    -Credential $cred -ProvisionVMAgent -EnableAutoUpdate  
$vmConfig = Set-AzureRmVMSourceImage -VM $vmConfig -PublisherName "MicrosoftWindowsServer" `  
    -Offer "WindowsServer" -Skus "2012-R2-Datacenter" -Version "latest"
```

Attach the two NICs you previously created:

```
$vmConfig = Add-AzureRmVMNetworkInterface -VM $vmConfig -Id $myNic1.Id -Primary  
$vmConfig = Add-AzureRmVMNetworkInterface -VM $vmConfig -Id $myNic2.Id
```

Configure the storage and virtual disk for your new VM:

```
$blobPath = "vhds/WindowsVMosDisk.vhd"  
$osDiskUri = $storageAcc.PrimaryEndpoints.Blob.ToString() + $blobPath  
$diskName = "windowsvmosdisk"  
$vmConfig = Set-AzureRmVMOSDisk -VM $vmConfig -Name $diskName -VhdUri $osDiskUri `  
    -CreateOption "fromImage"
```

Finally, create a VM:

```
New-AzureRmVM -VM $vmConfig -ResourceGroupName "myResourceGroup" -Location "WestUS"
```

Creating multiple NICs using Resource Manager templates

Azure Resource Manager templates use declarative JSON files to define your environment. You can read an [overview of Azure Resource Manager](#). Resource Manager templates provide a way to create multiple instances of a resource during deployment, such as creating multiple NICs. You use `copy` to specify the number of instances to create:

```
"copy": {  
    "name": "multiplenics"  
    "count": "[parameters('count')]"  
}
```

Read more about [creating multiple instances using `copy`](#).

You can also use a `copyIndex()` to then append a number to a resource name, which allows you to create `myNic1`, `MyNic2`, etc. The following shows an example of appending the index value:

```
"name": "[concat('myNic', copyIndex())]",
```

You can read a complete example of [creating multiple NICs using Resource Manager templates](#).

Next steps

Make sure to review [Windows VM sizes](#) when trying to creating a VM with multiple NICs. Pay attention to the maximum number of NICs each VM size supports.

Remember that you cannot add additional NICs to an existing VM, you must create all the NICs when you deploy the VM. Take care when planning your deployments to make sure that you have all the required network connectivity from the outset.

Create a virtual network using the Azure portal

1/17/2017 • 3 min to read • [Edit on GitHub](#)

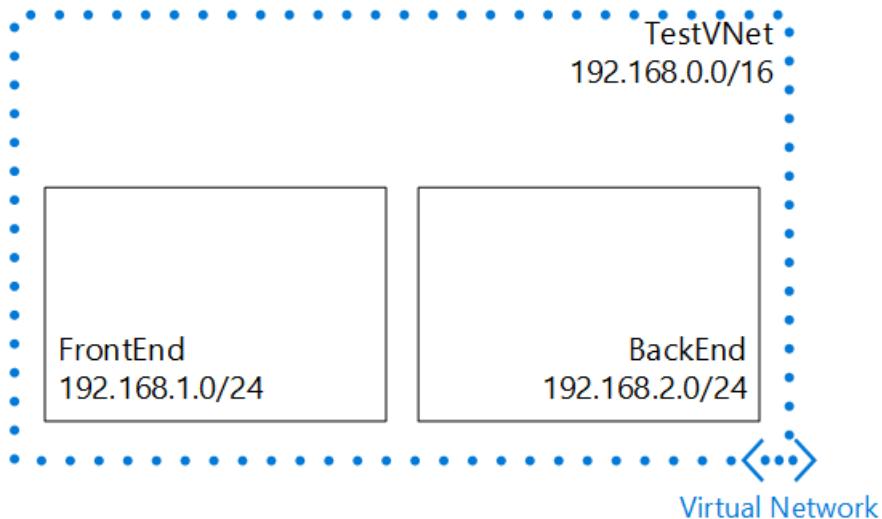
An Azure virtual network (VNet) is a representation of your own network in the cloud. You can control your Azure network settings and define DHCP address blocks, DNS settings, security policies, and routing. You can also further segment your VNet into subnets and deploy Azure IaaS virtual machines (VMs) and PaaS role instances, in the same way you can deploy physical and virtual machines to your on-premises datacenter. In essence, you can expand your network to Azure, bringing your own IP address blocks. Read the [virtual network overview](#) if you are not familiar with VNets.

Azure has two deployment models: Azure Resource Manager and classic. Microsoft recommends creating resources through the Resource Manager deployment model. To learn more about the differences between the two models, read the [Understand Azure deployment models](#) article.

This article explains how to create a VNet through the Resource Manager deployment model using the Azure portal. You can also create a VNet through Resource Manager using other tools or create a VNet through the classic deployment model by selecting a different option from the following list:

Scenario

To better illustrate how to create a VNet and subnets, this document will use the scenario below.



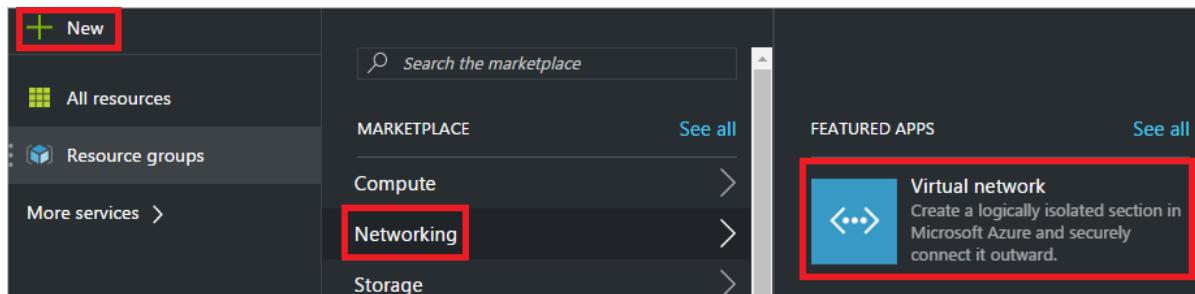
In this scenario you will create a VNet named **TestVNet** with a reserved CIDR block of **192.168.0.0/16**. Your VNet will contain the following subnets:

- **FrontEnd**, using **192.168.1.0/24** as its CIDR block.
- **BackEnd**, using **192.168.2.0/24** as its CIDR block.

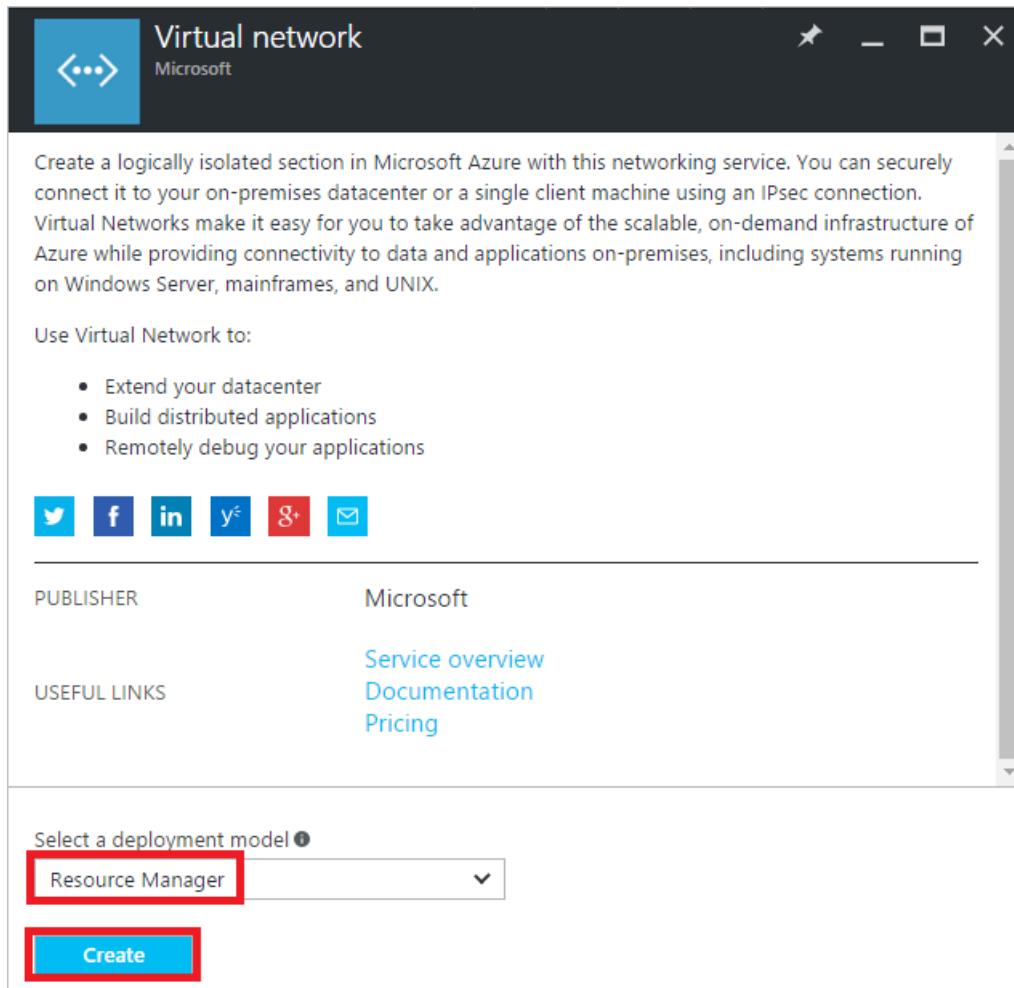
Create a virtual network

To create a virtual network using the Azure portal, complete the following steps:

1. From a browser, navigate to <http://portal.azure.com> and, if necessary, sign in with your Azure account.
2. Click **New > Networking > Virtual network**, as shown in the following picture:



3. In the **Virtual network** blade that appears, ensure that **Resource Manager** is selected and click **Create**, as shown in the following picture:



4. In the **Create virtual network** blade that appeared, enter **TestVNet** for **Name**, **192.168.0.0/16** for **Address space**, **FrontEnd** for **Subnet name** **192.168.1.0/24** for **Subnet address range**, **TestRG** for **Resource group**, select your **Subscription**, a **Location** and click the **Create** button, as shown in the following picture:

Create virtual network

* Name
TestVNet

* Address space
192.168.0.0/16
192.168.0.0 - 192.168.255.255 (65536 addresses)

* Subnet name
FrontEnd

* Subnet address range
192.168.1.0/24
192.168.1.0 - 192.168.1.255 (256 addresses)

* Subscription
▼

* Resource group
Create new Use existing
TestRG

* Location
West US

Pin to dashboard

Create [Automation options](#)

Alternatively, you can select an existing resource group. To learn more about resource groups, read the [Resource Manager overview](#) article. You can also select a different location. To learn more about Azure locations and regions, read the [Azure regions](#) article.

- The portal only enables you to create one subnet when creating a VNet. For this scenario, a second subnet must be created after the VNet is created. To create the second subnet, click **All resources**, then click **TestVNet** in the **All resources** blade, as shown in the following picture:

NAME	RESOURCE GROUP	LOCATION	TYPE
TestVNet	TestRG	West US	Virtual network

- In the **TestVNet** blade that appears, click **Subnet**, then click **+Subnet**, enter *BackEnd* for **Name**, 192.168.2.0/24 for **Address range** in the **Add subnet** blade, then click **OK**, as shown in the following picture:

The screenshot shows the Azure portal interface for managing subnets. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, and Subnets (which is highlighted with a red box). The main area shows a table of existing subnets: FrontEnd (192.168.1.0/24). To the right, a modal window titled 'Add subnet' is open, also with 'Subnet' selected in its tabs. It contains fields for Name ('BackEnd') and Address range ('192.168.2.0/24'), along with Network security group and Route table settings. A large red box highlights the 'OK' button at the bottom right of the modal.

7. The two subnets are listed, as shown in the following picture:

This screenshot shows the list of subnets in the Azure portal. The table has columns: NAME, ADDRESS RANGE, AVAILABLE ADDR..., and SECURITY GROUP. It lists two subnets: 'FrontEnd' with address range '192.168.1.0/24' and 'BackEnd' with address range '192.168.2.0/24'. Both have 251 available addresses and no security group assigned.

NAME	ADDRESS RANGE	AVAILABLE ADDR...	SECURITY GROUP
FrontEnd	192.168.1.0/24	251	-
BackEnd	192.168.2.0/24	251	-

This article explained how to create a virtual network with two subnets for testing. Before creating a virtual network for production use, we recommend reading the [Virtual network overview](#) and [Virtual network plan and design](#) articles to fully understand virtual networks and all settings.

Next steps

Learn how to connect:

- A virtual machine (VM) to a virtual network by reading the [Create a Windows VM](#) or [Create a Linux VM](#) articles. Instead of creating a VNet and subnet in the steps of the articles, you can select an existing VNet and subnet to connect a VM to.
- The virtual network to other virtual networks by reading the [Connect VNets](#) article.
- The virtual network to an on-premises network using a site-to-site virtual private network (VPN) or ExpressRoute circuit. Learn how by reading the [Connect a VNet to an on-premises network using a site-to-site VPN](#) and [Link a VNet to an ExpressRoute circuit](#) articles.

How to manage NSGs using the Azure portal

1/17/2017 • 3 min to read • [Edit on GitHub](#)

You can use an NSG to control traffic to one or more virtual machines (VMs), role instances, network adapters (NICs), or subnets in your virtual network. An NSG contains access control rules that allow or deny traffic based on traffic direction, protocol, source address and port, and destination address and port. The rules of an NSG can be changed at any time, and changes are applied to all associated instances.

For more information about NSGs, visit [what is an NSG](#).

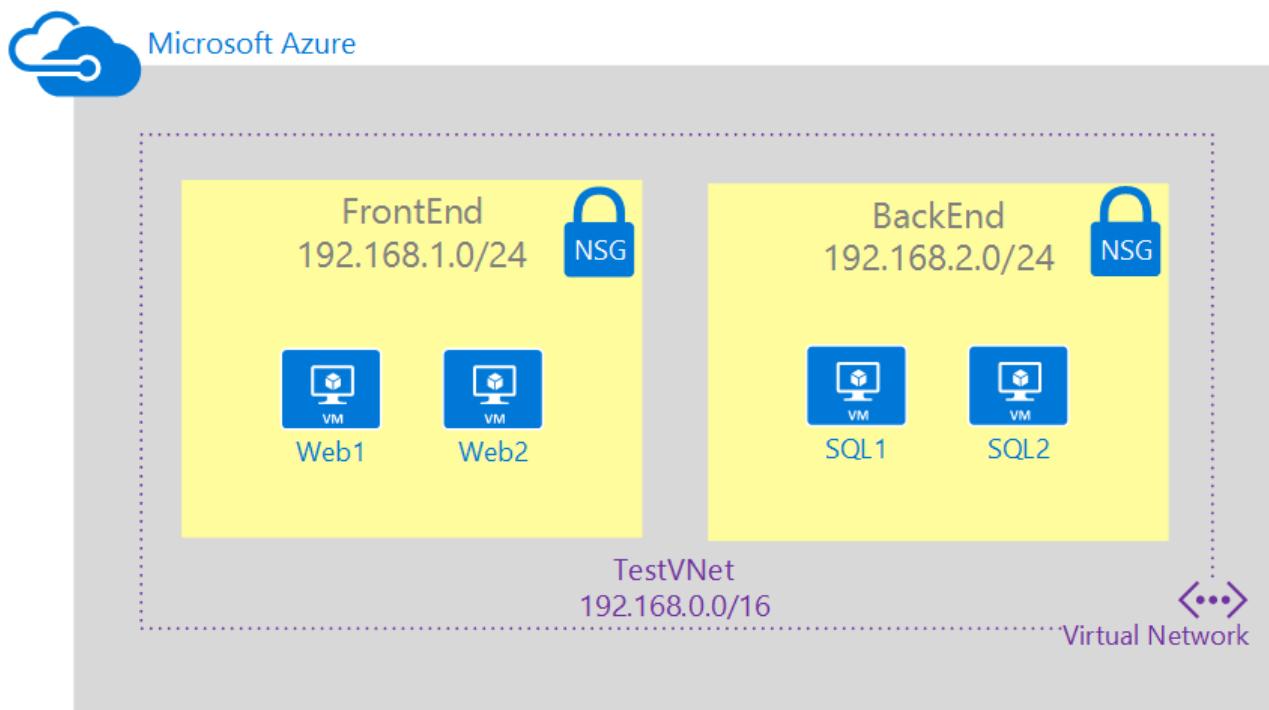
IMPORTANT

Before you work with Azure resources, it's important to understand that Azure currently has two deployment models: Azure Resource Manager and classic. Make sure you understand [deployment models and tools](#) before you work with any Azure resource. You can view the documentation for different tools by clicking the tabs at the top of this article.

This article covers the Resource Manager deployment model. You can also [create NSGs in the classic deployment model](#).

Scenario

To better illustrate how to create NSGs, this document will use the scenario below.



In this scenario you will create an NSG for each subnet in the **TestVNet** virtual network, as described below:

- **NSG-FrontEnd**. The front end NSG will be applied to the *FrontEnd* subnet, and contain two rules:
 - **rdp-rule**. This rule will allow RDP traffic to the *FrontEnd* subnet.
 - **web-rule**. This rule will allow HTTP traffic to the *FrontEnd* subnet.
- **NSG-BackEnd**. The back end NSG will be applied to the *BackEnd* subnet, and contain two rules:
 - **sql-rule**. This rule allows SQL traffic only from the *FrontEnd* subnet.
 - **web-rule**. This rule denies all internet bound traffic from the *BackEnd* subnet.

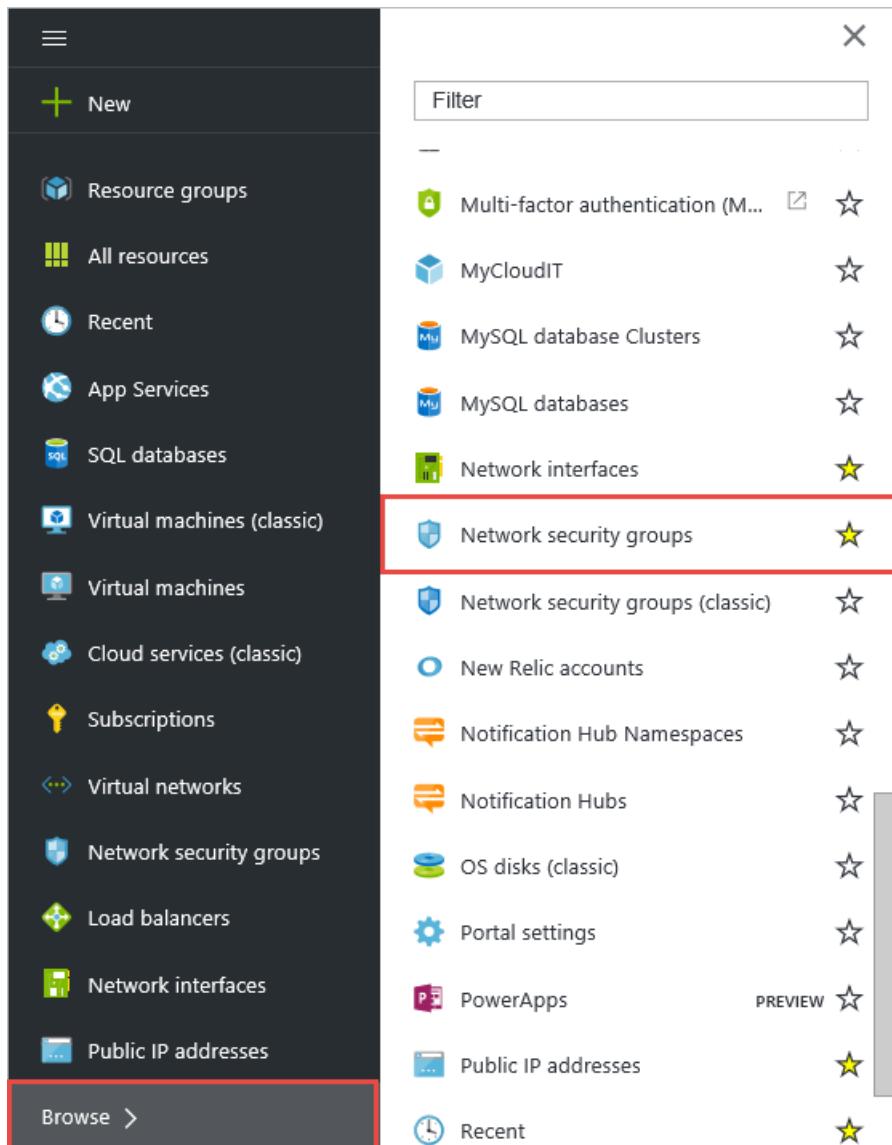
The combination of these rules create a DMZ-like scenario, where the back end subnet can only receive incoming traffic for SQL from the front end subnet, and has no access to the Internet, while the front end subnet can communicate with the Internet, and receive incoming HTTP requests only.

The sample PowerShell commands below expect a simple environment already created based on the scenario above. If you want to run the commands as they are displayed in this document, first build the test environment by deploying [this template](#), click **Deploy to Azure**, replace the default parameter values if necessary, and follow the instructions in the portal. The steps below use **RG-NSG** as the name of the resource group the template was deployed to.

Create the NSG-FrontEnd NSG

To create the **NSG-FrontEnd** NSG as shown in the scenario above, follow the steps below.

1. From a browser, navigate to <http://portal.azure.com> and, if necessary, sign in with your Azure account.
2. Click **Browse > > Network Security Groups**.



3. In the **Network security groups** blade, click **Add**.

The screenshot shows the 'Network security groups' blade in the Azure portal. At the top, there are three buttons: 'Add' (highlighted with a red box), 'Columns', and 'Refresh'. Below these are two filter options: 'Filter by name...' and 'NAME'. A message at the bottom states 'No network security groups to display'.

4. In the **Create network security group** blade, create an NSG named *NSG-FrontEnd* in the *RG-NSG* resource group, and then click **Create**.

The screenshot shows the 'Create network security group' blade. It includes fields for 'Name' (set to 'NSG-FrontEnd'), 'Subscription' (set to 'Microsoft Azure Internal Consumption (628)'), 'Resource Group' (set to 'RG-NSG'), 'Location' (set to 'West US'), and a checkbox for 'Pin to dashboard'. The 'Create' button is at the bottom.

Create rules in an existing NSG

To create rules in an existing NSG from the Azure portal, follow the steps below.

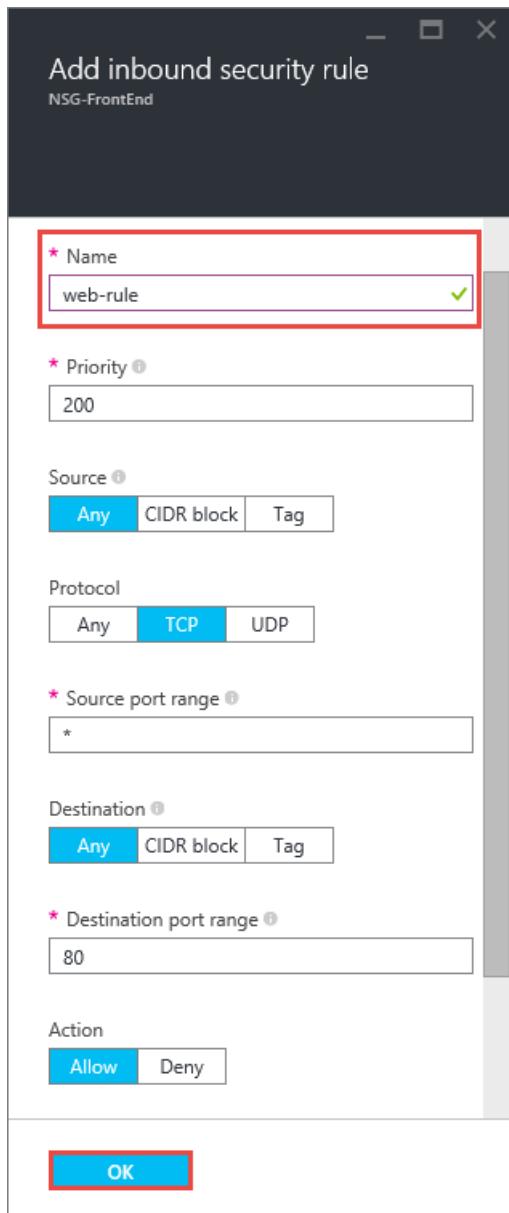
1. Click **Browse > > Network security groups**.
2. In the list of NSGs, click **NSG-FrontEnd > Inbound security rules**

The screenshot shows two windows side-by-side. On the left, the 'Network security groups' blade lists three entries: 'joamatestarm', 'NSG-BackEnd', and 'NSG-FrontEnd'. The 'NSG-FrontEnd' entry is highlighted with a red box. On the right, the 'NSG-FrontEnd' settings blade is open. In the 'Essentials' section, it shows the resource group 'TestRG', location 'West US', subscription 'Microsoft Azure Internal Consumption', and a single inbound rule. The 'Inbound security rules' link in the sidebar is also highlighted with a red box.

3. In the list of **Inbound security rules**, click **Add**.

The screenshot shows the 'NSG-FrontEnd' settings blade. The 'Inbound security rules' section is expanded, showing one existing rule named 'sql-rule' with priority 100, source '192.168.1.0/24', destination 'Any', and service 'TCP/1433'. A red box highlights the '+ Add' button under the list of rules.

4. In the **Add inbound security rule** blade, create a rule named *web-rule* with priority of *200* allowing access via *TCP* to port *80* to any VM from any source, and then click **OK**. Notice that most of these settings are default values already.



5. After a few seconds you will see the new rule in the NSG.

Inbound security rules					
PRIORITY	NAME	SOURCE	DESTINATION	SERVICE	ACTION
100	sql-rule	192.168.1.0/24	Any	TCP/1433	Allow
200	web-rule	Any	Any	TCP/80	Allow

6. Repeat steps to 6 to create an inbound rule named *rdp-rule* with a priority of *250* allowing access via *TCP* to port *3389* to any VM from any source.

Associate the NSG to the FrontEnd subnet

1. Click **Browse > > Resource groups > RG-NSG**.
2. In the **RG-NSG** blade, click ... > **TestVNet**.

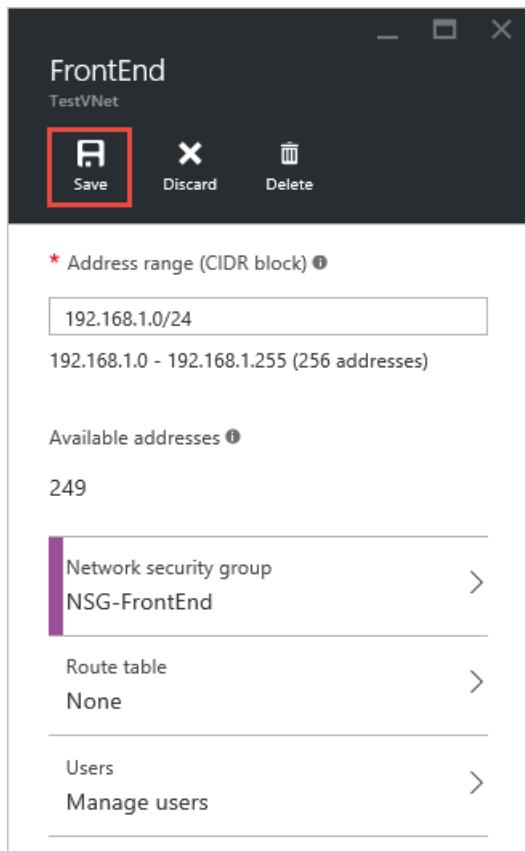
The screenshot shows the Azure portal interface. On the left, the 'RG-NSG' resource group blade displays basic information like Subscription name (Microsoft Azure Internal Consumption), Last deployment (2/15/2016 Succeeded), and a list of resources including sqlAvSet, webAvSet, SQL1, SQL2, Web1, Web2, TestNICSQL1, TestNICSQL2, and TestNICWeb1. On the right, the 'Resources' blade lists all resources under RG-NSG, with 'TestVNet' highlighted by a red box.

NAME	RESOURCE GROUP
sqlAvSet	RG-NSG
webAvSet	RG-NSG
SQL1	RG-NSG
SQL2	RG-NSG
Web1	RG-NSG
Web2	RG-NSG
TestNICSQL1	RG-NSG
TestNICSQL2	RG-NSG
TestNICWeb1	RG-NSG
TestNICWeb2	RG-NSG
NSG-FrontEnd	RG-NSG
TestPIPSQL1	RG-NSG
TestPIPSQL2	RG-NSG
TestPIPWeb1	RG-NSG
TestPIPWeb2	RG-NSG
TestVNet	RG-NSG

3. In the **Settings** blade, click **Subnets** > **FrontEnd** > **Network security group** > **NSG-FrontEnd**.

The screenshot shows three blades: 1) The 'Settings' blade for 'TestNet' with 'Subnets' selected. 2) The 'Subnets' blade for 'FrontEnd' where the 'FrontEnd' subnet is selected. 3) The 'Choose network security group' blade where 'NSG-FrontEnd' is selected for the network security group.

4. In the **FrontEnd** blade, click **Save**.



Create the NSG-BackEnd NSG

To create the **NSG-BackEnd** NSG and associate it to the **BackEnd** subnet, follow the steps below.

1. Repeat the steps in [Create the NSG-FrontEnd NSG](#) to create an NSG named *NSG-BackEnd*
2. Repeat the steps in [Create rules in an existing NSG](#) to create the **inbound** rules in the table below.

INBOUND RULE	OUTBOUND RULE
<p>* Name sql-rule ✓</p> <p>* Priority 100</p> <p>Source Any CIDR block Tag</p> <p>* Source IP address range 192.168.1.0/24 ✓</p> <p>Protocol Any TCP UDP</p> <p>* Source port range*</p> <p>Destination Any CIDR block Tag</p> <p>* Destination port range 1433 ✓</p> <p>Action Deny Allow</p>	<p>* Name web-rule ✓</p> <p>* Priority 100</p> <p>Destination Any CIDR block Tag</p> <p>Destination tag Internet</p> <p>* Destination port range*</p> <p>Source Any CIDR block Tag</p> <p>Protocol Any TCP UDP</p> <p>* Source port range*</p> <p>Action Deny Allow</p>

3. Repeat the steps in [Associate the NSG to the FrontEnd subnet](#) to associate the **NSG-Backend** NSG to the **BackEnd** subnet.

Next Steps

- Learn how to [manage existing NSGs](#)
- [Enable logging](#) for NSGs.

Creating an Internet-facing load balancer using the Azure portal

1/17/2017 • 5 min to read • [Edit on GitHub](#)

An Azure load balancer is a Layer-4 (TCP, UDP) load balancer. The load balancer provides high availability by distributing incoming traffic among healthy service instances in cloud services or virtual machines in a load balancer set. Azure Load Balancer can also present those services on multiple ports, multiple IP addresses, or both.

You can configure a load balancer to:

- Load balance incoming Internet traffic to virtual machines (VMs). We refer to a load balancer in this scenario as an [Internet-facing load balancer](#).
- Load balance traffic between VMs in a virtual network (VNet), between VMs in cloud services, or between on-premises computers and VMs in a cross-premises virtual network. We refer to a load balancer in this scenario as an [internal load balancer \(ILB\)](#).
- Forward external traffic to a specific VM instance.

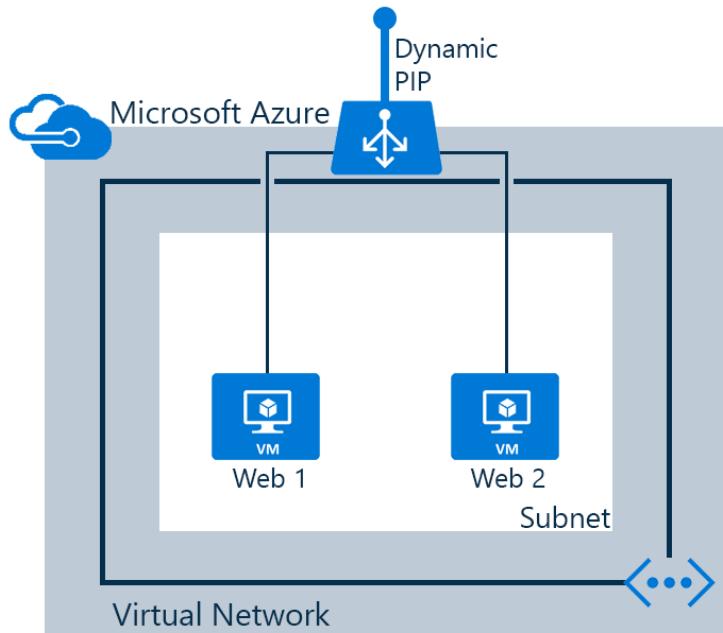
IMPORTANT

Before you work with Azure resources, it's important to understand that Azure currently has two deployment models: Azure Resource Manager and classic. Make sure you understand [deployment models and tools](#) before you work with any Azure resource. You can view the documentation for different tools by clicking the tabs at the top of this article.

This article covers the Resource Manager deployment model. You can also [Learn how to create an Internet-facing load balancer using classic deployment](#)

The following tasks will be done in this scenario:

- Create a load balancer that receives network traffic on port 80 and send load-balanced traffic to virtual machines "web1" and "web2"
- Create NAT rules for remote desktop access/SSH for virtual machines behind the load balancer
- Create health probes



This covers the sequence of individual tasks that have to be done to create a load balancer and explain in detail what is being done to accomplish the goal.

What is required to create an Internet-facing load balancer?

You need to create and configure the following objects to deploy a load balancer.

- Front-end IP configuration - contains public IP addresses for incoming network traffic.
- Back-end address pool - contains network interfaces (NICs) for the virtual machines to receive network traffic from the load balancer.
- Load balancing rules - contains rules mapping a public port on the load balancer to port in the back-end address pool.
- Inbound NAT rules - contains rules mapping a public port on the load balancer to a port for a specific virtual machine in the back-end address pool.
- Probes - contains health probes used to check availability of virtual machines instances in the back-end address pool.

You can get more information about load balancer components with Azure Resource Manager at [Azure Resource Manager support for Load Balancer](#).

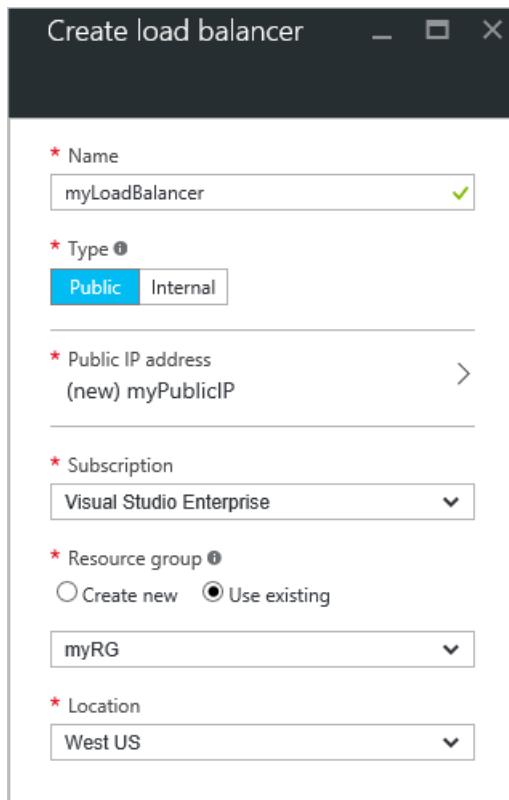
Set up a load balancer in Azure portal

IMPORTANT

This example assumes you have a virtual network called **myVNet**. Refer to [create virtual network](#) to do this. It also assumes there is a subnet within **myVNet** called **LB-Subnet-BE** and two VMs called **web1** and **web2** respectively within the same availability set called **myAvailSet** in **myVNet**. Refer to [this link](#) to create VMs.

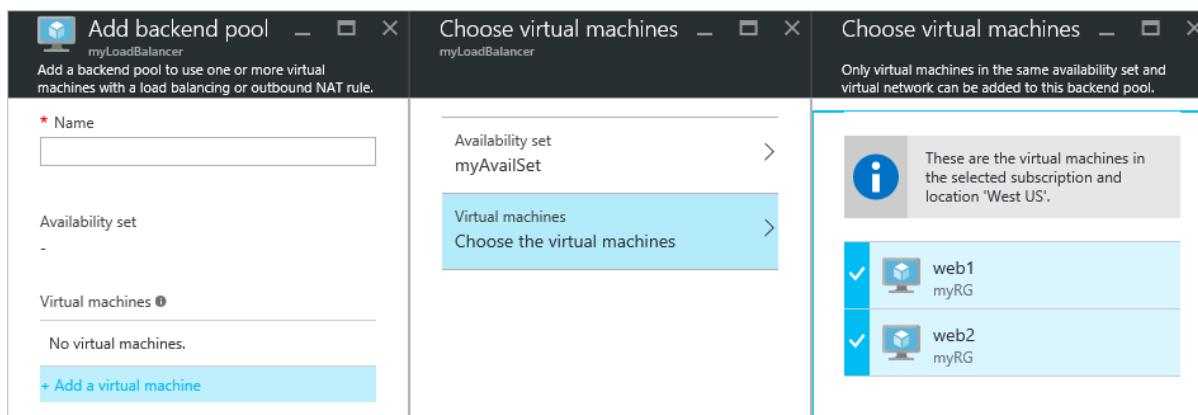
1. From a browser navigate to the Azure portal: <http://portal.azure.com> and login with your Azure account.
2. On the top left-hand side of the screen select **New > Networking > Load Balancer**.
3. In the **Create load balancer** blade, type a name for your load balancer. Here it is called **myLoadBalancer**.
4. Under **Type**, select **Public**.
5. Under **Public IP address**, create a new public IP called **myPublicIP**.
6. Under Resource Group, select **myRG**. Then select an appropriate **Location**, and then click **OK**. The load

balancer will then start to deploy and will take a few minutes to successfully complete deployment.



Create a back-end address pool

- Once your load balancer has successfully deployed, select it from within your resources. Under settings, select Backend Pools. Type a name for your backend pool. Then click on the **Add** button toward the top of the blade that shows up.
- Click on **Add a virtual machine** in the **Add backend pool** blade. Select **Choose an availability set** under **Availability set** and select **myAvailSet**. Next, select **Choose the virtual machines** under the Virtual Machines section in the blade and click on **web1** and **web2**, the two VMs created for load balancing. Ensure that both have blue check marks to the left as shown in the image below. Then, click **Select** in that blade followed by **OK** in the **Choose Virtual machines** blade and then **OK** in the **Add backend pool** blade.



- Check to make sure your notifications drop down list has an update regarding saving the load balancer backend pool in addition to updating the network interface for both the VMs **web1** and **web2**.

Create a probe, LB rule, and NAT rules

- Create a health probe.

Under Settings of your load balancer, select Probes. Then click **Add** located at the top of the blade.

There are two ways to configure a probe: HTTP or TCP. This example shows HTTP, but TCP can be configured in a similar manner. Update the necessary information. As mentioned, **myLoadBalancer** will load balance traffic on Port 80. The path selected is HealthProbe.aspx, Interval is 15 seconds, and Unhealthy threshold is 2. Once finished, click **OK** to create the probe.

Hover your pointer over the 'i' icon to learn more about these individual configurations and how they can be changed to cater to your requirements.

The screenshot shows a 'Add probe' dialog box for a load balancer named 'myLoadBalancer'. The dialog has the following fields:

- Name:** HealthProbe (marked with a red asterisk)
- Protocol:** HTTP (selected, highlighted in blue)
- Port:** 80
- Path:** HealthProbe.aspx
- Interval:** 15 seconds
- Unhealthy threshold:** 2 consecutive failures

2. Create a load balancer rule.

Click on Load balancing rules in the Settings section of your load balancer. In the new blade, click on **Add**. Name your rule. Here, it is HTTP. Choose the frontend port and Backend port. Here, 80 is chosen for both. Choose **LB-backend** as your Backend pool and the previously created **HealthProbe** as the Probe. Other configurations can be set according to your requirements. Then click OK to save the load balancing rule.

Add load balancing rule — □ X

myLoadBalancer

* Name
HTTP ✓

Protocol
TCP UDP

* Port
80

* Backend port ⓘ
80

Backend pool ⓘ
LB-backend (2 virtual machines) ▾

Probe ⓘ
HealthProbe (HTTP:80/HealthProbe.aspx) ▾

Session persistence ⓘ
None ▾

Idle timeout (minutes) ⓘ
 4

Floating IP (direct server return) ⓘ
Disabled Enabled

3. Create inbound NAT rules

Click on Inbound NAT rules under the settings section of your load balancer. In the new blade that, click **Add**. Then name your inbound NAT rule. Here it is called **inboundNATRule1**. The destination should be the Public IP previously created. Select Custom under Service and select the protocol you would like to use. Here TCP is selected. Enter the port, 3441, and the Target port, in this case, 3389. then click OK to save this rule.

Once the first rule is created, repeat this step for the second inbound NAT rule called inboundNATrule2 from port 3442 to Target port 3389.

Add inbound NAT rule – X

myLoadBalancer

Create an inbound NAT rule to route incoming traffic to a virtual machine in your virtual network.

* Name ✓

Destination i
52.160.109.200 (myPublicIP)

Service

Protocol

* Port ✓

Target i >
Choose a virtual machine

Port mapping i

Floating IP (direct server return) i

* Target port ✓

Remove a Load Balancer

To delete a load balancer, select the load balancer you want to remove. In the *Load Balancer* blade, click on **Delete** located at the top of the blade. Then select **Yes** when prompted.

Next steps

[Get started configuring an internal load balancer](#)

[Configure a load balancer distribution mode](#)

[Configure idle TCP timeout settings for your load balancer](#)

Create a VM with a static public IP using the Azure portal

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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. 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.

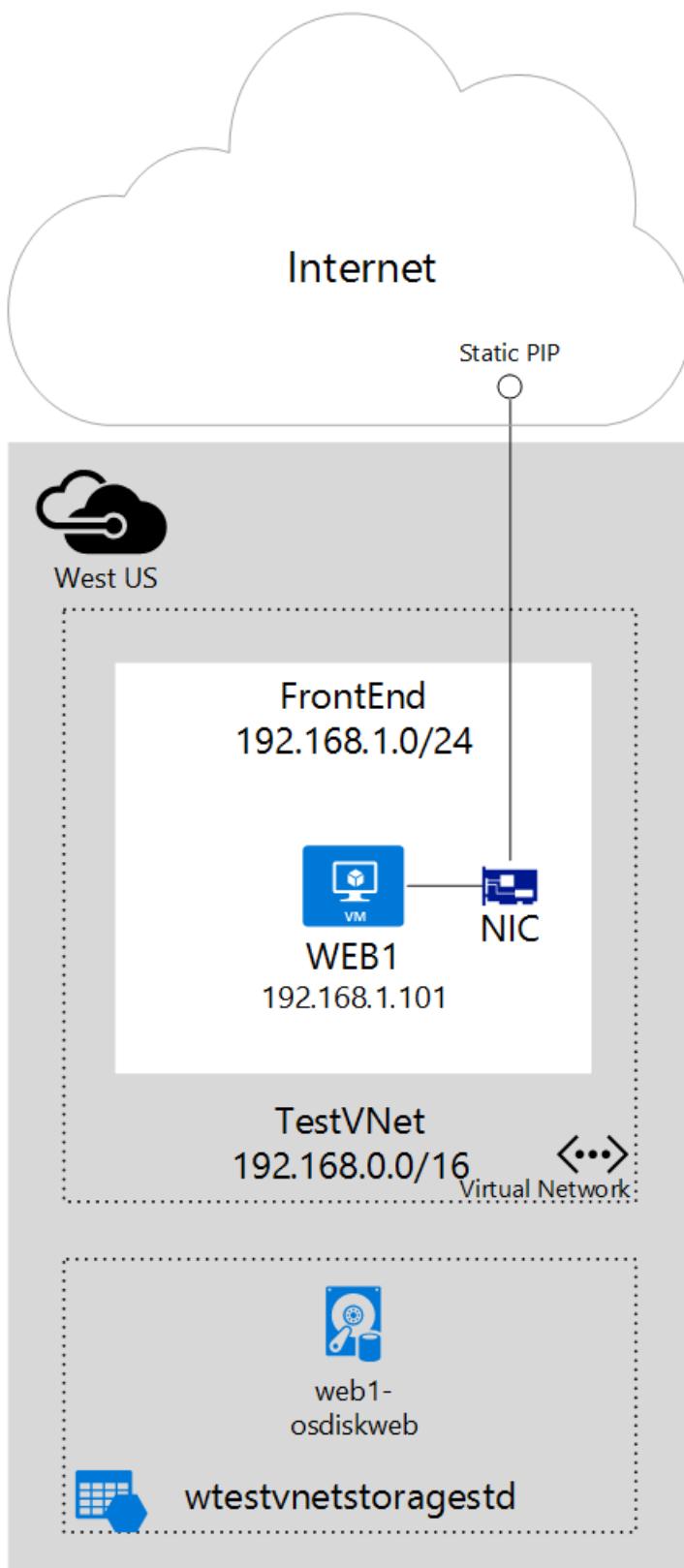
NOTE

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
***** ✓

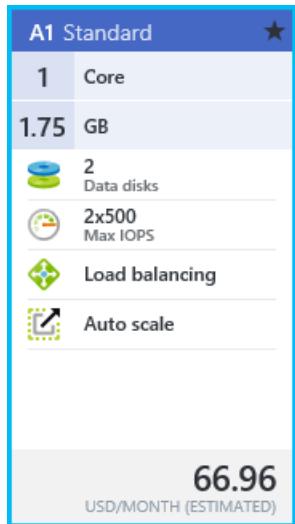
* Subscription
Microsoft Azure Internal Consumption

* Resource group
PublicIPTest ✕ ✓

Select existing

* Location
East US 2

- In the **Choose a size** blade, click **A1 Standard** as shown below, and then click **Select**.



- 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

- In the **Settings** blade, click **OK**.
- 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

- Notice the new tile in your dashboard.



- 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**

Status: **Running**

Location: **East US 2**

Subscription name: **Microsoft Azure Internal Consumption**

Subscription ID: **628dad04-b5d1-4f10-b3a4-dc61d88cf97c**

Computer name: **WEB1**

Size: **Standard A1 (1 core, 1.75 GB memory)**

Operating system: **Windows**

Public IP address/DNS name label: **104.208.232.131/<none>**

Virtual network/subnet: **PublicIPTest/default**

[All settings →](#)

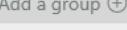
Monitoring

CPU percentage

100%
80%
60%
40%
20%
0%

11 AM 11:15 AM 11:30 AM 11:45 AM

CPU UTILIZATION
0.08 %

Add a group 

Add tiles 



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 >

Connect virtual networks from different deployment models in the portal

1/17/2017 • 15 min to read • [Edit on GitHub](#)

Azure currently has two management models: classic and Resource Manager (RM). If you have been using Azure for some time, you probably have Azure VMs and instance roles running in a classic VNet. Your newer VMs and role instances may be running in a VNet created in Resource Manager. This article walks you through connecting classic VNets to Resource Manager VNets to allow the resources located in the separate deployment models to communicate with each other over a gateway connection.

You can create a connection between VNets that are in different subscriptions and in different regions. You can also connect VNets that already have connections to on-premises networks, as long as the gateway that they have been configured with is dynamic or route-based. For more information about VNet-to-VNet connections, see the [VNet-to-VNet FAQ](#) at the end of this article.

Deployment models and methods for VNet-to-VNet connections

It's important to know that Azure currently works with two deployment models: Resource Manager and classic. Before you begin your configuration, make sure that you understand the deployment models and tools. You'll need to know which model that you want to work in. Not all networking features are supported yet for both models. For information about the deployment models, see [Understanding Resource Manager deployment and classic deployment](#).

We update the following table as new articles and additional tools become available for this configuration. When an article is available, we link directly to it from the table.

VNet-to-VNet

DEPLOYMENT MODEL/METHOD	AZURE PORTAL	CLASSIC PORTAL	POWERSHELL
Classic	Not Supported	Article*	Supported
Resource Manager	Article+	Not Supported	Article
Connections between different deployment models	Article*	Article*	Article

(+) denotes this deployment method is available only for VNets in the same subscription.

(*) denotes that this deployment method also requires PowerShell.

VNet peering

It's also possible to connect VNets without using a VPN gateway. If your VNets are in the same region, you may want to consider connecting them by using VNet peering. For more information, see the [VNet peering](#) article.

Before beginning

The following steps walk you through the settings necessary to configure a dynamic or route-based gateway for each VNet and create a VPN connection between the gateways. This configuration does not support static or

policy-based gateways.

In this article, we use the classic portal, the Azure portal, and PowerShell. Currently, it's not possible to create this configuration using only the Azure portal.

Prerequisites

- Both VNets have already been created.
- The address ranges for the VNets do not overlap with each other, or overlap with any of the ranges for other connections that the gateways may be connected to.
- You have installed the latest PowerShell cmdlets (1.0.2 or later). See [How to install and configure Azure PowerShell](#) for more information. Make sure you install both the Service Management (SM) and the Resource Manager (RM) cmdlets.

Example settings

You can use the example settings as reference.

Classic VNet settings

VNet Name = ClassicVNet

Location = West US

Virtual Network Address Spaces = 10.0.0.0/24

Subnet-1 = 10.0.0.0/27

GatewaySubnet = 10.0.0.32/29

Local Network Name = RMVNetLocal

Resource Manager VNet settings

VNet Name = RMVNet

Resource Group = RG1

Virtual Network IP Address Spaces = 192.168.0.0/16

Subnet-1 = 192.168.1.0/24

GatewaySubnet = 192.168.0.0/26

Location = East US

Virtual network gateway name = RMGateway

Gateway public IP name = gwpip

Gateway type = VPN

VPN type = Route-based

Local network gateway = ClassicVNetLocal

Section 1: Configure classic VNet settings

In this section, we create the local network and the gateway for your classic VNet. The instructions in this section use the classic portal. Currently, the Azure portal does not offer all the settings that pertain to a classic VNet.

Part 1 - Create a new local network

Open the [classic portal](#) and sign in with your Azure account.

1. On the bottom left corner of the screen, click **NEW > Network Services > Virtual Network > Add local network**.
2. In the **Specify your local network details** window, type a name for the RM VNet you want to connect to. In the **VPN device IP address (optional)** box, type any valid public IP address. This is just a temporary placeholder. You change this IP address later. On the bottom right corner of the window, click the arrow button.
3. On the **Specify the address space** page, in the **Starting IP** text box, type the network prefix and CIDR block for the Resource Manager VNet you want to connect to. This setting is used to specify the address space to route to the RM VNet.

Part 2 - Associate the local network to your VNet

1. Click **Virtual Networks** at the top of the page to switch to the Virtual Networks screen, then click to select your classic VNet. On the page for your VNet, click **Configure** to navigate to the configuration page.
2. Under the **site-to-site connectivity** connection section, select the **Connect to the local network** checkbox. Then select the local network that you created. If you have multiple local networks that you created, be sure to select the one that you created to represent your Resource Manager VNet from the dropdown.
3. Click **Save** at the bottom of the page.

Part 3 - Create the gateway

1. After saving the settings, click **Dashboard** at the top of the page to change to the Dashboard page. On the bottom of the Dashboard page, click **Create Gateway**, then click **Dynamic Routing**. Click **Yes** to begin creating your gateway. A Dynamic Routing gateway is required for this configuration.
2. Wait for the gateway to be created. This can sometimes take 45 minutes or more to complete.

Part 4 - View the gateway public IP address

After the gateway has been created, you can view the gateway IP address on the **Dashboard** page. This is the public IP address of your gateway. Write down or copy the public IP address. You use it in later steps when you create the local network for your Resource Manager VNet configuration.

Section 2: Configure Resource Manager VNet settings

In this section, we create the virtual network gateway and the local network for your Resource Manager VNet. Don't start the following steps until after you have retrieved the public IP address for the classic VNet's gateway.

The screenshots are provided as examples. Be sure to replace the values with your own. If you are creating this configuration as an exercise, refer to these [values](#).

Part 1 - Create a gateway subnet

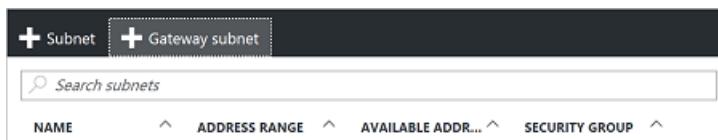
Before connecting your virtual network to a gateway, you first need to create the gateway subnet for the virtual network to which you want to connect. Create a gateway subnet with CIDR count of /28 or larger (/27, /26, etc.)

IMPORTANT

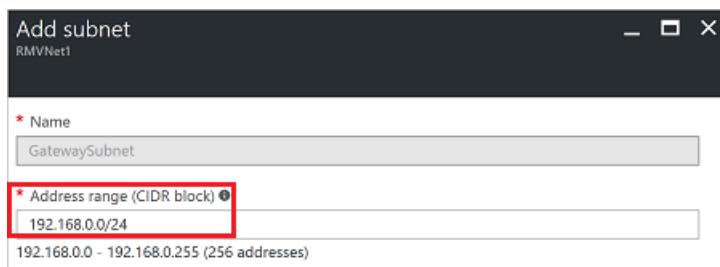
When working with gateway subnets, avoid associating a network security group (NSG) to the gateway subnet. Associating a network security group to this subnet may cause your VPN gateway to stop functioning as expected. For more information about network security groups, see [What is a network security group?](#)

From a browser, navigate to the [Azure portal](#) and sign in with your Azure account.

1. In the portal, navigate to the Resource Manager virtual network for which you want to create a virtual network gateway.
2. In the **Settings** section of your VNet blade, click **Subnets** to expand the Subnets blade.
3. On the **Subnets** blade, click **+Gateway subnet** at the top. This will open the **Add subnet** blade.



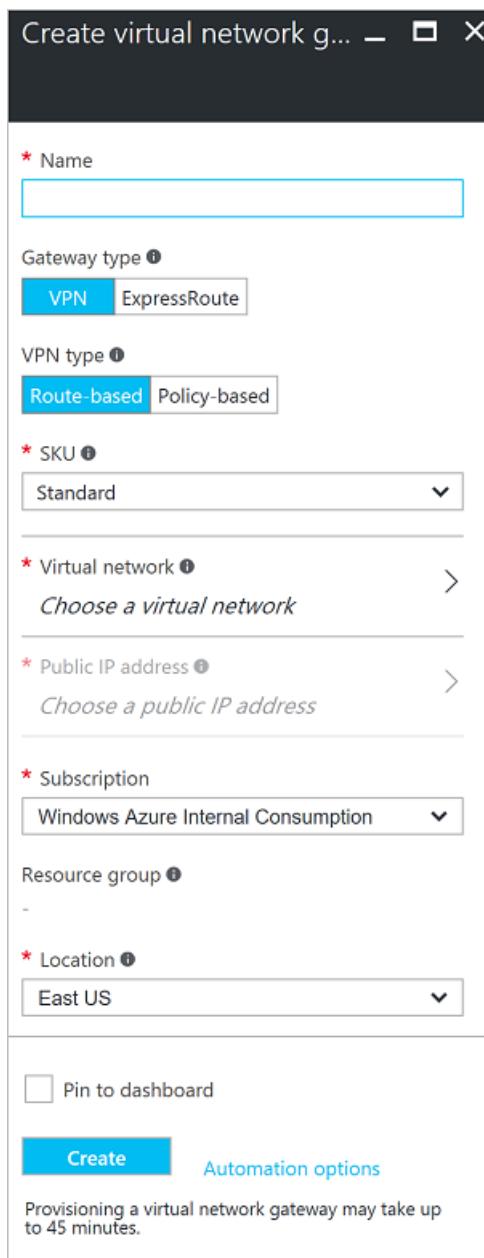
4. The **Name** for your subnet will automatically be filled in with the value 'GatewaySubnet'. This value is required in order for Azure to recognize the subnet as the gateway subnet. Adjust the auto-filled **Address range** values to match your configuration requirements.



5. Click **OK** at the bottom of the blade to create the subnet.

Part 2 - Create a virtual network gateway

1. In the portal, on the left side, click **+** and type "Virtual Network Gateway" in search. Locate **Virtual network gateway** in the search return and click the entry. On the **Virtual network gateway** blade, click **Create** at the bottom of the blade. This opens the **Create virtual network gateway** blade.
2. On the **Create virtual network gateway** blade, fill in the values for your virtual network gateway.



3. **Name:** Name your gateway. This is not the same as naming a gateway subnet. It's the name of the gateway object you are creating.
4. **Gateway type:** Select **VPN**. VPN gateways use the virtual network gateway type **VPN**.
5. **VPN type:** Select the VPN type that is specified for your configuration. Most configurations require a Route-based VPN type.

6. **SKU:** Select the gateway SKU from the dropdown. The SKUs listed in the dropdown depend on the VPN type you select.
7. **Location:** Adjust the **Location** field to point to the location where your virtual network is located. If the location is not pointing to the region where your virtual network resides, the virtual network will not appear in the 'Choose a virtual network' dropdown.
8. Choose the virtual network to which you want to add this gateway. Click **Virtual network** to open the **Choose a virtual network** blade. Select the VNet. If you don't see your VNet, make sure the **Location** field is pointing to the region in which your virtual network is located.
9. Choose a public IP address. Click **Public IP address** to open the **Choose public IP address** blade. Click **+Create New** to open the **Create public IP address blade**. Input a name for your public IP address. This blade creates a public IP address object to which a public IP address will be dynamically assigned. Click **OK** to save your changes to this blade.
10. **Subscription:** Verify that the correct subscription is selected.
11. **Resource group:** This setting is determined by the Virtual Network that you select.
12. Don't adjust the **Location** after you've specified the previous settings.
13. Verify the settings. You can select **Pin to dashboard** at the bottom of the blade if you want your gateway to appear on the dashboard.
14. Click **Create** to begin creating the gateway. The settings will be validated and you'll see the "Deploying Virtual network gateway" tile on the dashboard. Creating a gateway can take up to 45 minutes. You may need to refresh your portal page to see the completed status.



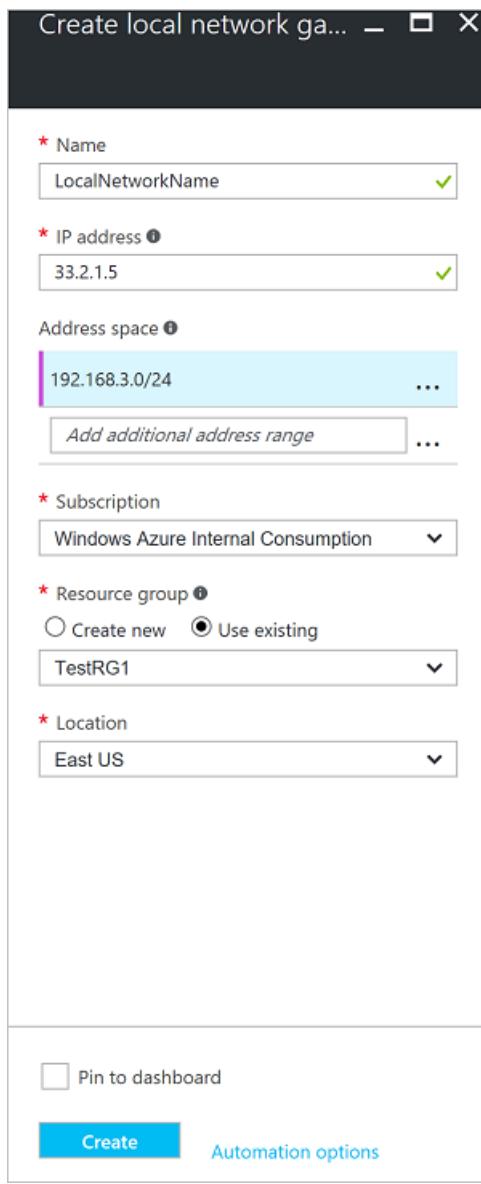
15. After the gateway is created, you can view the IP address that has been assigned to it by looking at the virtual network in the portal. The gateway will appear as a connected device. You can click the connected device (your virtual network gateway) to view more information.

Part 3 - Create a local network gateway

The 'local network gateway' typically refers to your on-premises location. It tells Azure which IP address ranges to route to the location and the public IP address of the device for that location. However, in this case, it refers to the address range and public IP address associated with your classic VNet and virtual network gateway.

Give the local network gateway a name by which Azure can refer to it. You can create your local network gateway while your virtual network gateway is being created. For this configuration, you use the public IP address that was assigned to your classic VNet gateway in the [previous section](#).

1. In the portal, from **All resources**, click **+Add**. In the **Everything** blade search box, type **Local network gateway**, then click to search. This will return a list. Click **Local network gateway** to open the blade, then click **Create** to open the **Create local network gateway** blade.



2. On the **Create local network gateway blade**, specify a **Name** for your local network gateway object.
3. Specify a valid public **IP address** for the VPN device or virtual network gateway to which you want to connect.
If this local network represents an on-premises location, this is the public IP address of the VPN device that you want to connect to. It cannot be behind NAT and has to be reachable by Azure.
If this local network represents another VNet, you will specify the public IP address that was assigned to the virtual network gateway for that VNet.
4. **Address Space** refers to the address ranges for the network that this local network represents. You can add multiple address space ranges. Make sure that the ranges you specify here do not overlap with ranges of other networks that you want to connect to.
5. For **Subscription**, verify that the correct subscription is showing.
6. For **Resource Group**, select the resource group that you want to use. You can either create a new resource group, or select one that you have already created.
7. For **Location**, select the location that this object will be created in. You may want to select the same location that your VNet resides in, but you are not required to do so.
8. Click **Create** to create the local network gateway.

Part 4 - Copy the public IP address

Once the virtual network gateway has finished creating, copy the public IP address that is associated with the gateway. You use it when you configure the local network settings for your classic VNet.

Section 3: Modify the local network for the classic VNet

Open the [classic portal](#).

1. In the classic portal, scroll down on the left side and click **Networks**. On the **networks** page, click **Local Networks** at the top of the page.
2. Click to select the local network that you configured in Part 1. At the bottom of the page, click **Edit**.
3. On the **Specify your local network details** page, replace the placeholder IP address with the public IP address for the Resource Manager gateway that you created in the previous section. Click the arrow to move to the next section. Verify that the **Address Space** is correct, and then click the checkmark to accept the changes.

Section 4: Create the connection

In this section, we create the connection between the VNets. The steps for this require PowerShell. You cannot create this connection in either of the portals. Make sure you have downloaded and installed both the classic (SM) and Resource Manager (RM) PowerShell cmdlets.

1. Log in to your Azure account in the PowerShell console. The following cmdlet prompts you for the login credentials for your Azure Account. After logging in, your account settings are downloaded so that they are available to Azure PowerShell.

```
Login-AzureRmAccount
```

Get a list of your Azure subscriptions if you have more than one subscription.

```
Get-AzureRmSubscription
```

Specify the subscription that you want to use.

```
Select-AzureRmSubscription -SubscriptionName "Name of subscription"
```

2. Add your Azure Account to use the classic PowerShell cmdlets. To do so, you can use the following command:

```
Add-AzureAccount
```

3. Set your shared key by running the following sample. In this sample, `-VNetName` is the name of the classic VNet and `-LocalNetworkSiteName` is the name you specified for the local network when you configured it in the classic portal. The `-SharedKey` is a value that you can generate and specify. The value you specify here must be the same value that you specify in the next step when you create your connection.

```
Set-AzureVNetGatewayKey -VNetName ClassicVNet `  
-LocalNetworkSiteName RMVNetLocal -SharedKey abc123
```

4. Create the VPN connection by running the following commands:

Set the variables

```
$vnet01gateway = Get-AzureRMLocalNetworkGateway -Name ClassicVNetLocal -ResourceGroupName RG1  
$vnet02gateway = Get-AzureRmVirtualNetworkGateway -Name RMGateway -ResourceGroupName RG1
```

Create the connection

Note that the `-ConnectionType` is 'IPsec', not 'Vnet2Vnet'. In this sample, `-Name` is the name that you want to call your connection. The following sample creates a connection named '*rm-to-classic-connection*'.

```
New-AzureRmVirtualNetworkGatewayConnection -Name rm-to-classic-connection -ResourceGroupName RG1 `  
-Location "East US" -VirtualNetworkGateway1 `  
$vnet02gateway -LocalNetworkGateway2 `  
$vnet01gateway -ConnectionType IPsec -RoutingWeight 10 -SharedKey 'abc123'
```

Verify your connection

You can verify your connection by using the classic portal, the Azure portal, or PowerShell. You can use the following steps to verify your connection. Replace the values with your own.

To verify your connection by using PowerShell

You can verify that your connection succeeded by using the `Get-AzureRmVirtualNetworkGatewayConnection` cmdlet, with or without `-Debug`.

1. Use the following cmdlet example, configuring the values to match your own. If prompted, select 'A' in order to run 'All'. In the example, `-Name` refers to the name of the connection that you created and want to test.

```
Get-AzureRmVirtualNetworkGatewayConnection -Name MyGWConnection -ResourceGroupName MyRG
```

2. After the cmdlet has finished, view the values. In the example below, the connection status shows as 'Connected' and you can see ingress and egress bytes.

```
Body:  
{  
    "name": "MyGWConnection",  
    "id":  
        "/subscriptions/086cfcaa0-0d1d-4b1c-94544-  
f8e3da2a0c7789/resourceGroups/MyRG/providers/Microsoft.Network/connections/MyGWConnection",  
    "properties": {  
        "provisioningState": "Succeeded",  
        "resourceGuid": "1c484f82-23ec-47e2-8cd8-231107450446b",  
        "virtualNetworkGateway1": {  
            "id":  
                "/subscriptions/086cfcaa0-0d1d-4b1c-94544-  
f8e3da2a0c7789/resourceGroups/MyRG/providers/Microsoft.Network/virtualNetworkGa  
teways/vnetgw1"  
            },  
            "localNetworkGateway2": {  
                "id":  
                    "/subscriptions/086cfcaa0-0d1d-4b1c-94544-  
f8e3da2a0c7789/resourceGroups/MyRG/providers/Microsoft.Network/localNetworkGate  
ways/LocalSite"  
                },  
                "connectionType": "IPsec",  
                "routingWeight": 10,  
                "sharedKey": "abc123",  
                "connectionStatus": "Connected",  
                "ingressBytesTransferred": 33509044,  
                "egressBytesTransferred": 4142431  
    }  
}
```

To verify your connection by using the Azure portal

In the Azure portal, you can view the connection status by navigating to the connection. There are multiple ways to do this. The following steps show one way to navigate to your connection and verify.

1. In the [Azure portal](#), click **All resources** and navigate to your virtual network gateway.
2. On the blade for your virtual network gateway, click **Connections**. You can see the status of each connection.
3. Click the name of the connection that you want to verify to open **Essentials**. In Essentials, you can view more information about your connection. The **Status** is 'Succeeded' and 'Connected' when you have made a

successful connection.

Essentials ^	
Resource group	Data in 2.35 KB
RG1	 Data out 3.14 KB
Status	Virtual network
Connected	RMVNet
Location	Virtual network gateway
East US	 RMGateway (40.114.5.29)
Subscription name	Local network gateway
Windows Azure Internal Consumption	 Site2 (40.76.7.127)
Subscription ID	

VNet-to-VNet FAQ

View the FAQ details for additional information about VNet-to-VNet connections.

- The virtual networks can be in the same or different Azure regions (locations).
- A cloud service or a load balancing endpoint CANNOT span across virtual networks, even if they are connected together.
- Connecting multiple Azure virtual networks together doesn't require any on-premises VPN gateways unless cross-premises connectivity is required.
- VNet-to-VNet supports connecting virtual networks. It does not support connecting virtual machines or cloud services NOT in a virtual network.
- VNet-to-VNet requires Azure VPN gateways with RouteBased (previously called Dynamic Routing) VPN types.
- Virtual network connectivity can be used simultaneously with multi-site VPNs. There is a maximum of 10 (Default/Standard Gateways) or 30 (HighPerformance Gateways) VPN tunnels for a virtual network VPN gateway connecting to either other virtual networks, or on-premises sites.
- The address spaces of the virtual networks and on-premises local network sites must not overlap. Overlapping address spaces will cause the creation of VNet-to-VNet connections to fail.
- Redundant tunnels between a pair of virtual networks are supported when one virtual network gateway is configured as active-active.
- All VPN tunnels of the virtual network share the available bandwidth on the Azure VPN gateway and the same VPN gateway uptime SLA in Azure.
- VNet-to-VNet traffic travels across the Microsoft Network, not the Internet.
- VNet-to-VNet traffic within the same region is free for both directions. Cross region VNet-to-VNet egress traffic is charged with the outbound inter-VNet data transfer rates based on the source regions. Please refer to the [pricing page](#) for details.

Using Azure DNS with other Azure services

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 <i>cloudapp.net</i> name. This rule applies to VMs created in the classic portal because they are deployed as a cloud service. For more information, see Configure a custom domain name in Cloud Services .
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 <i>azurewebsites.net</i> name. For more information, see Map a custom domain name to an Azure app

AZURE SERVICE	NETWORK INTERFACE	DESCRIPTION
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.
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.

Traffic Manager endpoints

1/17/2017 • 8 min to read • [Edit on GitHub](#)

Microsoft Azure Traffic Manager allows you to control how network traffic is distributed to application deployments running in different datacenters. You configure each application deployment as an 'endpoint' in Traffic Manager. When Traffic Manager receives a DNS request, it chooses an available endpoint to return in the DNS response. Traffic manager bases the choice on the current endpoint status and the traffic-routing method. For more information, see [How Traffic Manager Works](#).

There are three types of endpoint supported by Traffic Manager:

- **Azure endpoints** are used for services hosted in Azure.
- **External endpoints** are used for services hosted outside Azure, either on-premises or with a different hosting provider.
- **Nested endpoints** are used to combine Traffic Manager profiles to create more flexible traffic-routing schemes to support the needs of larger, more complex deployments.

There is no restriction on how endpoints of different types are combined in a single Traffic Manager profile. Each profile can contain any mix of endpoint types.

The following sections describe each endpoint type in greater depth.

Azure endpoints

Azure endpoints are used for Azure-based services in Traffic Manager. The following Azure resource types are supported:

- 'Classic' IaaS VMs and PaaS cloud services.
- Web Apps
- PublicIPAddress resources (which can be connected to VMs either directly or via an Azure Load Balancer). The publicIpAddress must have a DNS name assigned to be used in a Traffic Manager profile.

PublicIPAddress resources are Azure Resource Manager resources. They do not exist in the classic deployment model. Thus they are only supported in Traffic Manager's Azure Resource Manager experiences. The other endpoint types are supported via both Resource Manager and the classic deployment model.

When using Azure endpoints, Traffic Manager detects when a 'Classic' IaaS VM, cloud service, or a Web App is stopped and started. This status is reflected in the endpoint status. See [Traffic Manager endpoint monitoring](#) for details. When the underlying service is stopped, Traffic Manager does not perform endpoint health checks or direct traffic to the endpoint. No Traffic Manager billing events occur for the stopped instance. When the service is restarted, billing resumes and the endpoint is eligible to receive traffic. This detection does not apply to PublicIpEndpoint endpoints.

External endpoints

External endpoints are used for services outside of Azure. For example, a service hosted on-premises or with a different provider. External endpoints can be used individually or combined with Azure Endpoints in the same Traffic Manager profile. Combining Azure endpoints with External endpoints enables various scenarios:

- In either an active-active or active-passive failover model, use Azure to provide increased redundancy for an existing on-premises application.
- To reduce application latency for users around the world, extend an existing on-premises application to

additional geographic locations in Azure. For more information, see [Traffic Manager 'Performance' traffic routing](#).

- Use Azure to provide additional capacity for an existing on-premises application, either continuously or as a 'burst-to-cloud' solution to meet a spike in demand.

In certain cases, it is useful to use External endpoints to reference Azure services (for examples, see the [FAQ](#)). In this case, health checks are billed at the Azure endpoints rate, not the External endpoints rate. However, unlike Azure endpoints, if you stop or delete the underlying service, health check billing continues until you disable or delete the endpoint in Traffic Manager.

Nested endpoints

Nested endpoints combine multiple Traffic Manager profiles to create flexible traffic-routing schemes and support the needs of larger, complex deployments. With Nested endpoints, a 'child' profile is added as an endpoint to a 'parent' profile. Both the child and parent profiles can contain other endpoints of any type, including other nested profiles. For more information, see [nested Traffic Manager profiles](#).

Web Apps as endpoints

Some additional considerations apply when configuring Web Apps as endpoints in Traffic Manager:

1. Only Web Apps at the 'Standard' SKU or above are eligible for use with Traffic Manager. Attempts to add a Web App of a lower SKU fail. Downgrading the SKU of an existing Web App results in Traffic Manager no longer sending traffic to that Web App.
2. When an endpoint receives an HTTP request, it uses the 'host' header in the request to determine which Web App should service the request. The host header contains the DNS name used to initiate the request, for example 'contosoapp.azurewebsites.net'. To use a different DNS name with your Web App, the DNS name must be registered as a custom domain name for the App. When adding a Web App endpoint as an Azure endpoint, the Traffic Manager profile DNS name is automatically registered for the App. This registration is automatically removed when the endpoint is deleted.
3. Each Traffic Manager profile can have at most one Web App endpoint from each Azure region. To work around for this constraint, you can configure a Web App as an External endpoint. For more information, see the [FAQ](#).

Enabling and disabling endpoints

Disabling an endpoint in Traffic Manager can be useful to temporarily remove traffic from an endpoint that is in maintenance mode or being redeployed. Once the endpoint is running again, it can be re-enabled.

Endpoints can be enabled and disabled via the Traffic Manager portal, PowerShell, CLI or REST API, all of which are supported in both Resource Manager and the classic deployment model.

NOTE

Disabling an Azure endpoint has nothing to do with its deployment state in Azure. An Azure service (such as a VM or Web App) remains running and able to receive traffic even when disabled in Traffic Manager. Traffic can be addressed directly to the service instance rather than via the Traffic Manager profile DNS name. For more information, see [how Traffic Manager works](#).

The current eligibility of each endpoint to receive traffic depends on the following factors:

- The profile status (enabled/disabled)
- The endpoint status (enabled/disabled)
- The results of the health checks for that endpoint

For details, see [Traffic Manager endpoint monitoring](#).

NOTE

Since Traffic Manager works at the DNS level, it is unable to influence existing connections to any endpoint. When an endpoint is unavailable, Traffic Manager directs new connections to another available endpoint. However, the host behind the disabled or unhealthy endpoint may continue to receive traffic via existing connections until those sessions are terminated. Applications should limit the session duration to allow traffic to drain from existing connections.

If all endpoints in a profile are disabled, or if the profile itself is disabled, then Traffic Manager sends an 'NXDOMAIN' response to a new DNS query.

FAQ

Can I use Traffic Manager with endpoints from multiple subscriptions?

Using endpoints from multiple subscriptions is not possible with Azure Web Apps. Azure Web Apps requires that any custom domain name used with Web Apps is only used within a single subscription. It is not possible to use Web Apps from multiple subscriptions with the same domain name.

For other endpoint types, it is possible to use Traffic Manager with endpoints from more than one subscription. How you configure Traffic Manager depends on whether you are using the classic deployment model or the Resource Manager experience.

- In Resource Manager, endpoints from any subscription can be added to Traffic Manager, so long as the person configuring the Traffic Manager profile has read access to the endpoint. These permissions can be granted using [Azure Resource Manager role-based access control \(RBAC\)](#).
- In the classic deployment model interface, Traffic Manager requires that Cloud Services or Web Apps configured as Azure endpoints reside in the same subscription as the Traffic Manager profile. Cloud Service endpoints in other subscriptions can be added to Traffic Manager as 'external' endpoints. These external endpoints are billed as Azure endpoints, rather than the external rate.

Can I use Traffic Manager with Cloud Service 'Staging' slots?

Yes. Cloud Service 'staging' slots can be configured in Traffic Manager as External endpoints. Health checks are still be charged at the Azure Endpoints rate. Because the External endpoint type is in use, changes to the underlying service are not picked up automatically. With external endpoints, Traffic Manager cannot detect when the Cloud Service is stopped or deleted. Therefore, the Traffic Manager continues billing for health checks until the endpoint is disabled or deleted.

Does Traffic Manager support IPv6 endpoints?

Traffic Manager does not currently provide IPv6-addressable name servers. However, Traffic Manager can still be used by IPv6 clients connecting to IPv6 endpoints. A client does not make DNS requests directly to Traffic Manager. Instead, the client uses a recursive DNS service. An IPv6-only client sends requests to the recursive DNS service via IPv6. Then the recursive service should be able to contact the Traffic Manager name servers using IPv4.

Traffic Manager responds with the DNS name of the endpoint. To support an IPv6 endpoint, a DNS AAAA record pointing the endpoint DNS name to the IPv6 address must exist. Traffic Manager health checks only support IPv4 addresses. The service needs to expose an IPv4 endpoint on the same DNS name.

Can I use Traffic Manager with more than one Web App in the same region?

Typically, Traffic Manager is used to direct traffic to applications deployed in different regions. However, it can also be used where an application has more than one deployment in the same region. The Traffic Manager Azure endpoints do not permit more than one Web App endpoint from the same Azure region to be added to the same Traffic Manager profile.

The following steps provide a workaround to this constraint:

1. Check that your endpoints are in different web app 'scale units'. A domain name must map to a single site in a given scale unit. Therefore, two Web Apps in the same scale unit cannot share a Traffic Manager profile.
2. Add your vanity domain name as a custom hostname to each Web App. Each Web App must be in a different scale unit. All Web Apps must belong to the same subscription.
3. Add one (and only one) Web App endpoint to your Traffic Manager profile, as an Azure endpoint.
4. Add each additional Web App endpoint to your Traffic Manager profile as an External endpoint. External endpoints can only be added using the Resource Manager deployment model.
5. Create a DNS CNAME record in your vanity domain that points to your Traffic Manager profile DNS name (<...>.trafficmanager.net).
6. Access your site via the vanity domain name, not the Traffic Manager profile DNS name.

Next steps

- Learn [how Traffic Manager works](#).
- Learn about Traffic Manager [endpoint monitoring and automatic failover](#).
- Learn about Traffic Manager [traffic routing methods](#).

Azure Hybrid Use Benefit for Windows Server

1/17/2017 • 4 min to read • [Edit on GitHub](#)

For customers using Windows Server with Software Assurance, you can bring your on-premises Windows Server licenses to Azure and run Windows Server VMs in Azure at a reduced cost. The Azure Hybrid Use Benefit allows you to run Windows Server virtual machines (VMs) in Azure and only get billed for the base compute rate. For more information, please see the [Azure Hybrid Use Benefit licensing page](#). This article explains how to deploy Windows Server VMs in Azure to use this licensing benefit.

Ways to use Azure Hybrid Use Benefit

There are a couple of different ways to deploy Windows VMs with the Azure Hybrid Use Benefit:

1. If you have an Enterprise Agreement subscription, you can [deploy VMs from specific Marketplace images](#) that are pre-configured with Azure Hybrid Use Benefit.
2. Without an Enterprise Agreement, you [upload a custom VM](#) and [deploy using a Resource Manager template](#) or [Azure PowerShell](#).

Deploy a VM using the Azure Marketplace

For customers with [Enterprise Agreement subscriptions](#), images are available in the Marketplace pre-configured with Azure Hybrid Use Benefit. These images can be deployed directly from the Azure portal, Resource Manager templates, or Azure PowerShell, for example. Images in the Marketplace are noted by the [HUB] name as follows:

The screenshot shows the Azure Marketplace search interface. At the top, there's a dark header with the word 'Everything' and a 'Filter' button. Below it is a search bar containing the text 'Windows Server [HUB]'. The main area is titled 'Results' and contains a table with four rows of search results. The columns are labeled 'NAME', 'PUBLISHER', and 'CATEGORY'. Each row shows a small blue icon of a server, followed by the name '[HUB] Windows Server 2016 Datacenter', 'Microsoft', and 'Compute'. The other three rows follow a similar pattern for Windows Server 2012 Datacenter, 2008 R2 SP1, and 2012 R2 Datacenter respectively.

NAME	PUBLISHER	CATEGORY
[HUB] Windows Server 2016 Datacenter	Microsoft	Compute
[HUB] Windows Server 2012 Datacenter	Microsoft	Compute
[HUB] Windows Server 2008 R2 SP1	Microsoft	Compute
[HUB] Windows Server 2012 R2 Datacenter	Microsoft	Compute

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:

```
Get-AzureRMVMImageSku -Location "West US" -Publisher "MicrosoftWindowsServer" `  
-Offer "WindowsServer-HUB"
```

If you don't have an Enterprise Agreement subscription, continue reading for instructions on how to upload a custom VM and deploy with Azure Hybrid Use Benefit.

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 Server

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.

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"
```

NOTE

Microsoft SQL Server, SharePoint Server, and Dynamics can also utilize your Software Assurance licensing. You still 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 an uploaded VM via Resource Manager

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_Server",  
    "hardwareProfile": {  
        "vmSize": "[variables('vmSize')]"  
    },
```

Deploy an uploaded VM via PowerShell quickstart

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_Server"
```

You can [read a more detailed walkthrough on deploying a VM in Azure via PowerShell](#) below, or 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 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:

```
Type          : Microsoft.Compute/virtualMachines
Location     : westus
LicenseType  : Windows_Server
```

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  :
```

Detailed PowerShell deployment walkthrough

The following detailed PowerShell steps show a full deployment of a VM. You can read more context as to the actual cmdlets and different components being created in [Create a Windows VM using Resource Manager and PowerShell](#). You step through creating your resource group, storage account, and virtual networking, then define your VM and finally create your VM.

First, securely obtain credentials, set a location, and resource group name:

```
$cred = Get-Credential
$location = "West US"
$resourceGroupName = "myResourceGroup"
```

Create a public IP:

```
$publicIPName = "myPublicIP"
$publicIP = New-AzureRmPublicIpAddress -Name $publicIPName -ResourceGroupName $resourceGroupName ` 
    -Location $location -AllocationMethod "Dynamic"
```

Define your subnet, NIC, and VNET:

```
$subnetName = "mySubnet"
$nicName = "myNIC"
$vnetName = "myVnet"
$subnetconfig = New-AzureRmVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix 10.0.0.0/8
$vnet = New-AzureRmVirtualNetwork -Name $vnetName -ResourceGroupName $resourceGroupName -Location $location ` 
    -AddressPrefix 10.0.0.0/8 -Subnet $subnetconfig
$nic = New-AzureRmNetworkInterface -Name $nicName -ResourceGroupName $resourceGroupName -Location $location ` 
    -SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $publicIP.Id
```

Name your VM and create a VM config:

```
$vmName = "myVM"
$vmConfig = New-AzureRmVMConfig -VMName $vmName -VMSize "Standard_A1"
```

Define your OS:

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

Add your NIC to the VM:

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

Define the storage account to use:

```
$storageAcc = Get-AzureRmStorageAccount -ResourceGroupName $resourceGroupName -AccountName mystorageaccount
```

Upload your VHD, suitably prepared, and attach to your VM for use:

```
$osDiskName = "licensing.vhd"
$osDiskUri = '{0}vhds/{1}{2}.vhf' -f $storageAcc.PrimaryEndpoints.Blob.ToString(), $vmName.ToLower(), $osDiskName
$urlOfUploadedImageVhd = "https://mystorageaccount.blob.core.windows.net/vhd/myvhd.vhd"
$vm = Set-AzureRmVMOSDisk -VM $vm -Name $osDiskName -VhdUri $osDiskUri -CreateOption FromImage ` 
-SourceImageUri $urlOfUploadedImageVhd -Windows
```

Finally, create your VM and define the licensing type to utilize Azure Hybrid Use Benefit:

```
New-AzureRmVM -ResourceGroupName $resourceGroupName -Location $location -VM $vm -LicenseType "Windows_Server"
```

Next steps

Read more about [Azure Hybrid Use Benefit licensing](#).

Learn more about [using Resource Manager templates](#).

Navigate and select Windows virtual machine images in Azure with PowerShell or the CLI

1/17/2017 • 6 min to read • [Edit on GitHub](#)

This topic describes how to find VM image publishers, offers, skus, and versions for each location into which you might deploy. To give an example, some commonly used Windows VM images are:

Table of commonly used Windows images

PUBLISHERNAME	OFFER	SKU
MicrosoftDynamicsNAV	DynamicsNAV	2015
MicrosoftSharePoint	MicrosoftSharePointServer	2013
MicrosoftSQLServer	SQL2014-WS2012R2	Enterprise-Optimized-for-DW
MicrosoftSQLServer	SQL2014-WS2012R2	Enterprise-Optimized-for-OLTP
MicrosoftWindowsServer	WindowsServer	2012-R2-Datacenter
MicrosoftWindowsServer	WindowsServer	2012-Datacenter
MicrosoftWindowsServer	WindowsServer	2008-R2-SP1
MicrosoftWindowsServer	WindowsServer	Windows-Server-Technical-Preview
MicrosoftWindowsServerEssentials	WindowsServerEssentials	WindowsServerEssentials
MicrosoftWindowsServerHPCPack	WindowsServerHPCPack	2012R2

Azure CLI

NOTE

This article describes how to navigate and select virtual machine images, using a recent installation of either the Azure CLI or Azure PowerShell. As a prerequisite, you would need to change to the Resource Manager mode. With the Azure CLI, enter that mode by typing `azure config mode arm`.

The easiest and quickest way to locate an image to use either with `azure vm quick-create` or to create a resource group template file is to call the `azure vm image list` command and pass the location, the publisher name (it's not case-sensitive!), and an offer -- if you know the offer. For example, the following list is only a short example -- many lists are quite long -- if you know that "Canonical" is a publisher for the "UbuntuServer" offer.

```

azure vm image list westus canonical ubuntuserver
info:  Executing command vm image list
warn:  The parameter --sku if specified will be ignored
+ Getting virtual machine image skus (Publisher:"canonical" Offer:"ubuntuserver" Location:"westus")
data:  Publisher  Offer      Sku          OS    Version      Location  Urn
data:  -----  -----  -----  -----  -----
data:  canonical  ubuntuserver  16.04.0-LTS   Linux  16.04.201604203  westus
canonical:ubuntuserver:16.04.0-LTS:16.04.201604203
data:  canonical  ubuntuserver  16.04.0-LTS   Linux  16.04.201605161  westus
canonical:ubuntuserver:16.04.0-LTS:16.04.201605161
data:  canonical  ubuntuserver  16.04.0-LTS   Linux  16.04.201606100  westus
canonical:ubuntuserver:16.04.0-LTS:16.04.201606100
data:  canonical  ubuntuserver  16.04.0-LTS   Linux  16.04.201606270  westus
canonical:ubuntuserver:16.04.0-LTS:16.04.201606270
data:  canonical  ubuntuserver  16.04.0-LTS   Linux  16.04.201607210  westus
canonical:ubuntuserver:16.04.0-LTS:16.04.201607210
data:  canonical  ubuntuserver  16.04.0-LTS   Linux  16.04.201608150  westus
canonical:ubuntuserver:16.04.0-LTS:16.04.201608150
data:  canonical  ubuntuserver  16.10-DAILY   Linux  16.10.201607220  westus
canonical:ubuntuserver:16.10-DAILY:16.10.201607220
data:  canonical  ubuntuserver  16.10-DAILY   Linux  16.10.201607230  westus
canonical:ubuntuserver:16.10-DAILY:16.10.201607230
data:  canonical  ubuntuserver  16.10-DAILY   Linux  16.10.201607240  westus
canonical:ubuntuserver:16.10-DAILY:16.10.201607240

```

The **Urn** column will be the form you pass to `azure vm quick-create`.

Often, however, you don't yet know what is available. In this case, you can navigate images by discovering publishers first by using `azure vm image list-publishers` and responding to the location prompt with a data center location you expect to use for your resource group. For example, the following lists all image publishers in the West US location (pass the location argument by lowercasing and removing spaces from the standard locations)

```

azure vm image list-publishers
info:  Executing command vm image list-publishers
Location: westus
+ Getting virtual machine and/or extension image publishers (Location: "westus")
data:  Publisher          Location
data:  -----  -----
data:  a10networks        westus
data:  aiscaler-cache-control-ddos-and-url-rewriting- westus
data:  alertlogic          westus
data:  AlertLogic.Extension  westus

```

These lists can be quite long, so the example list above is just a snippet. Let's say that I noticed that Canonical is, indeed, an image publisher in the West US location. You can now find their offers by calling

`azure vm image list-offers` and pass the location and the publisher at the prompts, like the following example:

```

azure vm image list-offers
info:  Executing command vm image list-offers
Location: westus
Publisher: canonical
+ Getting virtual machine image offers (Publisher: "canonical" Location:"westus")
data:  Publisher  Offer          Location
data:  -----  -----  -----
data:  canonical  Ubuntu15.04Snappy    westus
data:  canonical  Ubuntu15.04SnappyDocker  westus
data:  canonical  UbunturollingSnappy   westus
data:  canonical  UbuntuServer         westus
data:  canonical  Ubuntu_Snappy_Core   westus
data:  canonical  Ubuntu_Snappy_Core_Docker  westus
info:  vm image list-offers command OK

```

Now we know that in the West US region, Canonical publishes the **UbuntuServer** offer on Azure. But what SKUs? To get those, you call `azure vm image list-skus` and respond to the prompt with the location, publisher, and offer that you have discovered.

```
azure vm image list-skus
info:  Executing command vm image list-skus
Location: westus
Publisher: canonical
Offer: ubuntuserver
+ Getting virtual machine image skus (Publisher:"canonical" Offer:"ubuntuserver" Location:"westus")
data:  Publisher  Offer      sku           Location
data:  -----    -----  -----  -----
data:  canonical  ubuntuserver  12.04.2-LTS   westus
data:  canonical  ubuntuserver  12.04.3-LTS   westus
data:  canonical  ubuntuserver  12.04.4-LTS   westus
data:  canonical  ubuntuserver  12.04.5-DAILY-LTS  westus
data:  canonical  ubuntuserver  12.04.5-LTS   westus
data:  canonical  ubuntuserver  12.10       westus
data:  canonical  ubuntuserver  14.04-beta   westus
data:  canonical  ubuntuserver  14.04.0-LTS  westus
data:  canonical  ubuntuserver  14.04.1-LTS  westus
data:  canonical  ubuntuserver  14.04.2-LTS  westus
data:  canonical  ubuntuserver  14.04.3-LTS  westus
data:  canonical  ubuntuserver  14.04.4-DAILY-LTS  westus
data:  canonical  ubuntuserver  14.04.4-LTS   westus
data:  canonical  ubuntuserver  14.04.5-DAILY-LTS  westus
data:  canonical  ubuntuserver  14.04.5-LTS   westus
data:  canonical  ubuntuserver  14.10       westus
data:  canonical  ubuntuserver  14.10-beta   westus
data:  canonical  ubuntuserver  14.10-DAILY  westus
data:  canonical  ubuntuserver  15.04       westus
data:  canonical  ubuntuserver  15.04-beta   westus
data:  canonical  ubuntuserver  15.04-DAILY  westus
data:  canonical  ubuntuserver  15.10       westus
data:  canonical  ubuntuserver  15.10-alpha  westus
data:  canonical  ubuntuserver  15.10-beta   westus
data:  canonical  ubuntuserver  15.10-DAILY  westus
data:  canonical  ubuntuserver  16.04-alpha  westus
data:  canonical  ubuntuserver  16.04-beta   westus
data:  canonical  ubuntuserver  16.04.0-DAILY-LTS  westus
data:  canonical  ubuntuserver  16.04.0-LTS   westus
data:  canonical  ubuntuserver  16.10-DAILY  westus
info:  vm image list-skus command OK
```

With this information, you can now find exactly the image you want by calling the original call at the top.

```

azure vm image list westus canonical ubuntuserver 16.04.0-LTS
info: Executing command vm image list
+ Getting virtual machine images (Publisher:"canonical" Offer:"ubuntuserver" Sku: "16.04.0-LTS"
Location:"westus")
data: Publisher Offer Sku OS Version Location Urn
data: ----- -----
-----
data: canonical ubuntuserver 16.04.0-LTS Linux 16.04.201604203 westus canonical:ubuntuserver:16.04.0-
LTS:16.04.201604203
data: canonical ubuntuserver 16.04.0-LTS Linux 16.04.201605161 westus canonical:ubuntuserver:16.04.0-
LTS:16.04.201605161
data: canonical ubuntuserver 16.04.0-LTS Linux 16.04.201606100 westus canonical:ubuntuserver:16.04.0-
LTS:16.04.201606100
data: canonical ubuntuserver 16.04.0-LTS Linux 16.04.201606270 westus canonical:ubuntuserver:16.04.0-
LTS:16.04.201606270
data: canonical ubuntuserver 16.04.0-LTS Linux 16.04.201607210 westus canonical:ubuntuserver:16.04.0-
LTS:16.04.201607210
data: canonical ubuntuserver 16.04.0-LTS Linux 16.04.201608150 westus canonical:ubuntuserver:16.04.0-
LTS:16.04.201608150
info: vm image list command OK

```

Now you can choose precisely the image you want to use. To create a virtual machine quickly by using the URN information, which you just found, or to use a template with that URN information, see [Using the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager](#).

PowerShell

NOTE

Install and configure the [latest Azure PowerShell](#). If you are using Azure PowerShell modules below 1.0, you still use the following commands but you must first `Switch-AzureMode AzureResourceManager`.

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, these values are needed for the `Set-AzureRMVMSourceImage` PowerShell cmdlet or with a resource group template file in which you must specify the type of virtual machine to be created.

If you need to determine these values, you can navigate the images to 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 display 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:

```
PS C:\> $locName="West US"  
PS C:\> Get-AzureRMVMImagePublisher -Location $locName | Select PublisherName  
  
PublisherName  
-----  
a10networks  
aiscaler-cache-control-ddos-and-url-rewriting-  
alertlogic  
AlertLogic.Extension  
Barracuda.Azure.ConnectivityAgent  
barracudanetworks  
basho  
boxless  
bssw  
Canonical  
...  
...
```

For the "MicrosoftWindowsServer" publisher:

```
PS C:\> $pubName="MicrosoftWindowsServer"  
PS C:\> Get-AzureRMVMImageOffer -Location $locName -Publisher $pubName | Select Offer  
  
Offer  
----  
WindowsServer
```

For the "WindowsServer" offer:

```
PS C:\> $offerName="WindowsServer"  
PS C:\> Get-AzureRMVMImageSku -Location $locName -Publisher $pubName -Offer $offerName | Select Skus  
  
Skus  
----  
2008-R2-SP1  
2012-Datacenter  
2012-R2-Datacenter  
2016-Nano-Server-Technical-Preview  
2016-Technical-Preview-with-Conta  
Windows-Server-Technical-Preview
```

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.

Using Windows client in Azure for dev/test scenarios

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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, we have made certain Windows 10 images 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

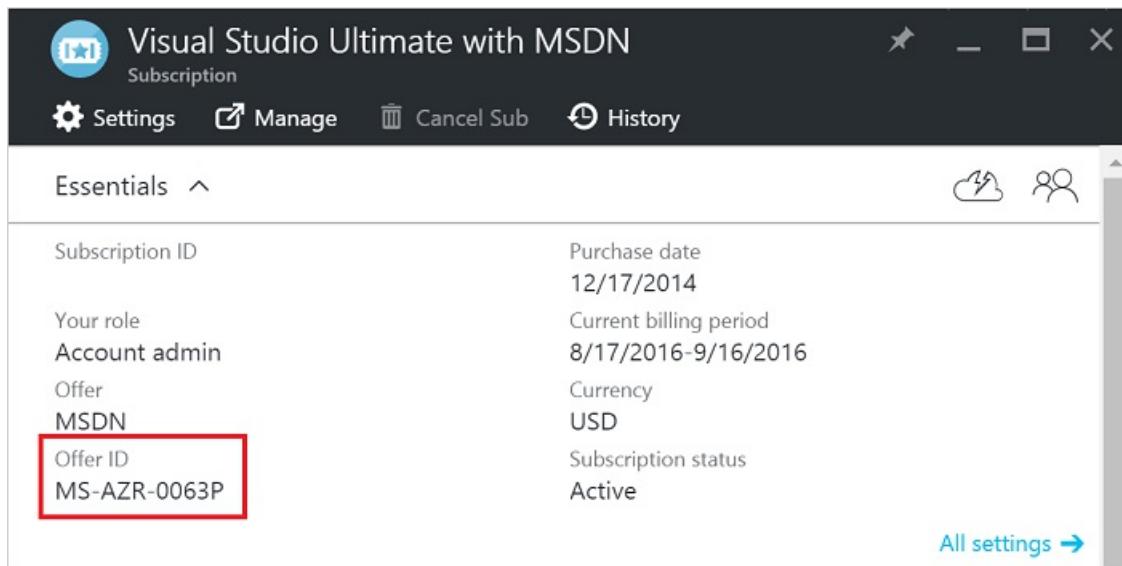
The following table details the offer IDs that are eligible to deploy Windows 10 through the Azure Gallery. The Windows 10 images are only visible to the following offers. Visual Studio subscribers who need to run Windows client in a different offer type require you to [adequately prepare and create](#) 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 or the Account portal.

The subscription offer ID is noted on the 'Subscriptions' blade within the Azure portal:



Visual Studio Ultimate with MSDN

Subscription

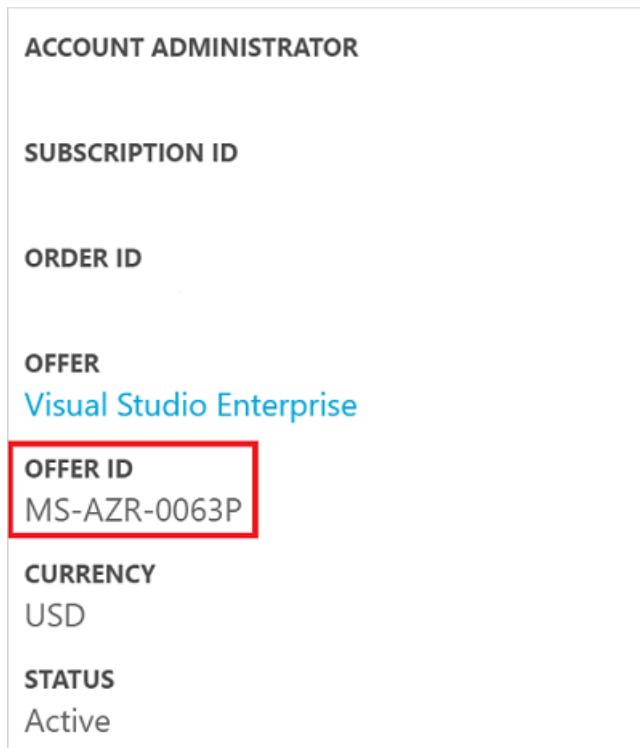
Settings Manage Cancel Sub History

Essentials ^

Subscription ID	Purchase date 12/17/2014
Your role Account admin	Current billing period 8/17/2016-9/16/2016
Offer MSDN	Currency USD
Offer ID MS-AZR-0063P	Subscription status Active

All settings →

You can also view the offer ID from the '[Subscriptions](#)' tab of the Azure Account portal:



ACCOUNT ADMINISTRATOR

SUBSCRIPTION ID

ORDER ID

OFFER
[Visual Studio Enterprise](#)

OFFER ID
MS-AZR-0063P

CURRENCY
USD

STATUS
Active

Next steps

You can now deploy your VMs using [PowerShell](#), [Resource Manager templates](#), or [Visual Studio](#).

Create a Windows virtual machine with a Resource Manager template

This article shows you how to deploy an Azure Resource Manager template using PowerShell. The [template](#) 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.

Step 1: 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.

Step 2: 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 **Central US** location:

```
New-AzureRmResourceGroup -Name "myResourceGroup" -Location "Central US"
```

You should see something like this example:

```
ResourceGroupName : myResourceGroup
Location        : centralus
ProvisioningState : Succeeded
Tags            :
ResourceId      : /subscriptions/{subscription-id}/resourceGroups/myResourceGroup
```

Step 3: Create the resources

Deploy the template and provide parameter values when prompted. This example deploys the 101-vm-simple-windows template in the resource group that you created:

```
New-AzureRmResourceGroupDeployment -ResourceGroupName "myResourceGroup" -TemplateUri
"https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/101-vm-simple-
windows/azuredeploy.json"
```

You are asked to provide the name of the administrator account on the VM, the password of the account, and the

DNS prefix.

You should see something like this example:

```
DeploymentName      : azuredploy
ResourceGroupName   : myResourceGroup
ProvisioningState   : Succeeded
Timestamp          : 12/29/2016 8:11:37 PM
Mode                : Incremental
TemplateLink        :
                      Uri           : https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/
                      101-vm-simple-windows/azuredploy.json
                      ContentVersion : 1.0.0.0
Parameters          :
                      Name          Type          Value
                      ======       ======       =====
                      adminUsername String        myAdminUser
                      adminPassword SecureString
                      dnsLabelPrefix String        myDomain
                      windowsOSVersion String      2016-Datacenter
Outputs             :
                      Name          Type          Value
                      ======       ======       =====
                      hostname     String        myDomain.centralus.cloudapp.azure.com
DeploymentLogLevel :
```

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, a next step would be to look at [Troubleshoot common Azure deployment errors with Azure Resource Manager](#)
- Learn how to manage the virtual machine that you created by reviewing [Manage virtual machines using Azure Resource Manager and PowerShell](#).

Creating Virtual Machine Scale Sets using PowerShell cmdlets

1/17/2017 • 3 min to read • [Edit on GitHub](#)

This is an example of how to create a Virtual Machine Scale Set(VMSS), it creates a VMSS of 3 nodes, with all the associated Networking and Storage.

First Steps

Ensure you have the latest Azure PowerShell module installed, this will contain the PowerShell commandlets needed to maintain and create VMSS. Go to the commandline tools [here](#) for the latest available Azure Modules.

To find VMSS related commandlets, use the search string *VMSS*.

Creating a VMSS

Create Resource Group

```
$loc = 'westus';
$rgname = 'mynewrgwu';
New-AzureRmResourceGroup -Name $rgname -Location $loc -Force;
```

Create Storage Account

Set storage account type / name.

```
$stoname = 'sto' + $rgname;
$stotype = 'Standard_LRS';
New-AzureRmStorageAccount -ResourceGroupName $rgname -Name $stoname -Location $loc -SkuName $stotype -Kind "Storage";

$stoaccount = Get-AzureRmStorageAccount -ResourceGroupName $rgname -Name $stoname;
```

Create Networking (VNET / Subnet)

Subnet Specification

```
$subnetName = 'websubnet'
$subnet = New-AzureRmVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix "10.0.0.0/24";
```

VNET Specification

```
$vnet = New-AzureRmVirtualNetwork -Force -Name ('vnet' + $rgname) -ResourceGroupName $rgname -Location $loc -AddressPrefix "10.0.0.0/16" -Subnet $subnet;
$vnet = Get-AzureRmVirtualNetwork -Name ('vnet' + $rgname) -ResourceGroupName $rgname;

#In this case below we assume the new subnet is the only one, note difference if you have one already or have adjusted this code to more than one subnet.
$subnetId = $vnet.Subnets[0].Id;
```

Create Public IP Resource to Allow External Access

This will be bound to the Load Balancer.

```
$pubip = New-AzureRmPublicIpAddress -Force -Name ('pubip' + $rgname) -ResourceGroupName $rgname -Location $loc -AllocationMethod Dynamic -DomainNameLabel ('pubip' + $rgname);
$pubip = Get-AzureRmPublicIpAddress -Name ('pubip' + $rgname) -ResourceGroupName $rgname;
```

Create and Configure Load Balancer

```
$frontendName = 'fe' + $rgname
$backendAddressPoolName = 'bepool' + $rgname
$probeName = 'vmssprobe' + $rgname
$inboundNatPoolName = 'innatpool' + $rgname
$lbruleName = 'lbrule' + $rgname
$lbName = 'vmsslb' + $rgname

#Bind Public IP to Load Balancer
$frontend = New-AzureRmLoadBalancerFrontendIpConfig -Name $frontendName -PublicIpAddress $pubip
```

Configure Load Balancer

Create Backend Address Pool Config, this will be shared by the NICs of the VMs in VMSS.

```
$backendAddressPool = New-AzureRmLoadBalancerBackendAddressPoolConfig -Name $backendAddressPoolName
```

Set Load Balanced Probe Port, change the settings as appropriate for your application.

```
$probe = New-AzureRmLoadBalancerProbeConfig -Name $probeName -RequestPath healthcheck.aspx -Protocol http -Port 80 -IntervalInSeconds 15 -ProbeCount 2
```

Create NAT rules for direct routed connectivity (not load balanced) to the VMs in the VMSS via the Load Balancer, note this demonstrates using RDP, this is just for demonstration and internal VNET methods should be used for RDP'ing to these servers.

```
$frontendpoolrangepstart = 3360
$frontendpoolrangeend = 3370
$backendvmpport = 3389
$inboundNatPool = New-AzureRmLoadBalancerInboundNatPoolConfig -Name $inboundNatPoolName -
FrontendIPConfigurationId ` 
$frontend.Id -Protocol Tcp -FrontendPortRangeStart $frontendpoolrangepstart -FrontendPortRangeEnd
$frontendpoolrangeend -BackendPort $backendvmpport;
```

Create the Load Balanced Rule, this example shows load balancing port 80 requests, using the settings from previous steps.

```
$protocol = 'Tcp'
$feLBPort = 80
$beLBPort = 80

$lbrule = New-AzureRmLoadBalancerRuleConfig -Name $lbruleName ` 
-FrontendIPConfiguration $frontend -BackendAddressPool $backendAddressPool ` 
-Probe $probe -Protocol $protocol -FrontendPort $feLBPort -BackendPort $beLBPort ` 
-IdleTimeoutInMinutes 15 -EnableFloatingIP -LoadDistribution SourceIP -Verbose;
```

Create Load Balancer with configuration.

```
$actualLb = New-AzureRmLoadBalancer -Name $lbName -ResourceGroupName $rgname -Location $loc ` 
-FrontendIpConfiguration $frontend -BackendAddressPool $backendAddressPool ` 
-Probe $probe -LoadBalancingRule $lbrule -InboundNatPool $inboundNatPool -Verbose;
```

Check LB settings, check load balanced port configs, note, you will not see Inbound NAT rules until the VM's in the VMSS are created.

```
$expectedLb = Get-AzureRmLoadBalancer -Name $lbName -ResourceGroupName $rgname
```

Configure and Create VMSS

Note, this infrastructure example shows how to setup distribute and scale web traffic across the VMSS, but the VMs

Images specified here do not have any web services installed.

```
#specify VMSS Name
$vmssName = 'vmss' + $rgname;

##specify VMSS specific details
$adminUsername = 'azadmin';
$adminPassword = "Password1234!";

$PublisherName = 'MicrosoftWindowsServer'
$Offer        = 'WindowsServer'
$Sku          = '2012-R2-Datacenter'
$Version      = 'latest'
$vmNamePrefix = 'winvmss'

$vhdContainer = "https://" + $stoname + ".blob.core.windows.net/" + $vmssName;

####add an extension
$extname = 'BGInfo';
$publisher = 'Microsoft.Compute';
$exttype = 'BGInfo';
$extver = '2.1';
```

Bind NIC to Load Balancer and Subnet

```
$ipCfg = New-AzureRmVmssIPConfig -Name 'nic' ` 
-LoadBalancerInboundNatPoolsId $actualLb.InboundNatPools[0].Id ` 
-LoadBalancerBackendAddressPoolsId $actualLb.BackendAddressPools[0].Id ` 
-SubnetId $subnetId;
```

Create VMSS Config

```
#Specify number of nodes
$numberofnodes = 3

$vmss = New-AzureRmVmssConfig -Location $loc -SkuCapacity $numberofnodes -SkuName 'Standard_A2' - 
UpgradePolicyMode 'automatic' ` 
| Add-AzureRmVmssNetworkInterfaceConfiguration -Name $subnetName -Primary $true -IPConfiguration $ipCfg ` 
| Set-AzureRmVmssOSProfile -ComputerNamePrefix $vmNamePrefix -AdminUsername $adminUsername -AdminPassword $adminPassword ` 
| Set-AzureRmVmssStorageProfile -Name 'test' -OsDiskCreateOption 'FromImage' -OsDiskCaching 'None' ` 
-ImageReferenceOffer $Offer -ImageReferenceSku $Sku -ImageReferenceVersion $Version ` 
-ImageReferencePublisher $PublisherName -VhdContainer $vhdContainer ` 
| Add-AzureRmVmssExtension -Name $extname -Publisher $publisher -Type $exttype -TypeHandlerVersion $extver - 
AutoUpgradeMinorVersion $true
```

Build VMSS Config

```
New-AzureRmVmss -ResourceGroupName $rgname -Name $vmssName -VirtualMachineScaleSet $vmss -Verbose;
```

Now you have created the VMSS. You can test connecting to the individual VM using RDP in this example:

```
VM0 : pubipmynewrgwu.westus.cloudapp.azure.com:3360
VM1 : pubipmynewrgwu.westus.cloudapp.azure.com:3361
VM2 : pubipmynewrgwu.westus.cloudapp.azure.com:3362
```

Create multiple Azure virtual machines

1/17/2017 • 2 min to read • [Edit on GitHub](#)

There are many scenarios where you need to create a large number of similar virtual machines (VMs). Some examples include high-performance computing (HPC), large-scale data analysis, scalable and often stateless middle-tier or backend servers (such as webservers), and distributed databases.

This article discusses the available options to create multiple VMs in Azure. These options go beyond the simple cases where you manually create a series of VMs. To create many VMs, the processes that you typically use don't scale well if you need to create more than a handful of VMs.

One way to create many similar VMs is to use the Azure Resource Manager construct of *resource loops*.

Resource loops

Resource loops are a syntactical shorthand in Azure Resource Manager templates. Resource loops can create a set of similarly configured resources in a loop. You can use resource loops to create multiple storage accounts, network interfaces, or virtual machines. For more information about resource loops, refer to [Create VMs in availability sets using resource loops](#).

Challenges of scale

Although resource loops make it easier to build out a cloud infrastructure at scale and produce more concise templates, certain challenges remain. For example, if you use a resource loop to create 100 virtual machines, you need to correlate network interface controllers (NICs) with corresponding VMs and storage accounts. Because the number of VMs is likely to be different from the number of storage accounts, you'll have to deal with different resource loop sizes, too. These are solvable problems, but the complexity increases significantly with scale.

Another challenge occurs when you need an infrastructure that scales elastically. For example, you might want an autoscale infrastructure that automatically increases or decreases the number of VMs in response to workload. VMs don't provide an integrated mechanism to vary in number (scale out and scale in). If you scale in by removing VMs, it's difficult to guarantee high availability by making sure that VMs are balanced across update and fault domains.

Finally, when you use a resource loop, multiple calls to create resources go to the underlying fabric. When multiple calls create similar resources, Azure has an implicit opportunity to improve upon this design and optimize deployment reliability and performance. This is where *virtual machine scale sets* come in.

Virtual machine scale sets

Virtual machine scale sets are an Azure Compute resource to deploy and manage a set of identical VMs. With all VMs configured the same, VM scale sets are easy to scale in and scale out. You simply change the number of VMs in the set. You can also configure VM scale sets to autoscale based on the demands of the workload.

For applications that need to scale Compute resources out and in, scale operations are implicitly balanced across fault and update domains.

Instead of correlating multiple resources such as NICs and VMs, a VM scale set has network, storage, virtual machine, and extension properties that you can configure centrally.

For an introduction to VM scale sets, refer to the [Virtual machine scale sets product page](#). For more detailed information, go to the [Virtual machines scale sets documentation](#).

Create a Windows Virtual machine with monitoring and diagnostics using Azure Resource Manager Template

1/17/2017 • 8 min to read • [Edit on GitHub](#)

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": "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('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 **IaaS.Diagnostics** 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 classic portal portal ensuring that the monitoring charts show up correctly in the Azure classic 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](#)

Creating and deploying Azure resource groups through Visual Studio

1/17/2017 • 9 min to read • [Edit on GitHub](#)

With Visual Studio and the [Azure SDK](#), you can create a project that deploys your infrastructure and code to Azure. For example, you can define the web host, web site, and database for your app, and deploy that infrastructure along with the code. Or, you can define a Virtual Machine, Virtual Network and Storage Account, and deploy that infrastructure along with a script that is executed on Virtual Machine. The **Azure Resource Group** deployment project enables you to deploy all the needed resources in a single, repeatable operation. For more information about deploying and managing your resources, see [Azure Resource Manager overview](#).

Azure Resource Group projects contain Azure Resource Manager JSON templates, which define the resources that you deploy to Azure. To learn about the elements of the Resource Manager template, see [Authoring Azure Resource Manager templates](#). Visual Studio enables you to edit these templates, and provides tools that simplify working with templates.

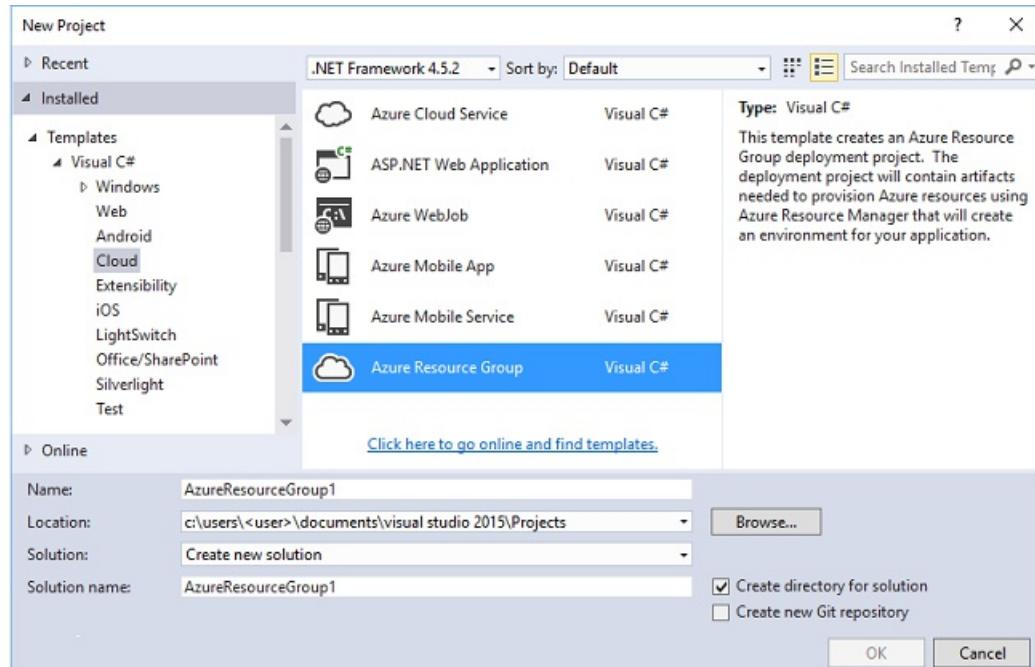
In this topic, you deploy a web app and SQL Database. However, the steps are almost the same for any type resource. You can as easily deploy a Virtual Machine and its related resources. Visual Studio provides many different starter templates for deploying common scenarios.

This article shows Visual Studio 2015 Update 2 and Microsoft Azure SDK for .NET 2.9. If you use Visual Studio 2013 with Azure SDK 2.9, your experience is largely the same. You can use versions of the Azure SDK from 2.6 or later; however, your experience of the user interface may be different than the user interface shown in this article. We strongly recommend that you install the latest version of the [Azure SDK](#) before starting the steps.

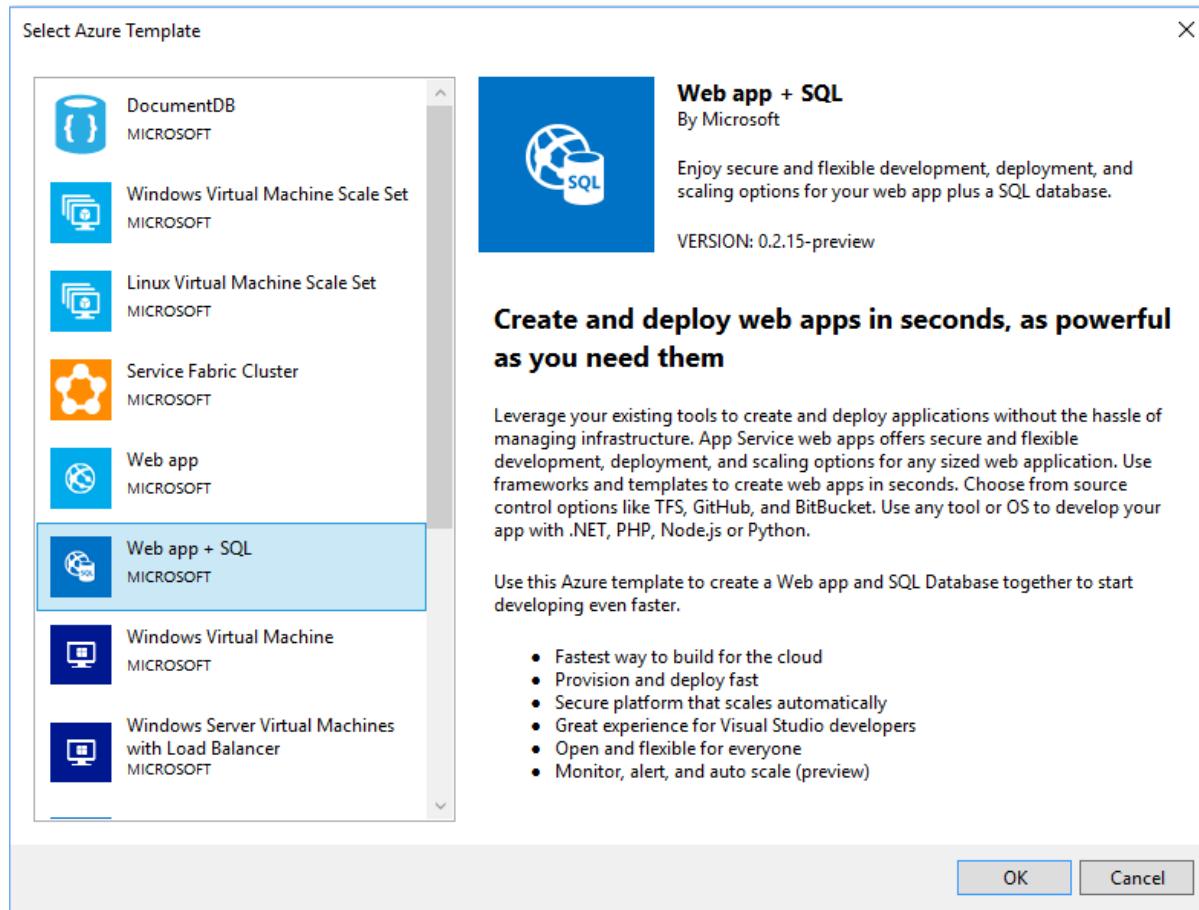
Create Azure Resource Group project

In this procedure, you create an Azure Resource Group project with a **Web app + SQL** template.

1. In Visual Studio, choose **File, New Project**, choose **C#** or **Visual Basic**. Then choose **Cloud**, and then choose **Azure Resource Group** project.



- Choose the template that you want to deploy to Azure Resource Manager. Notice there are many different options based on the type of project you wish to deploy. For this topic, choose the **Web app + SQL** template.



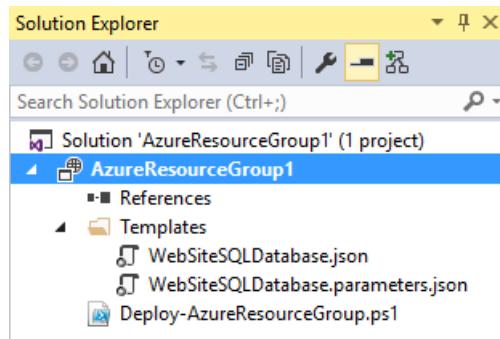
The template you pick is just a starting point; you can add and remove resources to fulfill your scenario.

NOTE

Visual Studio retrieves a list of available templates online. The list may change.

Visual Studio creates a resource group deployment project for the web app and SQL database.

- To see what you created, expand the nodes in the deployment project.



Since we chose the Web app + SQL template for this example, you see the following files:

FILE NAME	DESCRIPTION
-----------	-------------

FILE NAME	DESCRIPTION
Deploy-AzureResourceGroup.ps1	A PowerShell script that invokes PowerShell commands to deploy to Azure Resource Manager. Note Visual Studio uses this PowerShell script to deploy your template. Any changes you make to this script affect deployment in Visual Studio, so be careful.
WebSiteSQLDatabase.json	The Resource Manager template that defines the infrastructure you want to deploy to Azure, and the parameters you can provide during deployment. It also defines the dependencies between the resources so Resource Manager deploys the resources in the correct order.
WebSiteSQLDatabase.parameters.json	A parameters file that contains values needed by the template. You pass in parameter values to customize each deployment.

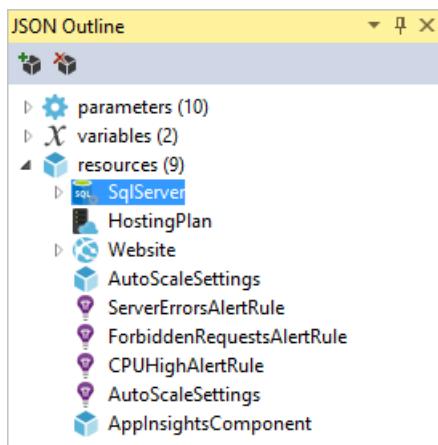
All resource group deployment projects contain these basic files. Other projects may contain additional files to support other functionality.

Customize the Resource Manager template

You can customize a deployment project by modifying the JSON templates that describe the resources you want to deploy. JSON stands for JavaScript Object Notation, and is a serialized data format that is easy to work with. The JSON files use a schema that you reference at the top of each file. If you want to understand the schema, you can download and analyze it. The schema defines what elements are valid, the types and formats of fields, the possible values of enumerated values, and so on. To learn about the elements of the Resource Manager template, see [Authoring Azure Resource Manager templates](#).

To work on your template, open **WebSiteSQLDatabase.json**.

The Visual Studio editor provides tools to assist you with editing the Resource Manager template. The **JSON Outline** window makes it easy to see the elements defined in your template.



Selecting any of the elements in the outline takes you to that part of the template and highlights the corresponding JSON.

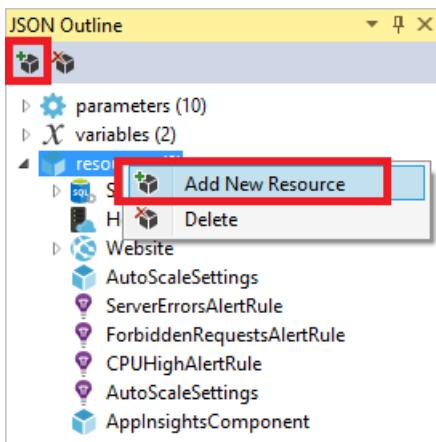
JSON Outline

```

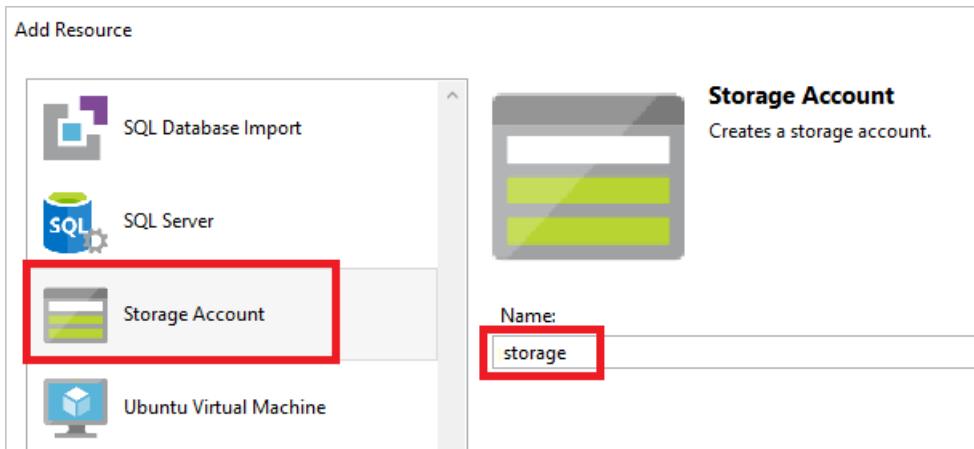
Schema: http://schema.management.azure.com/schemas/2015-01-01
{
  "name": "[parameters('hostingPlanName')]",
  "type": "Microsoft.Web/sites",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Web/serverFarms/',"
  ],
  "tags": {
    "[concat('hidden-related:', resourceGroup().name, 'Website')]"
  },
  "properties": {
    "name": "[variables('webSiteName')]",
    "serverFarmId": "[resourceId('Microsoft.Web/serverFarms',"
  }
}

```

You can add a resource by either selecting the **Add Resource** button at the top of the JSON Outline window, or by right-clicking **resources** and selecting **Add New Resource**.



For this tutorial, select **Storage Account** and give it a name. Provide a name that is no more than 11 characters, and only contains numbers and lower-case letters.



Notice that not only was the resource added, but also a parameter for the type storage account, and a variable for the name of the storage account.

JSON Outline

```

parameters (11)
  hostingPlanName
  skuName
  skuCapacity
  administratorLogin
  administratorLoginPassword
  databaseName
  collation
  edition
  maxSizeBytes
  requestedServiceObjectiveName
  storageType

variables (3)
  webSiteName
  sqlserverName
  storageName

resources (10)
  SqlServer
  HostingPlan
  Website
    connectionstrings
    AutoScaleSettings
    ServerErrorsAlertRule
    ForbiddenRequestsAlertRule
    CPUHighAlertRule
    AutoScaleSettings
    AppInsightsComponent
  storage

```

The **storageType** parameter is pre-defined with allowed types and a default type. You can leave these values or edit them for your scenario. If you do not want anyone to deploy a **Premium_LRS** storage account through this template, remove it from the allowed types.

```

"storageType": {
  "type": "string",
  "defaultValue": "Standard_LRS",
  "allowedValues": [
    "Standard_LRS",
    "Standard_ZRS",
    "Standard_GRS",
    "Standard_RAGRS"
  ]
}

```

Visual Studio also provides intellisense to help you understand what properties are available when editing the template. For example, to edit the properties for your App Service plan, navigate to the **HostingPlan** resource, and add a value for the **properties**. Notice that intellisense shows the available values and provides a description of that value.

```
{
  "apiVersion": "2015-08-01",
  "name": "[parameters('hostingPlanName')]",
  "type": "Microsoft.Web/serverfarms",
  "location": "[resourceGroup().location]",
  "tags": {
    "displayName": "HostingPlan"
  },
  "sku": {
    "name": "[parameters('skuName')]",
    "capacity": "[parameters('skuCapacity')]"
  },
  "properties": {
    "name": "[parameters('hostingPlanName')]",
    "numberOfWorkers": 1,
    "accessLevel": "Full"
  }
}
```

Microsoft.Web/serverfarms: The instance count, which is the number of workers per instance. Valid values are 1-10.

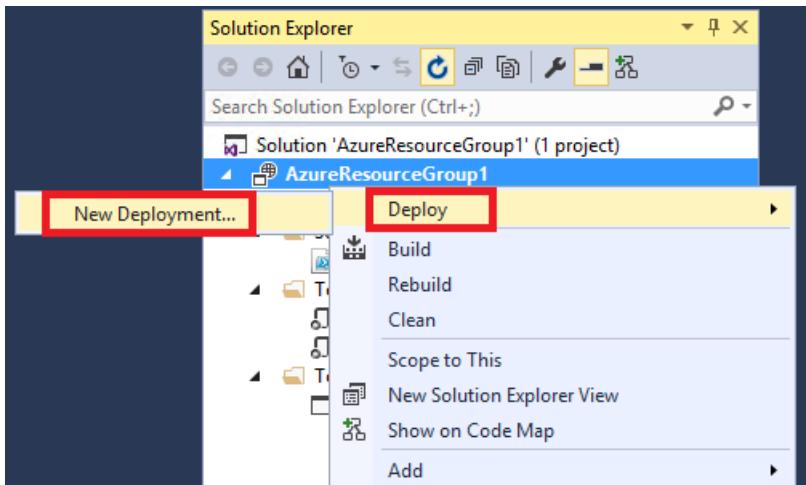
You can set **numberOfWorkers** to 1.

```
"properties": {
  "name": "[parameters('hostingPlanName')]",
  "numberOfWorkers": 1
}
```

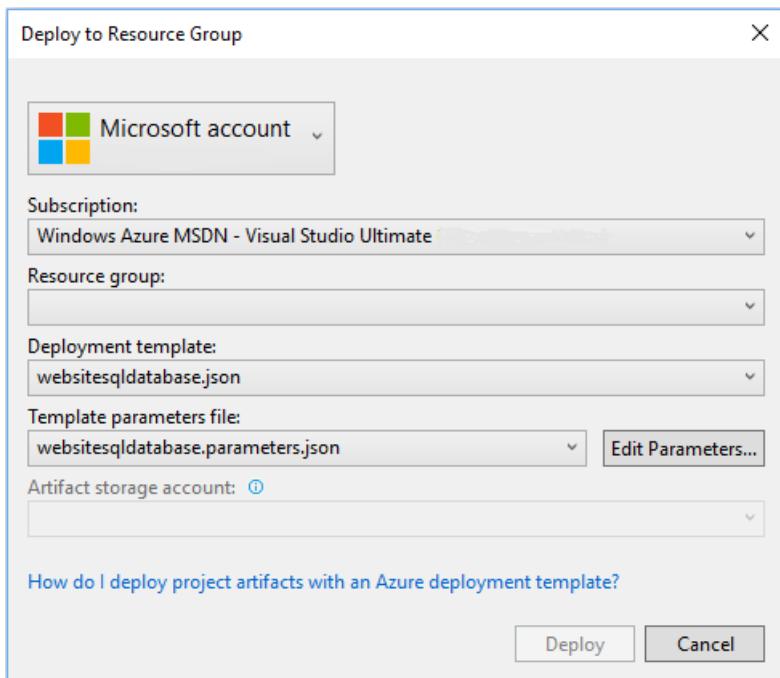
Deploy the Resource Group project to Azure

You are now ready to deploy your project. When you deploy an Azure Resource Group project, you deploy it to an Azure resource group. The resource group is a logical grouping of resources that share a common lifecycle.

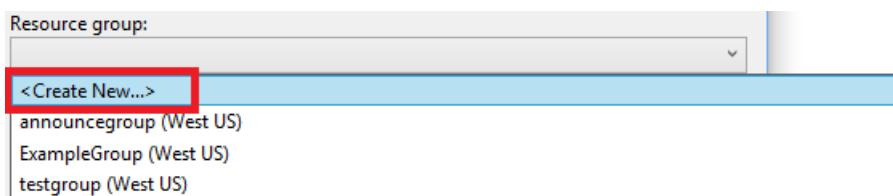
1. On the shortcut menu of the deployment project node, choose **Deploy > New Deployment**.



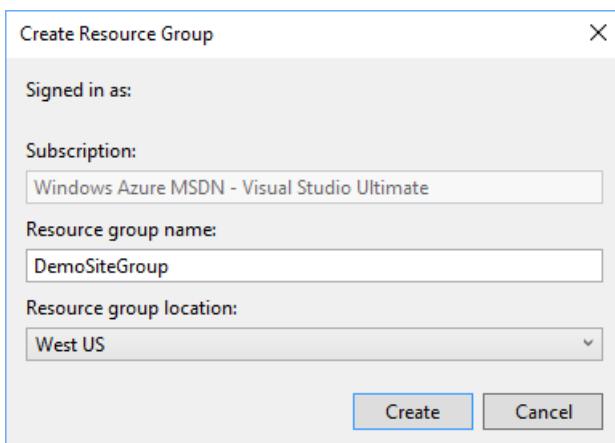
The **Deploy to Resource Group** dialog box appears.



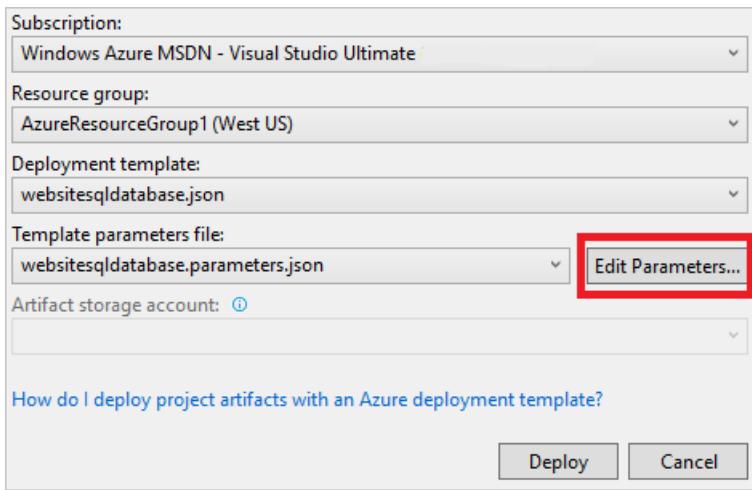
2. In the **Resource group** dropdown box, choose an existing resource group or create a new one. To create a resource group, open the **Resource Group** dropdown box and choose **Create New**.



The **Create Resource Group** dialog box appears. Give your group a name and location, and select the **Create** button.



3. Edit the parameters for the deployment by selecting the **Edit Parameters** button.



4. Provide values for the empty parameters and select the **Save** button. The empty parameters are **hostingPlanName**, **administratorLogin**, **administratorLoginPassword**, and **databaseName**.

hostingPlanName specifies a name for the [App Service plan](#) to create.

administratorLogin specifies the user name for the SQL Server administrator. Do not use common admin names like **sa** or **admin**.

The **administratorLoginPassword** specifies a password for SQL Server administrator. The **Save passwords as plain text in the parameters file** option is not secure; therefore, do not select this option. Since the password is not saved as plain text, you need to provide this password again during deployment.

databaseName specifies a name for the database to create.

Edit Parameters

The following parameter values will be used for this deployment:

Parameter Name	Value
hostingPlanName	DemoSitePlan
skuName	F1
skuCapacity	1
administratorLogin	DemoAdmin
administratorLoginPassword	*****
databaseName	DemoDatabase
collation	SQL_Latin1_General_CI_AS
edition	Basic
maxSizeBytes	1073741824
requestedServiceObjectiveName	Basic
storageType	Standard_LRS

Save passwords as plain text in the parameters file

Save **Cancel**

5. Choose the **Deploy** button to deploy the project to Azure. A PowerShell console opens outside of the Visual Studio instance. Enter the SQL Server administrator password in the PowerShell console when prompted. **Your PowerShell console may be hidden behind other items or minimized in the task bar.** Look for this console and select it to provide the password.

NOTE

Visual Studio may ask you to install the Azure PowerShell cmdlets. You need the Azure PowerShell cmdlets to successfully deploy resource groups. If prompted, install them.

6. The deployment may take a few minutes. In the **Output** windows, you see the status of the deployment. When the deployment has finished, the last message indicates a successful deployment with something

similar to:

```
...
18:00:58 - Successfully deployed template 'c:\users\user\documents\visual studio 2015\projects\azureresourcegroup1\azureresourcegroup1\templates\websitesqldatabase.json' to resource group 'DemoSiteGroup'.
```

7. In a browser, open the [Azure portal](#) and sign in to your account. To see the resource group, select **Resource groups** and the resource group you deployed to.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with options like 'New', 'Resource groups' (which is highlighted with a red box), 'All resources', 'Recent', 'App Services', 'SQL databases', and 'Virtual machines (classic)'. The main area is titled 'Resource groups' and shows a list of resource groups. The 'DemoSiteGroup' entry is also highlighted with a red box. The list includes:

- DemoSiteGroup
- ExampleGroup
- testgroup

8. You see all the deployed resources. Notice that the name of the storage account is not exactly what you specified when adding that resource. The storage account must be unique. The template automatically adds a string of characters to the name you provided to provide a unique name.

The screenshot shows the details of the 'DemoSiteGroup' resource group. At the top, it displays the group name and a 'Resource group' icon. Below that are three buttons: 'Settings', 'Add', and 'Delete'. The 'Essentials' tab is selected, showing the following information:

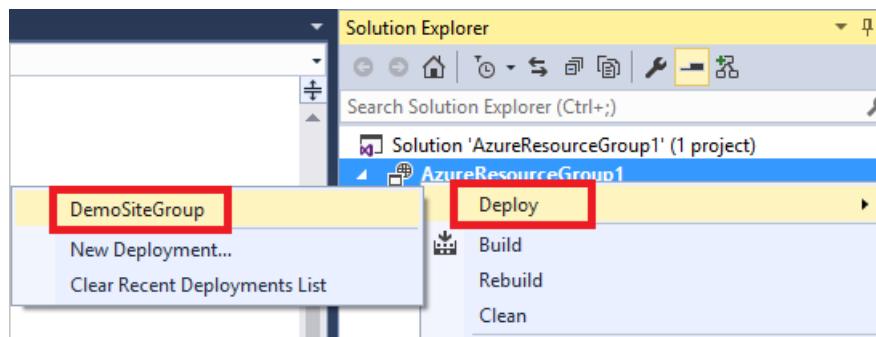
Subscription name	Subscription ID
Windows Azure MSDN - Visual Studio Ulti...	
Last deployment	Location
2/12/2016 (Succeeded)	West US

Below the essentials tab is a blue 'All settings →' button. The 'Summary' section lists the following resources:

- webSitekzzwhf4j3fdvu
- sqlserverkzzwhf4j3fdvu
- DemoDatabase
- storagekzzwhf4j3fdvu
- DemoSitePlan
- webSitekzzwhf4j3fdvu

9. If you make changes and want to redeploy your project, choose the existing resource group from the

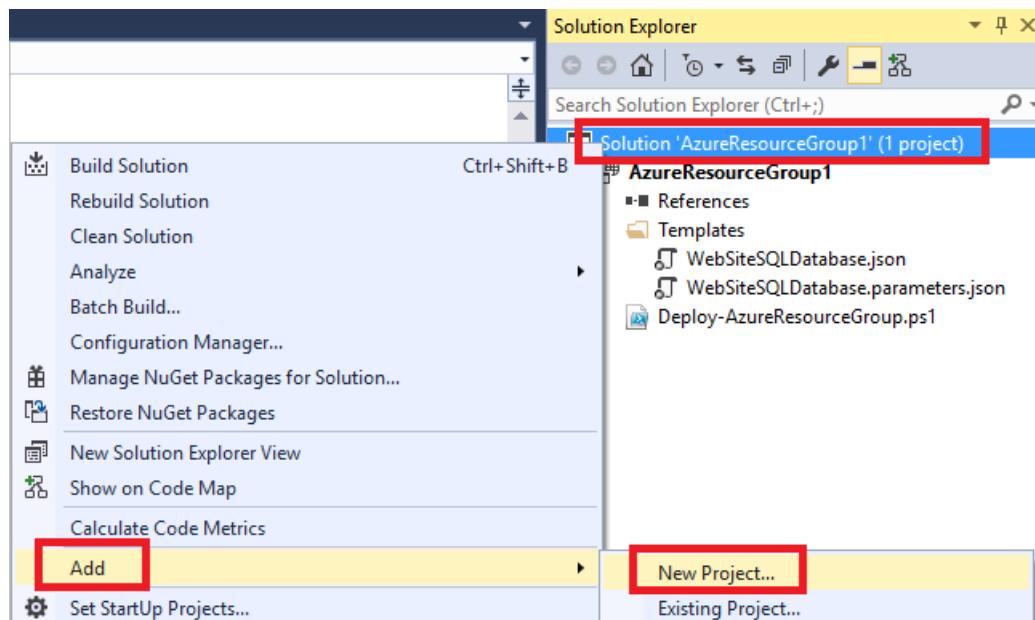
shortcut menu of Azure resource group project. On the shortcut menu, choose **Deploy**, and then choose the resource group you deployed.



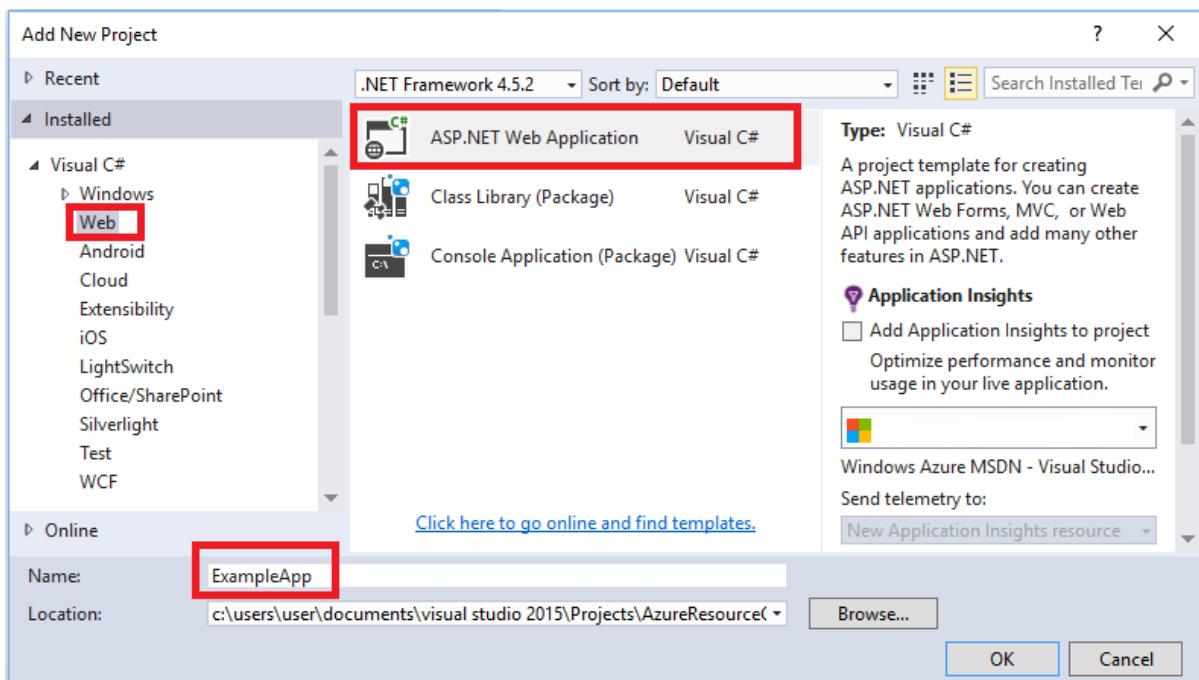
Deploy code with your infrastructure

At this point, you have deployed the infrastructure for your app, but there is no actual code deployed with the project. This topic shows how to deploy a web app and SQL Database tables during deployment. If you are deploying a Virtual Machine instead of a web app, you want to run some code on the machine as part of deployment. The process for deploying code for a web app or for setting up a Virtual Machine is almost the same.

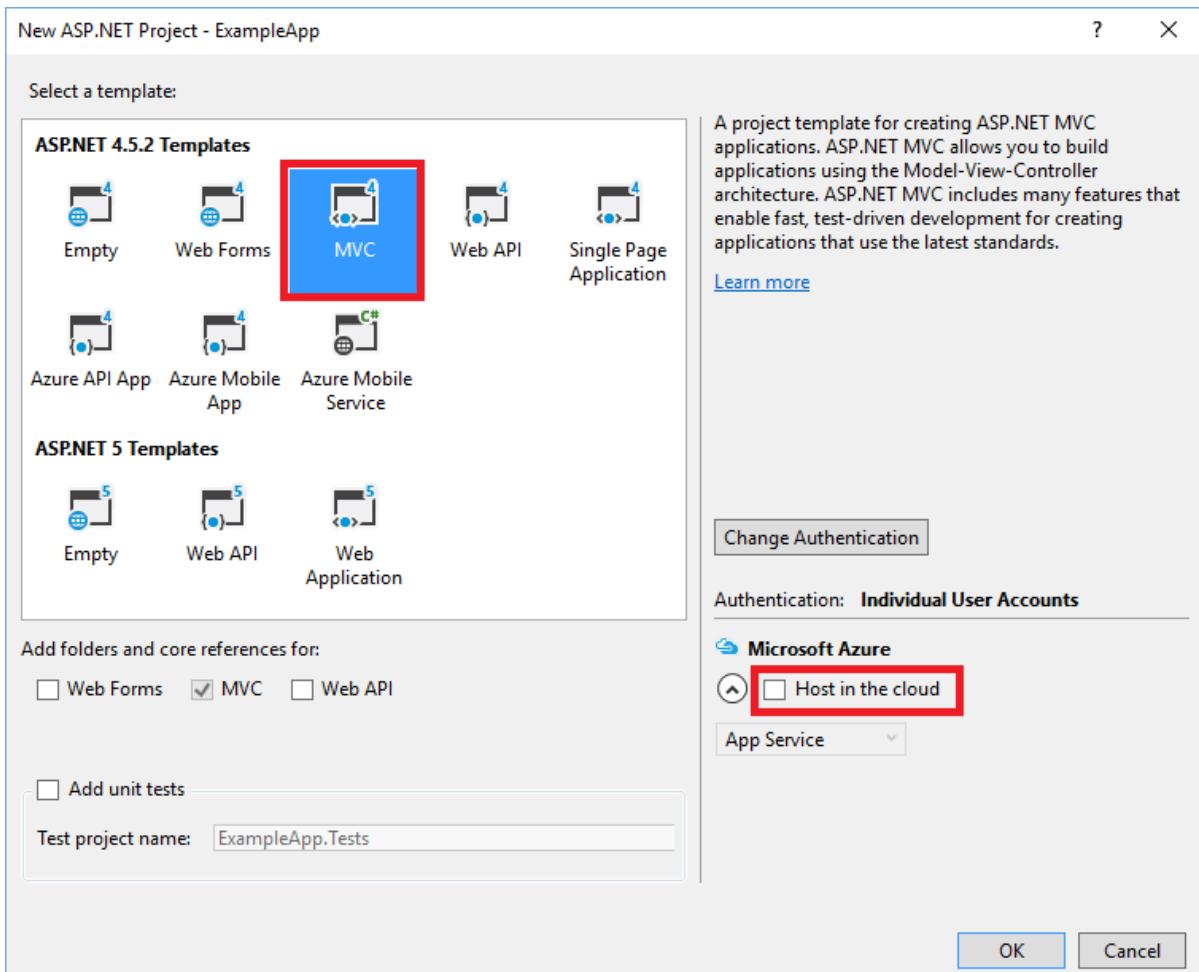
1. Add a project to your Visual Studio solution. Right-click the solution, and select **Add > New Project**.



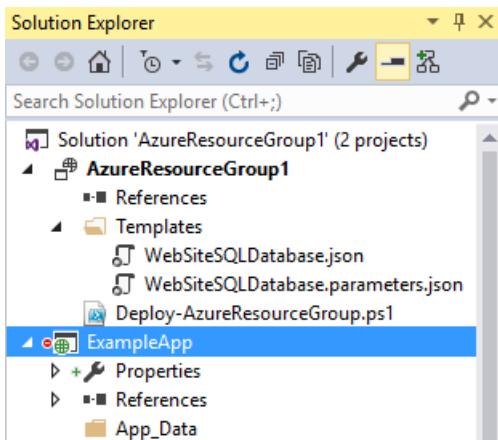
2. Add an **ASP.NET Web Application**.



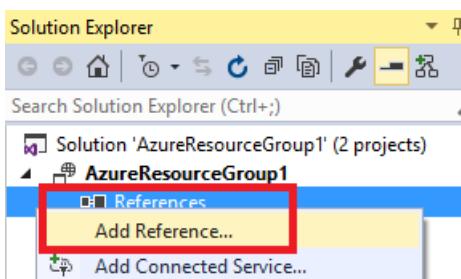
3. Select **MVC** and clear the field for **Host in the cloud** because the resource group project performs that task.



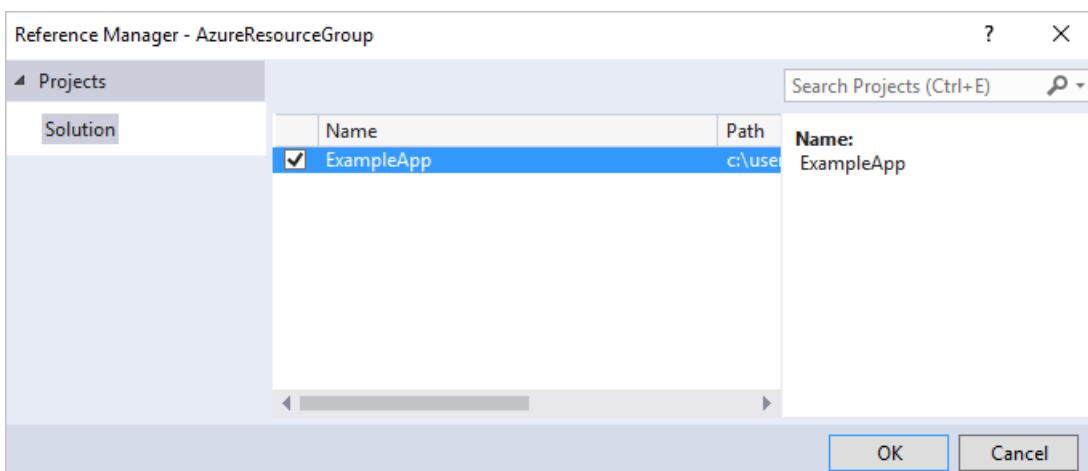
4. After Visual Studio creates your web app, you see both projects in the solution.



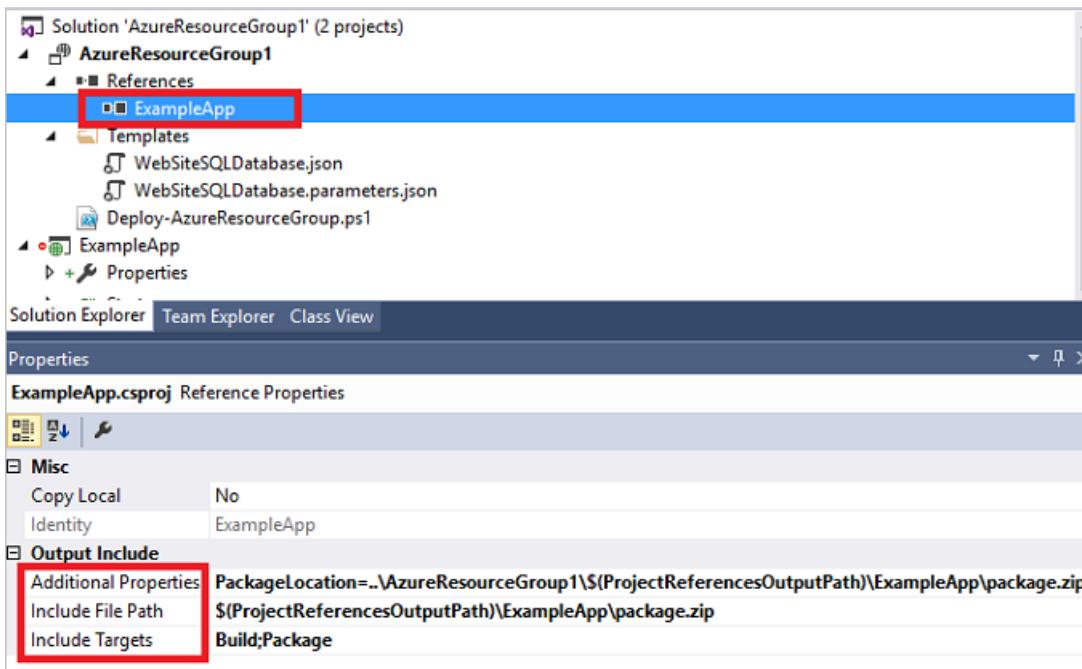
- Now, you need to make sure your resource group project is aware of the new project. Go back to your resource group project (AzureResourceGroup1). Right-click **References** and select **Add Reference**.



- Select the web app project that you created.



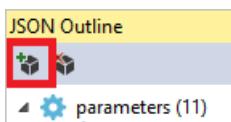
By adding a reference, you link the web app project to the resource group project, and automatically set three key properties. You see these properties in the **Properties** window for the reference.



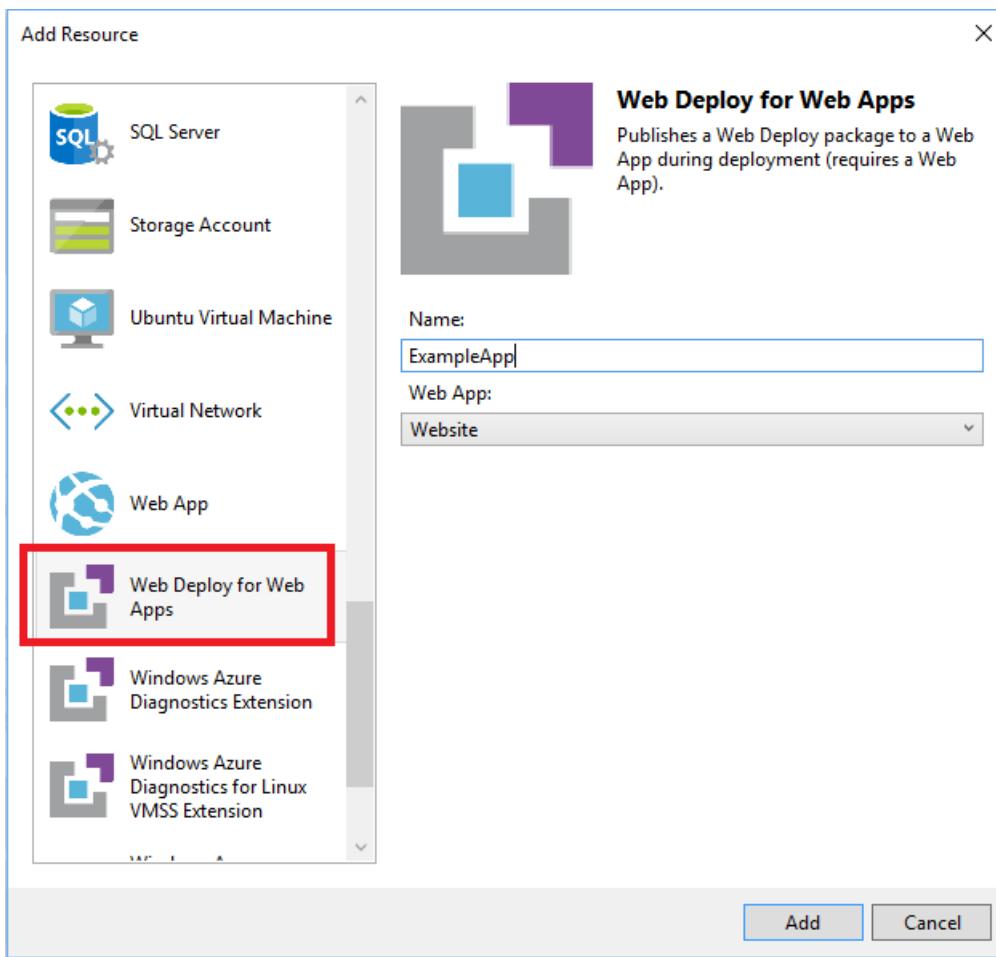
- The **Additional Properties** contains the web deployment package staging location that is pushed to the Azure Storage. Note the folder (ExampleApp) and file (package.zip). You need to know these values because you provide them as parameters when deploying the app.
- The **Include File Path** contains the path where the package is created. The **Include Targets** contains the command that deployment executes.
- The default value of **Build;Package** enables the deployment to build and create a web deployment package (package.zip).

You do not need a publish profile as the deployment gets the necessary information from the properties to create the package.

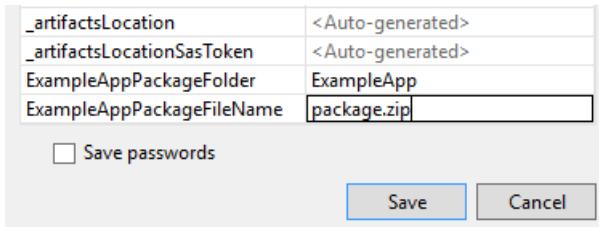
7. Add a resource to the template.



8. This time select **Web Deploy for Web Apps**.

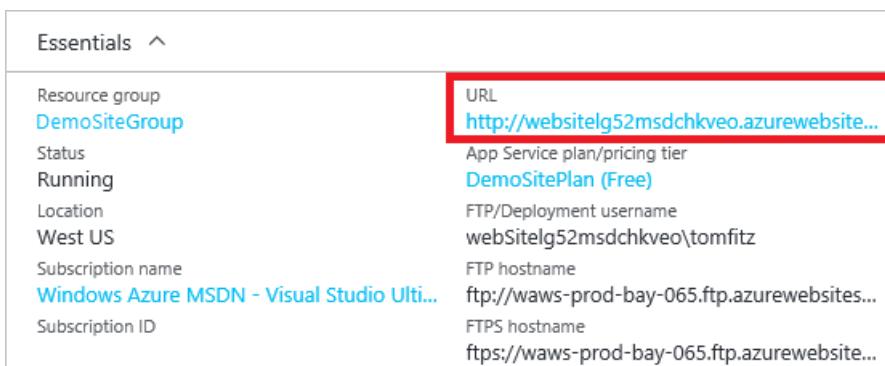


9. Redeploy your resource group project to the resource group. This time there are some new parameters. You do not need to provide values for **_artifactsLocation** or **_artifactsLocationSasToken** because Visual Studio automatically generates those values. However, you have to set the folder and file name to the path that contains the deployment package (shown as **ExampleAppPackageFolder** and **ExampleAppPackageName** in the following image). Provide the values you saw earlier in the reference properties (**ExampleApp** and **package.zip**).



For the **Artifact storage account**, select the one deployed with this resource group.

10. After the deployment has finished, select your web app in the portal. Select the URL to browse to the site.



11. Notice that you have successfully deployed the default ASP.NET app.

ASP.NET

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

[Learn more »](#)

Next steps

- To learn about managing your resources through the portal, see [Using the Azure portal to manage your Azure resources](#).
- To learn more about templates, see [Authoring Azure Resource Manager templates](#).

Deploy Azure Resources using C#

1/17/2017 • 9 min to read • [Edit on GitHub](#)

This article shows you how to create Azure resources using C#.

You first need to make sure you've finished these tasks:

- Install [Visual Studio](#)
- Verify the installation of [Windows Management Framework 3.0](#) or [Windows Management Framework 4.0](#)
- Get an [authentication token](#)

It takes about 30 minutes to do these steps.

Step 1: Create a Visual Studio project and install the libraries

NuGet packages are the easiest way to install the libraries that you need to finish this tutorial. To get the libraries that you need in Visual Studio, do these steps:

1. Click **File > New > Project**.
2. In **Templates > Visual C#**, select **Console Application**, enter the name and location of the project, and then click **OK**.
3. Right-click the project name in the Solution Explorer, and then click **Manage NuGet Packages**.
4. Type *Active Directory* in the search box, click **Install** for the Active Directory Authentication Library package, and then follow the instructions to install the package.
5. At the top of the page, select **Include Prerelease**. Type *Microsoft.Azure.Management.Compute* in the search box, click **Install** for the Compute .NET Libraries, and then follow the instructions to install the package.
6. Type *Microsoft.Azure.Management.Network* in the search box, click **Install** for the Network .NET Libraries, and then follow the instructions to install the package.
7. Type *Microsoft.Azure.Management.Storage* in the search box, click **Install** for the Storage .NET Libraries, and then follow the instructions to install the package.
8. Type *Microsoft.Azure.Management.ResourceManager* in the search box, click **Install** for the Resource Management Libraries.

Now you're ready to start using the libraries to create your application.

Step 2: Create the credentials that are used to authenticate requests

Now you format the application information that you previously created into credentials that are used to authenticate requests to Azure Resource Manager.

1. Open the Program.cs file for the project that you created, and then add these using statements to the top of the file:

```

using Microsoft.Azure;
using Microsoft.IdentityModel.Clients.ActiveDirectory;
using Microsoft.Azure.Management.ResourceManager;
using Microsoft.Azure.Management.ResourceManager.Models;
using Microsoft.Azure.Management.Storage;
using Microsoft.Azure.Management.Storage.Models;
using Microsoft.Azure.Management.Network;
using Microsoft.Azure.Management.Network.Models;
using Microsoft.Azure.Management.Compute;
using Microsoft.Azure.Management.Compute.Models;
using Microsoft.Rest;

```

- To create the token that is needed, add this method to the Program class:

```

private static async Task<AuthenticationResult> GetAccessTokenAsync()
{
    var cc = new ClientCredential("{client-id}", "{client-secret}");
    var context = new AuthenticationContext("https://login.windows.net/{tenant-id}");
    var token = await context.AcquireTokenAsync("https://management.azure.com/", cc);
    if (token == null)
    {
        throw new InvalidOperationException("Could not get the token");
    }
    return token;
}

```

Replace {client-id} with the identifier of the Azure Active Directory application, {client-secret} with the access key of the AD application, and {tenant-id} with the tenant identifier for your subscription. You can find the tenant id by running Get-AzureRmSubscription. You can find the access key by using the Azure portal.

- To call the method that you previously added, add this code to the Main method in the Program.cs file:

```

var token = GetAccessTokenAsync();
var credential = new TokenCredentials(token.Result.AccessToken);

```

- Save the Program.cs file.

Step 3: Register the resource providers and create the resources

Register the providers and create a resource group

All resources must be contained in a resource group. Before you can add resources to a group, your subscription must be registered with the resource providers.

- Add variables to the Main method of the Program class to specify the names that you want to use for the resources:

```

var groupName = "resource group name";
var subscriptionId = "subscription id";
var location = "location name";
var storageName = "storage account name";
var ipName = "public ip name";
var subnetName = "subnet name";
var vnetName = "virtual network name";
var nicName = "network interface name";
var avSetName = "availability set name";
var vmName = "virtual machine name";
var adminName = "administrator account name";
var adminPassword = "administrator account password";

```

Replace all the variable values with the names and identifier that you want to use. You can find the

subscription identifier by running Get-AzureRmSubscription.

2. To create the resource group and register the providers, add this method to the Program class:

```
public static async Task<ResourceGroup> CreateResourceGroupAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location)
{
    var resourceManagementClient = new ResourceManagementClient(credential)
        { SubscriptionId = subscriptionId };

    Console.WriteLine("Registering the providers...");
    var rpResult = resourceManagementClient.Providers.Register("Microsoft.Storage");
    Console.WriteLine(rpResult.RegistrationState);
    rpResult = resourceManagementClient.Providers.Register("Microsoft.Network");
    Console.WriteLine(rpResult.RegistrationState);
    rpResult = resourceManagementClient.Providers.Register("Microsoft.Compute");
    Console.WriteLine(rpResult.RegistrationState);

    Console.WriteLine("Creating the resource group...");
    var resourceGroup = new ResourceGroup { Location = location };
    return await resourceManagementClient.ResourceGroups.CreateOrUpdateAsync(groupName, resourceGroup);
}
```

3. To call the method that you previously added, add this code to the Main method:

```
var rgResult = CreateResourceGroupAsync(
    credential,
    groupName,
    subscriptionId,
    location);
Console.WriteLine(rgResult.Result.Properties.ProvisioningState);
Console.ReadLine();
```

Create a storage account

A [storage account](#) is needed to store the virtual hard disk file that is created for the virtual machine.

1. To create the storage account, add this method to the Program class:

```
public static async Task<StorageAccount> CreateStorageAccountAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location,
    string storageName)
{
    Console.WriteLine("Creating the storage account...");
    var storageManagementClient = new StorageManagementClient(credential)
        { SubscriptionId = subscriptionId };
    return await storageManagementClient.StorageAccounts.CreateAsync(
        groupName,
        storageName,
        new StorageAccountCreateParameters()
        {
            Sku = new Microsoft.Azure.Management.Storage.Models.Sku()
                { Name = SkuName.StandardLRS },
            Kind = Kind.Storage,
            Location = location
        }
    );
}
```

2. To call the method that you previously added, add this code to the Main method of the Program class:

```
var stResult = CreateStorageAccountAsync(
    credential,
    groupName,
    subscriptionId,
    location,
    storageName);
Console.WriteLine(stResult.Result.ProvisioningState);
Console.ReadLine();
```

Create the public IP address

A public IP address is needed to communicate with the virtual machine.

1. To create the public IP address of the virtual machine, add this method to the Program class:

```
public static async Task<PublicIPAddress> CreatePublicIPAddressAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location,
    string ipName)
{
    Console.WriteLine("Creating the public ip...");
    var networkManagementClient = new NetworkManagementClient(credential)
        { SubscriptionId = subscriptionId };
    return await networkManagementClient.PublicIPAddresses.CreateOrUpdateAsync(
        groupName,
        ipName,
        new PublicIPAddress
        {
            Location = location,
            PublicIPAllocationMethod = "Dynamic"
        });
}
```

2. To call the method that you previously added, add this code to the Main method of the Program class:

```
var ipResult = CreatePublicIPAddressAsync(
    credential,
    groupName,
    subscriptionId,
    location,
    ipName);
Console.WriteLine(ipResult.Result.ProvisioningState);
Console.ReadLine();
```

Create the virtual network

A virtual machine that's created with the Resource Manager deployment model must be in a virtual network.

1. To create a subnet and a virtual network, add this method to the Program class:

```

public static async Task<VirtualNetwork> CreateVirtualNetworkAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location,
    string vnetName,
    string subnetName)
{
    Console.WriteLine("Creating the virtual network...");
    var networkManagementClient = new NetworkManagementClient(credential)
        { SubscriptionId = subscriptionId };

    var subnet = new Subnet
    {
        Name = subnetName,
        AddressPrefix = "10.0.0.0/24"
    };

    var address = new AddressSpace {
        AddressPrefixes = new List<string> { "10.0.0.0/16" }
    };

    return await networkManagementClient.VirtualNetworks.CreateOrUpdateAsync(
        groupName,
        vnetName,
        new VirtualNetwork
        {
            Location = location,
            AddressSpace = address,
            Subnets = new List<Subnet> { subnet }
        }
    );
}

```

2. To call the method that you previously added, add this code to the Main method of the Program class:

```

var vnResult = CreateVirtualNetworkAsync(
    credential,
    groupName,
    subscriptionId,
    location,
    vnetName,
    subnetName);
Console.WriteLine(vnResult.Result.ProvisioningState);
Console.ReadLine();

```

Create the network interface

A virtual machine needs a network interface to communicate on the virtual network.

1. To create a network interface, add this method to the Program class:

```

public static async Task<NetworkInterface> CreateNetworkInterfaceAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location,
    string subnetName,
    string vnetName,
    string ipName,
    string nicName)
{
    Console.WriteLine("Creating the network interface...");
    var networkManagementClient = new NetworkManagementClient(credential)
        { SubscriptionId = subscriptionId };
    var subnetResponse = await networkManagementClient.Subnets.GetAsync(
        groupName,
        vnetName,
        subnetName
    );
    var pubipResponse = await networkManagementClient.PublicIPAddresses.GetAsync(groupName, ipName);

    return await networkManagementClient.NetworkInterfaces.CreateOrUpdateAsync(
        groupName,
        nicName,
        new NetworkInterface
        {
            Location = location,
            IpConfigurations = new List<NetworkInterfaceIPConfiguration>
            {
                new NetworkInterfaceIPConfiguration
                {
                    Name = nicName,
                    PublicIPAddress = pubipResponse,
                    Subnet = subnetResponse
                }
            }
        }
    );
}

```

2. To call the method that you previously added, add this code to the Main method of the Program class:

```

var ncResult = CreateNetworkInterfaceAsync(
    credential,
    groupName,
    subscriptionId,
    location,
    subnetName,
    vnetName,
    ipName,
    nicName);
Console.WriteLine(ncResult.Result.ProvisioningState);
Console.ReadLine();

```

Create an availability set

Availability sets make it easier for you to manage the maintenance of the virtual machines used by your application.

1. To create the availability set, add this method to the Program class:

```

public static async Task<AvailabilitySet> CreateAvailabilitySetAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location,
    string avsetName)
{
    Console.WriteLine("Creating the availability set...");
    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    return await computeManagementClient.AvailabilitySets.CreateOrUpdateAsync(
        groupName,
        avsetName,
        new AvailabilitySet()
        {
            Location = location
        }
    );
}

```

- To call the method that you previously added, add this code to the Main method of the Program class:

```

var avResult = CreateAvailabilitySetAsync(
    credential,
    groupName,
    subscriptionId,
    location,
    avSetName);
Console.ReadLine();

```

Create a virtual machine

Now that you created all the supporting resources, you can create a virtual machine.

- To create the virtual machine, add this method to the Program class:

```

public static async Task<VirtualMachine> CreateVirtualMachineAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId,
    string location,
    string nicName,
    string avsetName,
    string storageName,
    string adminName,
    string adminPassword,
    string vmName)
{
    var networkManagementClient = new NetworkManagementClient(credential)
        { SubscriptionId = subscriptionId };
    var nic = networkManagementClient.NetworkInterfaces.Get(groupName, nicName);

    var computeManagementClient = new ComputeManagementClient(credential);
    computeManagementClient.SubscriptionId = subscriptionId;
    var avSet = computeManagementClient.AvailabilitySets.Get(groupName, avsetName);

    Console.WriteLine("Creating the virtual machine...");
    return await computeManagementClient.VirtualMachines.CreateOrUpdateAsync(
        groupName,
        vmName,
        new VirtualMachine()
        {
            Location = location,
            AvailabilitySet = new Microsoft.Azure.Management.Compute.Models.SubResource()
            {
                Id = avSet.Id
            }
        }
    );
}

```

```

},
HardwareProfile = new HardwareProfile
{
    VmSize = "Standard_A0"
},
OsProfile = new OSProfile
{
    AdminUsername = adminName,
    AdminPassword = adminPassword,
    ComputerName = vmName,
    WindowsConfiguration = new WindowsConfiguration
    {
        ProvisionVMAgent = true
    }
},
NetworkProfile = new NetworkProfile
{
    NetworkInterfaces = new List<NetworkInterfaceReference>
    {
        new NetworkInterfaceReference { Id = nic.Id }
    }
},
StorageProfile = new StorageProfile
{
    ImageReference = new ImageReference
    {
        Publisher = "MicrosoftWindowsServer",
        Offer = "WindowsServer",
        Sku = "2012-R2-Datacenter",
        Version = "latest"
    },
    OsDisk = new OSDisk
    {
        Name = "mytestod1",
        CreateOption = DiskCreateOptionTypes.FromImage,
        Vhd = new VirtualHardDisk
        {
            Uri = "http://" + storageName + ".blob.core.windows.net/vhds/mytestod1.vhd"
        }
    }
}
);
}
}

```

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 method that you previously added, add this code to the Main method:

```

var vmResult = CreateVirtualMachineAsync(
    credential,
    groupName,
    subscriptionId,
    location,
    nicName,
    avSetName,
    storageName,
    adminName,
    adminPassword,
    vmName);
Console.WriteLine(vmResult.Result.ProvisioningState);
Console.ReadLine();

```

Step 4: Delete the resources

Because you are charged for resources used in Azure, it is 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, add this method to the Program class:

```

public static async void DeleteResourceGroupAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId)
{
    Console.WriteLine("Deleting resource group...");
    var resourceManagementClient = new ResourceManagementClient(credential)
        { SubscriptionId = subscriptionId };
    await resourceManagementClient.ResourceGroups.DeleteAsync(groupName);
}

```

2. To call the method that you previously added, add this code to the Main method:

```

DeleteResourceGroupAsync(
    credential,
    groupName,
    subscriptionId);
Console.ReadLine();

```

Step 5: Run the console application

1. To run the console application, click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.
2. Press **Enter** after each status code is returned to create each resource. After the virtual machine is created, do the next step before pressing Enter to delete all the resources.

It should take about five minutes for this console application to run completely from start to finish. 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 before you delete them.

3. To see the status of the resources, browse to the Audit Logs in the Azure portal:

OPERATION	LEVEL	STATUS	RESOURCE	TIME
Microsoft.Compute/virtualMachines/write	Information	Succeeded	...virtualMachines/mytestvm1	Just now
Microsoft.Compute/availabilitySets/write	Information	Succeeded	...availabilitySets/mytestav1	1 min ago
Microsoft.Storage/storageAccounts/write	Information	Succeeded	...storageAccounts/mytests1	Just now
microsoft.network/networkInterfaces/wri...	Information	Succeeded	...networkInterfaces/mytestnic1	1 min ago
microsoft.network/publicIPAddresses/wr...	Information	Succeeded	...publicIPAddresses/mytestip1	1 min ago
microsoft.network/virtualnetworks/write	Information	Succeeded	...virtualnetworks/mytestvn1	1 min ago
Update resource group	Information	Succeeded	...mytestrg1	Just now

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 how to manage the virtual machine that you created by reviewing [Manage virtual machines using Azure Resource Manager and PowerShell](#).

Deploy an Azure Virtual Machine using C# and a Resource Manager template

1/17/2017 • 7 min to read • [Edit on GitHub](#)

By using resource groups and templates, you're able to manage all the resources together that support your application. This article shows you how to use Visual Studio and C# to set up authentication, create a template, and then deploy Azure resources using the template that you created.

You first need to make sure you've done these setup steps:

- Install [Visual Studio](#)
- Verify the installation of [Windows Management Framework 3.0](#) or [Windows Management Framework 4.0](#)
- Get an [authentication token](#)
- Create a resource group using [Azure PowerShell](#), [Azure CLI](#), or [Azure portal](#).

It takes about 30 minutes to do these steps.

Step 1: Create the Visual Studio project, the template file, and the parameters file

Create the template file

An Azure Resource Manager template makes it possible for you to deploy and manage Azure resources together. The template is a JSON description of the resources and associated deployment parameters.

In Visual Studio, do these steps:

1. Click **File > New > Project**.
2. In **Templates > Visual C#**, select **Console Application**, enter the name and location of the project, and then click **OK**.
3. Right-click the project name in Solution Explorer, click **Add > New Item**.
4. Click **Web**, select **JSON File**, enter **VirtualMachineTemplate.json** for the name, and then click **Add**.
5. In the opening and closing brackets of the **VirtualMachineTemplate.json** file, add the required schema element and the required **contentVersion** element:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json",  
  "contentVersion": "1.0.0.0",  
}
```

6. [Parameters](#) are not always required, but they provide a way to input values when the template is deployed. Add the **parameters** element and its child elements after the **contentVersion** element:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json",  
  "contentVersion": "1.0.0.0",  
  "parameters": {  
    "adminUserName": { "type": "string" },  
    "adminPassword": { "type": "securestring" }  
  },  
}
```

7. [Variables](#) can be used in a template to specify values that may change frequently or values that need to be created from a combination of parameter values. Add the variables element after the parameters section:

```
{
  "$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','myvn1')]",
    "subnetRef": "[concat(variables('vnetID'), '/subnets/mysn1')]"
  },
}
```

8. [Resources](#) such as the virtual machine, the virtual network, and the storage account are defined next in the template. Add the resources section after the variables section:

```
{
  "$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','myvn1')]",
    "subnetRef": "[concat(variables('vnetID'), '/subnets/mysn1')]"
  },
  "resources": [
    {
      "type": "Microsoft.Storage/storageAccounts",
      "name": "mystorage1",
      "apiVersion": "2015-06-15",
      "location": "[resourceGroup().location]",
      "properties": { "accountType": "Standard_LRS" }
    },
    {
      "apiVersion": "2016-03-30",
      "type": "Microsoft.Network/publicIPAddresses",
      "name": "myip1",
      "location": "[resourceGroup().location]",
      "properties": {
        "publicIPAllocationMethod": "Dynamic",
        "dnsSettings": { "domainNameLabel": "mydns1" }
      }
    },
    {
      "apiVersion": "2016-03-30",
      "type": "Microsoft.Network/virtualNetworks",
      "name": "myvn1",
      "location": "[resourceGroup().location]",
      "properties": {
        "addressSpace": { "addressPrefixes": [ "10.0.0.0/16" ] },
        "subnets": [ {
          "name": "mysn1",
          "properties": { "addressPrefix": "10.0.0.0/24" }
        } ]
      }
    },
    {
      "apiVersion": "2016-03-30",
      "type": "Microsoft.Network/networkInterfaces",
      "name": "mync1",
      "location": "[resourceGroup().location]",
```

```

"dependsOn": [
    "Microsoft.Network/publicIPAddresses/myip1",
    "Microsoft.Network/virtualNetworks/myvn1"
],
"properties": {
    "ipConfigurations": [ {
        "name": "ipconfig1",
        "properties": {
            "privateIPAllocationMethod": "Dynamic",
            "publicIPAddress": {
                "id": "[resourceId('Microsoft.Network/publicIPAddresses', 'myip1')]"
            },
            "subnet": { "id": "[variables('subnetRef')]" }
        }
    } ]
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "myvm1",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "Microsoft.Network/networkInterfaces/mync1",
        "Microsoft.Storage/storageAccounts/mystorage1"
    ],
    "properties": {
        "hardwareProfile": { "vmSize": "Standard_A1" },
        "osProfile": {
            "computerName": "myvm1",
            "adminUsername": "[parameters('adminUsername')]",
            "adminPassword": "[parameters('adminPassword')]"
        },
        "storageProfile": {
            "imageReference": {
                "publisher": "MicrosoftWindowsServer",
                "offer": "WindowsServer",
                "sku": "2012-R2-Datacenter",
                "version" : "latest"
            },
            "osDisk": {
                "name": "myosdisk1",
                "vhd": {
                    "uri": "https://mystorage1.blob.core.windows.netvhds/myosdisk1.vhd"
                },
                "caching": "ReadWrite",
                "createOption": "FromImage"
            }
        },
        "networkProfile": {
            "networkInterfaces" : [ {
                "id": "[resourceId('Microsoft.Network/networkInterfaces', 'mync1')]"
            } ]
        }
    }
}
}

```

9. Save the template file that you created.

Create the parameters file

To specify values for the resource parameters that were defined in the template, you create a parameters file that contains the values that are used when the template is deployed. In Visual Studio, do these steps:

1. Right-click the project name in Solution Explorer, click **Add > New Item**.
2. Click Web, select JSON File, enter *Parameters.json* for the Name, and then click **Add**.

3. Open the parameters.json file and then add this JSON content:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json",  
  "contentVersion": "1.0.0.0",  
  "parameters": {  
    "adminUserName": { "value": "mytestacct1" },  
    "adminPassword": { "value": "mytestpass1" }  
  }  
}
```

NOTE

This article 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](#).

4. Save the parameters file that you created.

Step 2: Install the libraries

NuGet packages are the easiest way to install the libraries that you need to finish this tutorial. You need the Azure Resource Management Library and the Azure Active Directory Authentication Library to create the resources. To get these libraries in Visual Studio, do these steps:

1. Right-click the project name in the Solution Explorer, click **Manage NuGet Packages**, and then click Browse.
2. Type *Active Directory* in the search box, click **Install** for the Active Directory Authentication Library package, and then follow the instructions to install the package.
3. At the top of the page, select **Include Prerelease**. Type *Microsoft.Azure.Management.ResourceManager* in the search box, click **Install** for the Microsoft Azure Resource Management Libraries, and then follow the instructions to install the package.

Now you're ready to start using the libraries to create your application.

Step 3: Create the credentials that are used to authenticate requests

The Azure Active Directory application is created and the authentication library is installed. Now you format the application information into credentials that are used to authenticate requests to the Azure Resource Manager.

1. Open the Program.cs file for the project that you created, and then add these using statements to the top of the file:

```
using Microsoft.Azure;  
using Microsoft.IdentityModel.Clients.ActiveDirectory;  
using Microsoft.Azure.Management.ResourceManager;  
using Microsoft.Azure.Management.ResourceManager.Models;  
using Microsoft.Rest;  
using System.IO;
```

2. Add this method to the Program class to get the token that's needed to create the credentials:

```

private static async Task<AuthenticationResult> GetAccessTokenAsync()
{
    var cc = new ClientCredential("{client-id}", "{client-secret}");
    var context = new AuthenticationContext("https://login.windows.net/{tenant-id}");
    var token = await context.AcquireTokenAsync("https://management.azure.com/", cc);
    if (token == null)
    {
        throw new InvalidOperationException("Could not get the token.");
    }
    return token;
}

```

Replace {client-id} with the identifier of the Azure Active Directory application, {client-secret} with the access key of the AD application, and {tenant-id} with the tenant identifier for your subscription. You can find the tenant id by running `Get-AzureRmSubscription`. You can find the access key by using the Azure portal.

- To create the credentials, add this code to the Main method in the Program.cs file:

```

var token = GetAccessTokenAsync();
var credential = new TokenCredentials(token.Result.AccessToken);

```

- Save the Program.cs file.

Step 4: Deploy the template

In this step, you use the resource group that you previously created, but you could also create a resource group using the [ResourceGroup](#) and the [ResourceManagementClient](#) classes.

- Add variables to the Main method of the Program class to specify the names of the resources that you previously created, the deployment name, and your subscription identifier:

```

var groupName = "resource group name";
var subscriptionId = "subscription id";
var deploymentName = "deployment name";

```

Replace the value of groupName with the name of your resource group. Replace the value of deploymentName with the name that you want to use for the deployment. You can find the subscription identifier by running `Get-AzureRmSubscription`.

- Add this method to the Program class to deploy the resources to the resource group by using the template that you defined:

```

public static async Task<DeploymentExtended> CreateTemplateDeploymentAsync(
    TokenCredentials credential,
    string groupName,
    string deploymentName,
    string subscriptionId)
{
    Console.WriteLine("Creating the template deployment...");
    var deployment = new Deployment();
    deployment.Properties = new DeploymentProperties
    {
        Mode = DeploymentMode.Incremental,
        Template = File.ReadAllText("../..\\VirtualMachineTemplate.json"),
        Parameters = File.ReadAllText("../..\\Parameters.json")
    };
    var resourceManagementClient = new ResourceManagementClient(credential)
        { SubscriptionId = subscriptionId };
    return await resourceManagementClient.Deployments.CreateOrUpdateAsync(
        groupName,
        deploymentName,
        deployment);
}

```

If you wanted to deploy the template from a storage account, you can replace the `Template` property with the `TemplateLink` property.

- To call the method that you just added, add this code to the Main method:

```

var dpResult = CreateTemplateDeploymentAsync(
    credential,
    groupName,
    deploymentName,
    subscriptionId);
Console.WriteLine(dpResult.Result.Properties.ProvisioningState);
Console.ReadLine();

```

Step 5: Delete the resources

Because you are charged for resources used in Azure, it is always a 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 method to the Program class:

```

public static async void DeleteResourceGroupAsync(
    TokenCredentials credential,
    string groupName,
    string subscriptionId)
{
    Console.WriteLine("Deleting resource group...");
    var resourceManagementClient = new ResourceManagementClient(credential)
        { SubscriptionId = subscriptionId };
    await resourceManagementClient.ResourceGroups.DeleteAsync(groupName);
}

```

- To call the method that you just added, add this code to the Main method:

```

DeleteResourceGroupAsync(
    credential,
    groupName,
    subscriptionId);
Console.ReadLine();

```

Step 6: Run the console application

1. To run the console application, click **Start** in Visual Studio, and then sign in to Azure AD using the same credentials that you use with your subscription.
2. Press **Enter** after the Accepted status appears.

It should take about five minutes for this console application to run completely from start to finish. 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 before you delete them.

3. To see the status of the resources, browse to the Audit Logs in the Azure portal:

OPERATION	LEVEL	STATUS	RESOURCE	TIME
Microsoft.Compute/virtualMachines/write	Info	Succeeded	...virtualMachines/mytestvm1	Just now
Microsoft.Compute/availabilitySets/write	Info	Succeeded	...availabilitySets/mytestav1	1 min ago
Microsoft.Storage/storageAccounts/write	Info	Succeeded	...storageAccounts/mytests1	Just now
microsoft.network/networkInterfaces/wri...	Info	Succeeded	...networkInterfaces/mytestnic1	1 min ago
microsoft.network/publicIPAddresses/wr...	Info	Succeeded	...publicIPAddresses/mytestip1	1 min ago
microsoft.network/virtualnetworks/write	Info	Succeeded	...virtualnetworks/mytestvn1	1 min ago
Update resource group	Info	Succeeded	...mytestrg1	Just now

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 manage the virtual machine that you created by reviewing [Manage virtual machines using Azure Resource Manager and PowerShell](#).

Deploy popular application frameworks using Azure Resource Manager templates

1/17/2017 • 2 min to read • [Edit on GitHub](#)

Workloads usually require many resources to function according to design. Azure Resource Manager templates make it possible for you to not only define how applications are configured, but also how the resources are deployed to support configured applications. This article introduces you to the most popular templates in the gallery and gives you information for using the Azure portal, Azure CLI, or PowerShell to deploy them. You can also [see the Linux version of this topic](#).

Applications

From this table you can find more information about the parameters that are used in the template, you can inspect the template before you deploy it, or you can deploy the template directly from the Azure portal.

APPLICATION	LEARN MORE	VIEW THE TEMPLATE	DEPLOY IT NOW
Active Directory	Gallery	GitHub	 Deploy to Azure
Apache	Gallery	GitHub	 Deploy to Azure
Couchbase	Gallery	GitHub	 Deploy to Azure
DataStax	Gallery	GitHub	 Deploy to Azure
Django	Gallery	GitHub	 Deploy to Azure
Docker	Gallery	GitHub	 Deploy to Azure
Elasticsearch	Gallery	GitHub	 Deploy to Azure
Jenkins	Gallery	GitHub	 Deploy to Azure
Kafka	Gallery	GitHub	 Deploy to Azure
LAMP	Gallery	GitHub	 Deploy to Azure
MongoDB	Gallery	GitHub	 Deploy to Azure

APPLICATION	LEARN MORE	VIEW THE TEMPLATE	DEPLOY IT NOW
Redis	Gallery	GitHub	 Deploy to Azure
SharePoint	Gallery	GitHub	 Deploy to Azure
Spark	Gallery	GitHub	 Deploy to Azure
Tomcat	Gallery	GitHub	 Deploy to Azure
WordPress	Gallery	GitHub	 Deploy to Azure
ZooKeeper	Gallery	GitHub	 Deploy to Azure

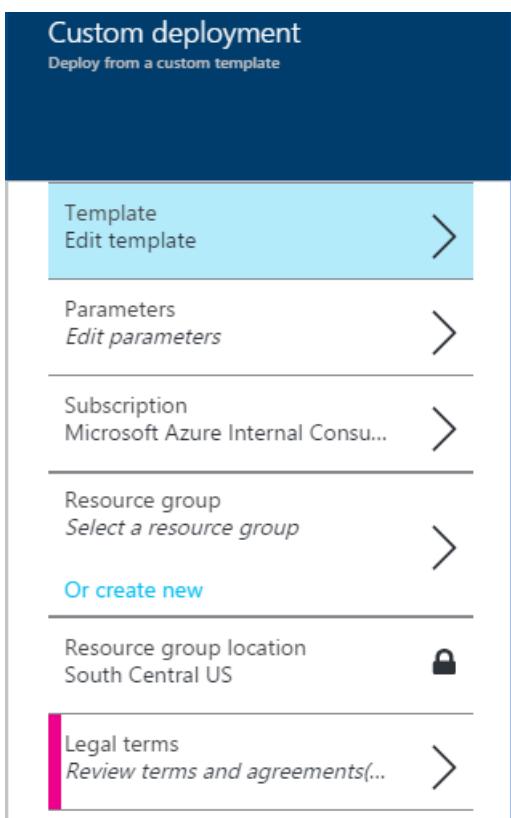
In addition to these templates, you can search through the [gallery templates](#).

Azure portal

Deploying a template by using the Azure portal is easy to do by just sending a URL to it. You need the name of the template file to deploy it. You can find the name by looking at the pages in the template gallery or by looking in the Github repository. Change {template name} in this URL to the name of the template that you want to deploy and then enter it into your browser:

```
https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fraw.githubusercontent.com%2FAzure%2Fazure-quickstart-templates%2Fmaster%2F{template name}%2Fazuredeploy.json
```

You should see the custom deployment blade:



1. For the **Template** pane, click **Save**.
2. Click **Parameters**. On the **Parameters** pane, enter new values, select from allowed values, or accept default values, and then click **OK**.
3. If needed, click **Subscription** and select the correct Azure subscription.
4. Click **Resource group** and select an existing resource group. Alternately, click **Or create new** to create a new one for this deployment.
5. If needed, click **Location** and select the correct Azure location.
6. If needed, click **Legal terms** to review the terms and agreement for using the template.
7. Click **Create**.

Depending on the template, it can take some time for Azure to deploy the resources.

Azure PowerShell

Run these commands to create the resource group and the deployment after you replace the text in brackets with the resource group name, location, deployment name, and template name:

```
New-AzureRmResourceGroup -Name {resource-group-name} -Location {location}
New-AzureRmResourceGroupDeployment -Name {deployment-name} -ResourceGroupName {resource-group-name} -TemplateUri
"https://raw.githubusercontent.com/azure/azure-quickstart-templates/master/{template-name}/azuredploy.json"
```

When you run the **New-AzureRmResourceGroupDeployment** command, you are prompted to enter values for the parameters in the template. Depending on the template, it can take some time for Azure to deploy the resources.

Azure CLI

Install [Azure CLI](#), log in, and make sure you enable Resource Manager commands. For information about how to do this, see [Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager](#).

Run these commands to create the resource group and the deployment after you replace the text in brackets with the resource group name, location, deployment name, and template name:

```
azure group create {resource-group-name} {location}
azure group deployment create --template-uri https://raw.githubusercontent.com/azure/azure-quickstart-
templates/master/{template-name}/azuredploy.json {resource-group-name} {deployment-name}
```

When you run the **azure group deployment create** command, you are prompted to enter values for the parameters in the template. Depending on the template, it can take some time for Azure to deploy the resources.

Next steps

Discover all the templates at your disposal on [GitHub](#).

Learn more about [Azure Resource Manager](#).

Prepare a Windows VHD or VHDX to upload to Azure

1/17/2017 • 9 min to read • [Edit on GitHub](#)

To upload a Windows VM from on-premises to Azure, you must prepare the virtual hard disk (VHD or VHDX). Azure only supports generation 1 virtual machines 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 virtual machine from VHDX to the VHD file format and from dynamically expanding to a fixed sized disk. But you can't change a virtual machine's generation. For more information, see [Should I create a generation 1 or 2 virtual machine in Hyper-V?](#)

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 it, click **Action** > **Edit Disk**.
2. On the **Locate Virtual Hard Disk** screen, browse to, and select your virtual disk.
3. On the **Choose Action** screen, select **Convert** and **Next**.
4. If you need to convert from VHDX, select **VHD** and click **Next**
5. If you need to convert from dynamically expanding disk, select **Fixed size** and click **Next**
6. Browse to and select a path to save the new VHD file.
7. Click **Finish** to close.

Convert disk using PowerShell

You can convert a virtual disk by using the [Convert-VHD](#) cmdlet in Windows PowerShell. Select **Run as administrator** when you start PowerShell. The following example shows you how to convert from a 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
```

Replace the values for **-Path** with the path to the virtual hard disk that you want to convert and **-DestinationPath** with the new path and name for 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 Virtual Machine Converter](#). For more information, see the blog [How to Convert a VMware VMDK to Hyper-V VHD](#).

Set Windows configurations for Azure

On the virtual machine you plan to upload to Azure, run all the following commands from the command prompt window with [administrative privileges](#).

1. Remove any static persistent route on the routing table:

- To view the route table, run `route print` from the command prompt window.
 - 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
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**:

```
REG ADD HKLM\SYSTEM\CurrentControlSet\Control\TimeZoneInformation /v RealTimeIsUniversal /t REG_DWORD /d 1
```

```
sc config w32time start= auto
```

Set services startup to Windows default values

Make sure that each of the following Windows services is set to the **Windows default values**. To reset the startup settings, run the following commands:

```
sc config bfe start= auto  
  
sc config dcomlaunch start= auto  
  
sc config dhcp start= auto  
  
sc config dnscache start= auto  
  
sc config IKEEXT start= auto  
  
sc config iphlpsvc start= auto  
  
sc config PolicyAgent start= demand  
  
sc config LSM start= auto  
  
sc config netlogon start= demand  
  
sc config netman start= demand  
  
sc config NcaSvc start= demand  
  
sc config netprofm start= demand  
  
sc config NlaSvc start= auto  
  
sc config nsi start= auto  
  
sc config RpcSs start= auto  
  
sc config RpcEptMapper start= auto  
  
sc config termService start= demand  
  
sc config MpsSvc start= auto  
  
sc config WinHttpAutoProxySvc start= demand  
  
sc config LanmanWorkstation start= auto  
  
sc config RemoteRegistry start= auto
```

Update Remote Desktop registry settings

1. If there are any self-signed certificates tied to the Remote Desktop Protocol (RDP) listener, remove them:

```
REG DELETE "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp\SSLCertificateSHA1Hash"
```

For more information about configuring certificates for RDP listener, see [Listener Certificate Configurations in Windows Server](#)

2. Configure the **KeepAlive** values for RDP service:

```
REG ADD "HKLM\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services" /v KeepAliveEnable /t REG_DWORD /d 1 /f
```

```
REG ADD "HKLM\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services" /v KeepAliveInterval /t REG_DWORD /d 1 /f
```

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\Winstations\RDP-Tcp" /v KeepAliveTimeout  
/t REG_DWORD /d 1 /f
```

3. Configure the authentication mode for the RDP service:

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v  
UserAuthentication /t REG_DWORD /d 1 /f
```

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v SecurityLayer /t  
REG_DWORD /d 1 /f
```

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v  
fAllowSecProtocolNegotiation /t REG_DWORD /d 1 /f
```

4. Enable RDP service by adding the following subkeys to the registry:

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0  
/f
```

Configure Windows Firewall rules

1. Run the following command in PowerShell to allow WinRM through the three firewall profiles (Domain, Private, and Public) and enable PowerShell Remote service:

```
Enable-PSRemoting -force
```

2. Run the following commands in the command prompt window to make sure that the following guest operating system firewall rules are in place:

- Inbound

```
netsh advfirewall firewall set rule dir=in name="File and Printer Sharing (Echo Request - ICMPv4-In)" new  
enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (LLMNR-UDP-In)" new enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (NB-Datagram-In)" new enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (NB-Name-In)" new enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (Pub-WSD-In)" new enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (SSDP-In)" new enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (UPnP-In)" new enable=yes

netsh advfirewall firewall set rule dir=in name="Network Discovery (WSD EventsSecure-In)" new enable=yes

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
```

- Inbound and outbound

```
netsh advfirewall firewall set rule group="Remote Desktop" new enable=yes

netsh advfirewall firewall set rule group="Core Networking" new enable=yes
```

- Outbound

```
netsh advfirewall firewall set rule dir=out name="Network Discovery (LLMNR-UDP-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (NB-Datagram-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (NB-Name-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (Pub-WSD-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (SSDP-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (UPnPHost-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (UPnP-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (WSD Events-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (WSD EventsSecure-Out)" new enable=yes  
netsh advfirewall firewall set rule dir=out name="Network Discovery (WSD-Out)" new enable=yes
```

Verify VM is healthy, secure, and accessible with RDP

1. In the command prompt window, run `winmgmt /verifyrepository` to confirm that the Windows Management Instrumentation (WMI) repository is consistent. If the repository is corrupted, see the blog post [WMI: Repository Corruption, or Not?](#)
2. Set the Boot Configuration Data (BCD) settings:

```
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. Remove any extra Transport Driver Interface filters, such as software that analyzes TCP packets.
4. To make sure the disk is healthy and consistent, run the `CHKDSK /#` command in the command prompt window. Type "Y" to schedule the check and restart the VM.
5. Uninstall any other third-party software and driver related to physical components or any other virtualization technology.
6. Make sure that a third-party application is not using Port 3389. This port is used for the RDP service in Azure. You can run `netstat -anob` in the command prompt window to see the ports that are used by the applications.
7. If the Windows VHD that you want to upload is a domain controller, follow [these extra steps](#) to prepare the disk.
8. Reboot the VM to make sure that Windows is still healthy can be reached by using the RDP connection.
9. 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."

Install Windows Updates

Install the latest updates for Windows. If that's not possible, make sure that the following updates are installed:

- [KB3137061](#) Microsoft Azure VMs don't recover from a network outage and data corruption issues occur
- [KB3115224](#) Reliability improvements for VMs that are running on a Windows Server 2012 R2 or Windows Server 2012 host
- [KB3140410](#) MS16-031: Security update for Microsoft Windows to address elevation of privilege: March 8, 2016
- [KB3063075](#) Many ID 129 events are logged when you run a Windows Server 2012 R2 virtual machine in Microsoft Azure
- [KB3137061](#) Microsoft Azure VMs don't recover from a network outage and data corruption issues occur
- [KB3114025](#) Slow performance when you access Azure files storage from Windows 8.1 or Server 2012 R2
- [KB3033930](#) Hotfix increases the 64K limit on RIO buffers per process for Azure service in Windows
- [KB3004545](#) You cannot access virtual machines that are hosted on Azure hosting services through a VPN connection in Windows
- [KB3082343](#) Cross-Premises VPN connectivity is lost when Azure site-to-site VPN tunnels use Windows Server 2012 R2 RRAS
- [KB3140410](#) MS16-031: Security update for Microsoft Windows to address elevation of privilege: March 8, 2016
- [KB3146723](#) MS16-048: Description of the security update for CSRSS: April 12, 2016
- [KB2904100](#) System freezes during disk I/O in Windows

Run Sysprep

If you want to create an image to deploy to multiple VMs, you need to [generalize the image by running Sysprep](#) before you upload the VHD to Azure. You don't need to run Sysprep to use a specialized VHD. For more information, see the following articles:

- [Generalize a Windows virtual machine using Sysprep](#)
- [Sysprep Support for Server Roles](#)

Complete recommended configurations

The following settings do not affect VHD uploading. However, we strongly recommend that you have them configured.

- Install the [Azure Virtual Machines Agent](#). After you install the agent, you can enable VM extensions. The VM extensions implement most of the critical functionality that you want to use with your VMs like resetting passwords, configuring RDP, and many others.
- The Dump log can be helpful in troubleshooting Windows crash issues. Enable the Dump log collection:

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\CrashControl" /v CrashDumpEnabled /t REG_DWORD /d 2 /f

REG ADD "HKLM\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps" /v DumpFolder /t
REG_EXPAND_SZ /d "c:\CrashDumps" /f

REG ADD "HKLM\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps" /v DumpCount /t REG_DWORD
/d 10 /f

REG ADD "HKLM\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps" /v DumpType /t REG_DWORD
/d 2 /f

sc config wer start= auto
```

- After the VM is created in Azure, configure the system defined size pagefile on drive D:

```
REG ADD "HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management" /t REG_MULTI_SZ /v  
PagingFiles /d "D:\pagefile.sys 0 0" /f
```

Next steps

- [Upload a Windows VM image to Azure for Resource Manager deployments](#)

Generalize a Windows virtual machine using Sysprep

1/17/2017 • 1 min to read • [Edit on GitHub](#)

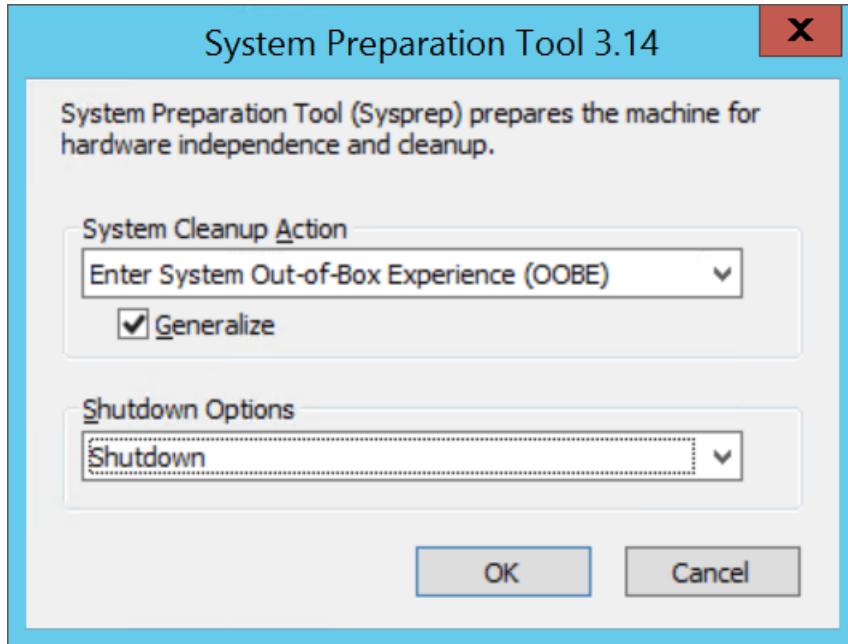
This section shows you how to generalize your Windows virtual machine for use as an image. Sysprep removes all your personal account information, among other things, and prepares the machine to be used as an image. For details about Sysprep, see [How to Use Sysprep: An Introduction](#).

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.

Next Steps

- If the VM is on-premises, you can now [upload the VHD to Azure](#).
- If the VM is already in Azure, you can now [create an image from the generalized VM](#).

Upload a Windows VHD from an on-premises VM to Azure

1/17/2017 • 3 min to read • [Edit on GitHub](#)

This article shows you how to create and upload a Windows virtual hard disk (VHD) to be used in creating an Azure VM. You can upload a VHD from either a generalized VM or a specialized VM.

For more details about disks and VHDs in Azure, see [About disks and VHDs for virtual machines](#).

Prepare the VM

You can upload both generalized and specialized VHDs to Azure. Each type requires that you prepare the VM before starting.

- **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 VHD to upload to Azure](#).
 - [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.

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](#).

1. 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
```

2. Get the subscription IDs for your available subscriptions.

```
Get-AzureRmSubscription
```

3. Set the correct subscription using the subscription ID. Replace <subscriptionID> 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.

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 **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"
```

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 image 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

Next steps

- [Create a VM in Azure from a generalized VHD](#)
- [Create a VM in Azure from a specialized VHD](#) by attaching it as an OS disk when you create a new VM.

Azure Virtual Machine Agent overview

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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.

How to capture a VM image from a generalized Azure VM

1/17/2017 • 2 min to read • [Edit on GitHub](#)

This article shows you how to use Azure PowerShell to create an image of a generalized Azure VM. You can then use the image to create another VM. The image includes the OS disk and the data disks that are attached to the virtual machine. The image doesn't include the virtual network resources, so you need to set up those resources when you create the new VM.

Prerequisites

- You need to have already [generalized the VM](#). Generalizing a VM removes all your personal account information, among other things, and prepares the machine to be used as an image.
- You need to have Azure PowerShell version 1.0.x or newer installed. If you haven't already installed PowerShell, read [How to install and configure Azure PowerShell](#) for installation steps.

Log in to Azure PowerShell

1. Open Azure PowerShell and sign in to your Azure account.

```
Login-AzureRmAccount
```

A pop-up window opens for you to enter your Azure account credentials.

2. Get the subscription IDs for your available subscriptions.

```
Get-AzureRmSubscription
```

3. Set the correct subscription using the subscription ID.

```
Select-AzureRmSubscription -SubscriptionId "<subscriptionID>"
```

Deallocate the VM and set the state to generalized

1. Deallocate the VM resources.

```
Stop-AzureRmVM -ResourceGroupName <resourceGroup> -Name <vmName>
```

The *Status* for the VM in the Azure portal changes from **Stopped** to **Stopped (deallocated)**.

2. Set the status of the virtual machine to **Generalized**.

```
Set-AzureRmVm -ResourceGroupName <resourceGroup> -Name <vmName> -Generalized
```

3. Check the status of the VM. The **OSState/generalized** section for the VM should have the **DisplayStatus** set to **VM generalized**.

```
$vm = Get-AzureRmVM -ResourceGroupName <resourceGroup> -Name <vmName> -Status  
$vm.Statuses
```

Create the image

1. Copy the virtual machine image to the destination storage container using this command. The image is created in the same storage account as the original virtual machine. The `-Path` parameter saves a copy of the JSON template locally. The `-DestinationContainerName` parameter is the name of the container that you want to hold your images. If the container doesn't exist, it is created for you.

```
Save-AzureRmVMImage -ResourceGroupName <resourceGroupName> -Name <vmName> `  
-DestinationContainerName <destinationContainerName> -VHDNamePrefix <templateNamePrefix> `  
-Path <C:\local\filepath\Filename.json>
```

You can get the URL of your image from the JSON file template. Go to the **resources > storageProfile > osDisk > image > uri** section for the complete path of your image. The URL of the image looks like:

```
https://<storageAccountName>.blob.core.windows.net/system/Microsoft.Compute/Images/<imagesContainer>/<templatePrefix>-osDisk>.xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.vhd
```

You can also verify the URI in the portal. The image is copied to a container named **system** in your storage account.

Next steps

- Now you can [create a VM from the image](#).

Create a copy of a specialized Windows VM running in Azure

1/17/2017 • 3 min to read • [Edit on GitHub](#)

This article shows you how to use the AZCopy tool to create a copy of the VHD from a specialized Windows VM that is running in Azure. You can then use the copy of the VHD to create a new VM.

- If want to copy a generalized VM, see [How to create a VM image from an existing generalized Azure VM](#).
- If you want to upload a VHD from an on-premises VM, like one created using Hyper-V, the see [Upload a Windows VHD from an on-premises VM to Azure](#).

Before you begin

Make sure that you:

- Have information about the **source and destination storage accounts**. For the source VM, you need to storage account and container names. Usually, the container name will be **vhd**s. You also need to have a destination storage account. If you don't already have one, you can create one using either the portal ([More Services](#) > Storage accounts > Add) or using the [New-AzureRmStorageAccount](#) cmdlet.
- Have Azure [PowerShell 1.0](#) (or later) installed.
- Have downloaded and installed the [AzCopy tool](#).

Deallocate the VM

Deallocate the VM, which frees up the VHD to be copied.

- **Portal:** Click **Virtual machines** > **myVM** > Stop
- **Powershell:** `Stop-AzureRmVM -ResourceGroupName myResourceGroup -Name myVM` deallocates the VM named **myVM** in resource group **myResourceGroup**.

The **Status** for the VM in the Azure portal changes from **Stopped** to **Stopped (deallocated)**.

Get the storage account URLs

You need the URLs of the source and destination storage accounts. The URLs look like:

`https://<storageaccount>.blob.core.windows.net/<containerName>/`. If you already know the storage account and container name, you can just replace the information between the brackets to create your URL.

You can use the Azure portal or Azure Powershell to get the URL:

- **Portal:** Click **More services** > **Storage accounts** > **Blobs** and your source VHD file is probably in the **vhd**s container. Click **Properties** for the container, and copy the text labeled **URL**. You'll need the URLs of both the source and destination containers.
- **Powershell:** `Get-AzureRmVM -ResourceGroupName "myResourceGroup" -Name "myVM"` gets the information for VM named **myVM** in the resource group **myResourceGroup**. In the results, look in the **Storage profile** section for the **Vhd Uri**. The first part of the Uri is the URL to the container and the last part is the OS VHD name for the VM.

Get the storage access keys

Find the access keys for the source and destination storage accounts. For more information about access keys, see [About Azure storage accounts](#).

- **Portal:** Click **More services > Storage accounts > All Settings > Access keys**. Copy the key labeled as **key1**.
- **Powershell:** `Get-AzureRmStorageAccountKey -Name mystorageaccount -ResourceGroupName myResourceGroup` gets the storage key for the storage account **mystorageaccount** in the resource group **myResourceGroup**. Copy the key labeled as **key1**.

Copy the VHD

You can copy files between storage accounts using AzCopy. For the destination container, if the specified container doesn't exist, it will be created for you.

To use AzCopy, open a command prompt on your local machine and navigate to the folder where AzCopy is installed. It will be similar to *C:\Program Files (x86)\Microsoft SDKs\Azure\AzCopy*.

To copy all of the files within a container, you use the **/S** switch. This can be used to copy the OS VHD and all of the data disks if they are in the same container. This example shows how to copy all of the files in the container **mysourcecontainer** in storage account **mysourcestorageaccount** to the container **mydestinationcontainer** in the **mydestinationstorageaccount** storage account. Replace the names of the storage accounts and containers with your own. Replace `<sourceStorageAccountKey1>` and `<destinationStorageAccountKey1>` with your own keys.

```
AzCopy /Source:https://mysourcestorageaccount.blob.core.windows.net/mysourcecontainer `  
/Dest:https://mydestinationstorageaccount.blob.core.windows.net/mydestinationcontainer `  
/SourceKey:<sourceStorageAccountKey1> /DestKey:<destinationStorageAccountKey1> /S
```

If you only want to copy a specific VHD in a container with multiple files, you can also specify the file name using the **/Pattern** switch. In this example, only the file named **myFileName.vhd** will be copied.

```
AzCopy /Source:https://mysourcestorageaccount.blob.core.windows.net/mysourcecontainer `  
/Dest:https://mydestinationstorageaccount.blob.core.windows.net/mydestinationcontainer `  
/SourceKey:<sourceStorageAccountKey1> /DestKey:<destinationStorageAccountKey1> `  
/Pattern:myFileName.vhd
```

When it is finished, you will get a message that looks something like:

```
Finished 2 of total 2 file(s).  
[2016/10/07 17:37:41] Transfer summary:  
-----  
Total files transferred: 2  
Transfer successfully: 2  
Transfer skipped: 0  
Transfer failed: 0  
Elapsed time: 00.00:13:07
```

Troubleshooting

- When you use AZCopy, if you see the error "Server failed to authenticate the request. Make sure the value of Authorization header is formed correctly including the signature." and you are using Key 2 or the secondary storage key, try using the primary or 1st storage key.

Next steps

- You can create a new VM by [attaching the copy of the VHD to a VM as an OS disk](#).

Create a VM from a generalized VHD image

1/17/2017 • 4 min to read • [Edit on GitHub](#)

A generalized VHD image has had all of your personal account information removed using [Sysprep](#). You can create a generalized VHD by running Sysprep on an on-premises VM, then [uploading the VHD to Azure](#), or by running Sysprep on an existing Azure VM and then [copying the VHD](#).

If you want to create a VM from a specialized VHD, see [Create a VM from a specialized VHD](#).

The quickest way to create a VM from a generalized VHD is to use a [quick start template](#).

Prerequisites

If you are going to use a VHD uploaded from an on-premises VM, like one created using Hyper-V, you should make sure you followed the directions in [Prepare a Windows VHD to upload to Azure](#).

Both uploaded VHDs and existing Azure VM VHDs need to be generalized before you can create a VM using this method. For more information, see [Generalize a Windows virtual machine using Sysprep](#).

Set the URI of the VHD

The URI for the VHD to use is in the format:

`https://mystorageaccount.blob.core.windows.net/mycontainer/MyVhdName.vhf`. In this example the VHD named **myVHD** is in the storage account **mystorageaccount** in the container **mycontainer**.

```
$imageURI = "https://mystorageaccount.blob.core.windows.net/mycontainer/myVhd.vhd"
```

Create a virtual network

Create the vNet and subnet of the [virtual network](#).

1. Create the subnet. The following sample creates a subnet named **mySubnet** in the resource group **myResourceGroup** with the address prefix of **10.0.0.0/24**.

```
$rgName = "myResourceGroup"
$subnetName = "mySubnet"
$singleSubnet = New-AzureRmVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix 10.0.0.0/24
```

2. Create the virtual network. The following sample creates a virtual network named **myVnet** in the **West US** location with the address prefix of **10.0.0.0/16**.

```
$location = "West US"
$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 security rule 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"

$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 $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
```

Create the VM

The following PowerShell script shows how to set up the virtual machine configurations and use the uploaded VM image as the source for the new installation.

```

# Enter a new user name and password to use as the local administrator account
# for remotely accessing the VM.
$cred = Get-Credential

# Name of the storage account where the VHD is located. This example sets the
# storage account name as "myStorageAccount"
$storageAccName = "myStorageAccount"

# Name of the virtual machine. This example sets the VM name as "myVM".
$vmName = "myVM"

# Size of the virtual machine. This example creates "Standard_D2_v2" sized VM.
# See the VM sizes documentation for more information:
# https://azure.microsoft.com/documentation/articles/virtual-machines-windows-sizes/
$vmSize = "Standard_D2_v2"

# Computer name for the VM. This examples sets the computer name as "myComputer".
$computerName = "myComputer"

# Name of the disk that holds the OS. This example sets the
# OS disk name as "myOsDisk"
$osDiskName = "myOsDisk"

# Assign a SKU name. This example sets the SKU name as "Standard_LRS"
# 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.
$skuName = "Standard_LRS"

# Get the storage account where the uploaded image is stored
$storageAcc = Get-AzureRmStorageAccount -ResourceGroupName $rgName -AccountName $storageAccName

# Set the VM name and size
$vmConfig = New-AzureRmVMConfig -VMName $vmName -VMSize $vmSize

#Set the Windows operating system configuration and add the NIC
$vm = Set-AzureRmVMOperatingSystem -VM $vmConfig -Windows -ComputerName $computerName `

-Credential $cred -ProvisionVMAgent -EnableAutoUpdate
$vm = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id

# Create the OS disk URI
$osDiskUri = '{0}vhds/{1}-{2}.vhf' `

-f $storageAcc.PrimaryEndpoints.Blob.ToString(), $vmName.ToLower(), $osDiskName

# Configure the OS disk to be created from the existing VHD image (-CreateOption fromImage).
$vm = Set-AzureRmVMOSDisk -VM $vm -Name $osDiskName -VhdUri $osDiskUri `

-CreateOption fromImage -SourceImageUri $imageURI -Windows

# Create the new VM
New-AzureRmVM -ResourceGroupName $rgName -Location $location -VM $vm

```

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 [Manage virtual machines using Azure Resource Manager and PowerShell](#).

Create a VM from a specialized VHD

1/17/2017 • 4 min to read • [Edit on GitHub](#)

Create a new VM by attaching a specialized VHD as the OS disk using Powershell. A specialized VHD maintains the user accounts, applications and other state data from your original VM.

If you want to create a VM from a generalized VHD, see [Create a VM from a generalized VHD image](#).

Create the subNet and vNet

Create the vNet and subNet of the [virtual network](#).

1. Create the subNet. This example creates a subnet named **mySubNet**, in the resource group **myResourceGroup**, and sets the subnet address prefix to **10.0.0.0/24**.

```
$rgName = "myResourceGroup"  
$subnetName = "mySubNet"  
$singleSubnet = New-AzureRmVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix 10.0.0.0/24
```

2. 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**.

```
$location = "West US"  
$vnetName = "myVnetName"  
$vnet = New-AzureRmVirtualNetwork -Name $vnetName -ResourceGroupName $rgName -Location $location `  
-AddressPrefix 10.0.0.0/16 -Subnet $singleSubnet
```

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.

1. Create the public IP. In this example, the public IP address name is set to **myIP**.

```
$ipName = "myIP"  
$pip = New-AzureRmPublicIpAddress -Name $ipName -ResourceGroupName $rgName -Location $location `  
-AllocationMethod Dynamic
```

2. Create the NIC. In this example, the NIC name is set to **myNicName**.

```
$nicName = "myNicName"  
$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 an security rule that allows RDP access on port 3389. Because the VHD for the new VM was created from an existing specialized VM, after the VM is created you can use an existing account from the source virtual machine that had permission to log on using 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 $rgName -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 the VM configuration

Set up the VM configuration to attach the copied VHD as the OS VHD.

```

# Set the URI for the VHD that you want to use. In this example, the VHD file named "myOsDisk.vhd" is kept
# in a storage account named "myStorageAccount" in a container named "myContainer".
$osDiskUri = "https://myStorageAccount.blob.core.windows.net/myContainer/myOsDisk.vhd"

# 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 by using the URL of the copied OS VHD. In this example, when the OS disk is created, the
# term "osDisk" is appended to the VM name to create the OS disk name. This example also specifies that this
# Windows-based VHD should be attached to the VM as the OS disk.
$osDiskName = $vmName + "osDisk"
$vm = Set-AzureRmVMOSDisk -VM $vm -Name $osDiskName -VhdUri $osDiskUri -CreateOption attach -Windows

```

If you have data disks that need to be attached to the VM, you should also add the following:

```

# Optional: Add data disks by using the URLs of the copied data VHDs at the appropriate Logical Unit
# Number (Lun).
$dataDiskName = $vmName + "dataDisk"
$vm = Add-AzureRmVMDataDisk -VM $vm -Name $dataDiskName -VhdUri $dataDiskUri -Lun 0 -CreateOption attach

```

The data and operating system disk URLs look something like this:

<https://StorageAccountName.blob.core.windows.net/BlobContainerName/DiskName.vhd>. You can find this on the portal by browsing to the target storage container, clicking the operating system or data VHD that was copied, and then copying the contents of the URL.

Create the VM

Create the VM using the configurations that we just created.

```

#Create the new VM
New-AzureRmVM -ResourceGroupName $rgName -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 $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](#).

Create an availability set

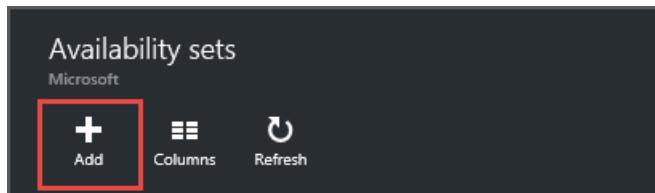
1/17/2017 • 3 min to read • [Edit on GitHub](#)

When using the portal, if you want your VM to be part of an availability set, you need to create the availability set first.

For more information about creating and using availability sets, see [Manage the availability of virtual machines](#).

Use the portal to create an availability set before creating your VMs

1. In the hub menu, click **Browse** and select **Availability sets**.
2. On the **Availability sets blade**, click **Add**.



3. On the **Create availability set** blade, complete the information for your set.

The 'Create availability set' blade contains the following fields:

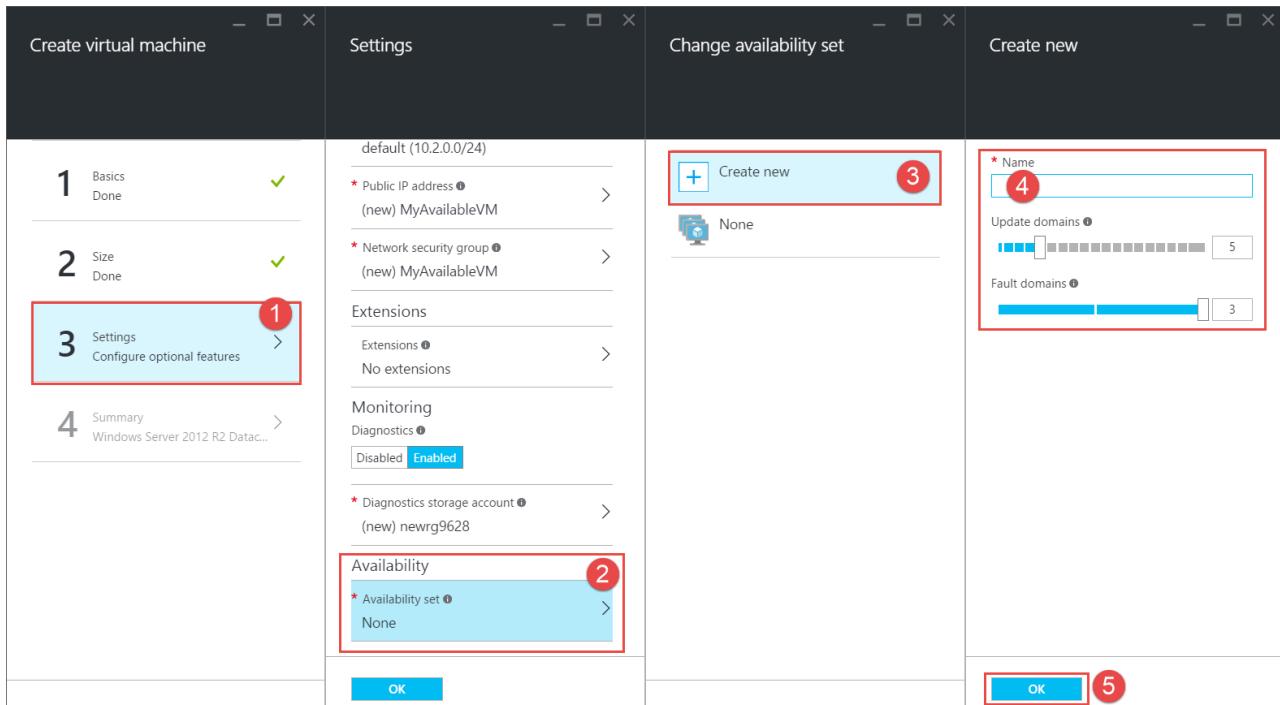
- Name**: An empty text input field.
- Fault domains**: A slider set to 3.
- Update domains**: A slider set to 5.
- Subscription**: A dropdown menu showing "Visual Studio Ultimate with MSDN".
- Resource group**: A dropdown menu showing "+ New".
- New resource group name**: An empty text input field.
- Location**: A dropdown menu showing "West US".
- A large note area at the bottom with placeholder text: "Note: This blade is used to create an availability set before creating your VMs. You can also use the blade to edit or delete an existing availability set."

- **Name** - the name should be 1-80 characters made up of numbers, letters, periods, underscores and dashes. The first character must be a letter or number. The last character must be a letter, number or underscore.
- **Fault domains** - fault domains define the group of virtual machines that share a common power source and network switch. By default, the VMs are separated across up to three fault domains and can be changed to between 1 and 3.
- **Update domains** - five update domains are assigned by default and this can be set to between 1 and 20. Update domains indicate groups of virtual machines and underlying physical hardware that can be rebooted at the same time. For example, if we specify five update domains, when more than five virtual machines are configured within a single Availability Set, the sixth virtual machine will be placed into the same update domain as the first virtual machine, the seventh in the same UD as the second virtual machine, and so on. The order of the reboots may not be sequential, but only one update domain will be rebooted at a time.
- **Subscription** - select the subscription to use if you have more than one.
- **Resource group** - select an existing resource group by clicking the arrow and selecting a resource group from the drop down. You can also create a new resource group by typing in a name. The name can contain any of the following characters: letters, numbers, periods, dashes, underscores and opening or closing parenthesis. The name cannot end in a period. All of the VMs in the availability group need to be created in the same resource group as the availability set.
- **Location** - select a location from the drop-down.

4. When you are done entering the information, click **Create**. Once the availability group has been created, you can see it in the list after you refresh the portal.

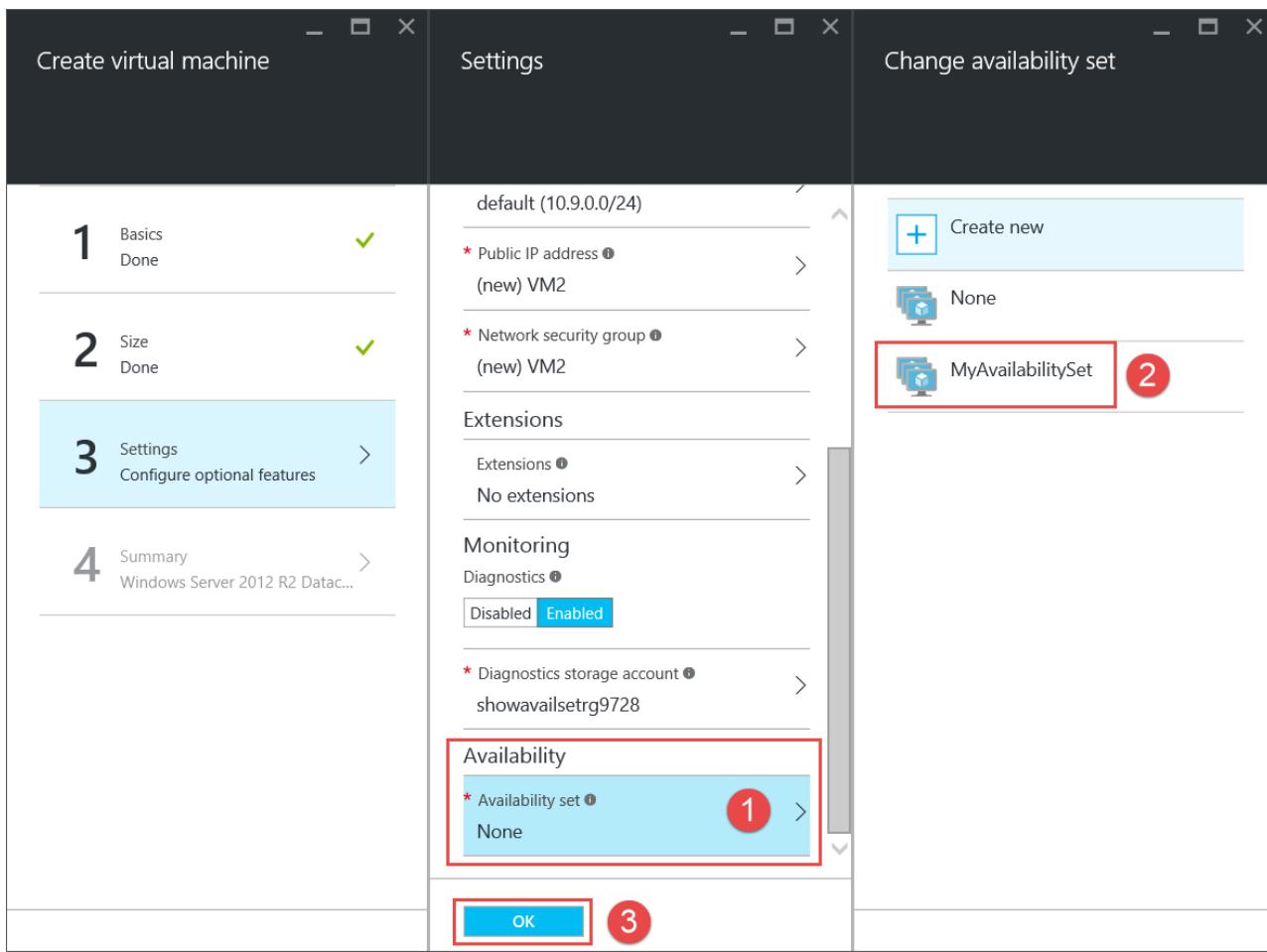
Use the portal to create a virtual machine and an availability set at the same time

If you are creating a new VM using the portal, you can also create a new availability set for the VM while you create the first VM in the set.



Add a new VM to an existing availability set

For each additional VM that you create that should belong in the set, make sure that you create it in the same **resource group** and then select the existing availability set in Step 3.



Use PowerShell to create an availability set

This example creates an availability set in the **RMResGroup** resource group in the **West US** location. This needs to be done before you create the first VM that will be in the set.

```
New-AzureRmAvailabilitySet -ResourceGroupName "RMResGroup" -Name "AvailabilitySet03" -Location "West US"
```

For more information, see [New-AzureRmAvailabilitySet](#).

Troubleshooting

- When you create a VM, if the availability set you want isn't in the drop-down list in the portal you may have created it in a different resource group. If you don't know the resource group for your availability set, go to the hub menu and click **Browse > Availability sets** to see a list of your availability sets and which resource groups they belong to.

Next steps

Add additional storage to your VM by adding an additional [data disk](#).

Change the availability set for a Windows VM

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 [Extension Configuration Samples](#). 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-azurermvm -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.NetworkInterfaceIDs) {
    Add-AzureRmVMNetworkInterface -VM $NewVM -Id $nic
}

#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](#).

Move a Windows VM to another Azure subscription or resource group

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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.

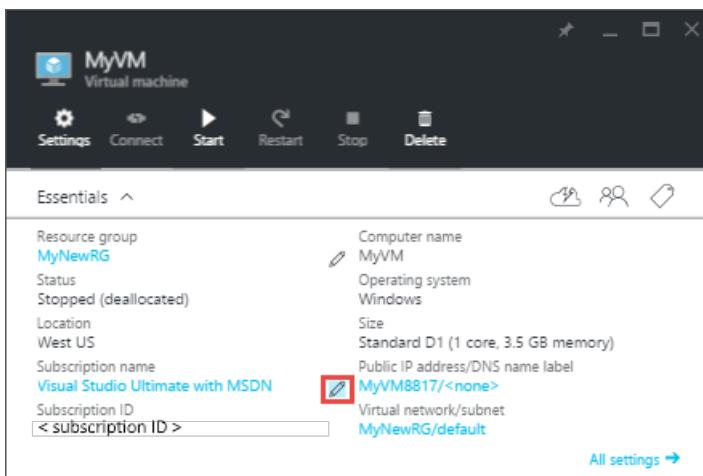
NOTE

New resource IDs will be created as part of the move. Once the VM has been moved, you will need to update your tools and scripts to use the new resource IDs.

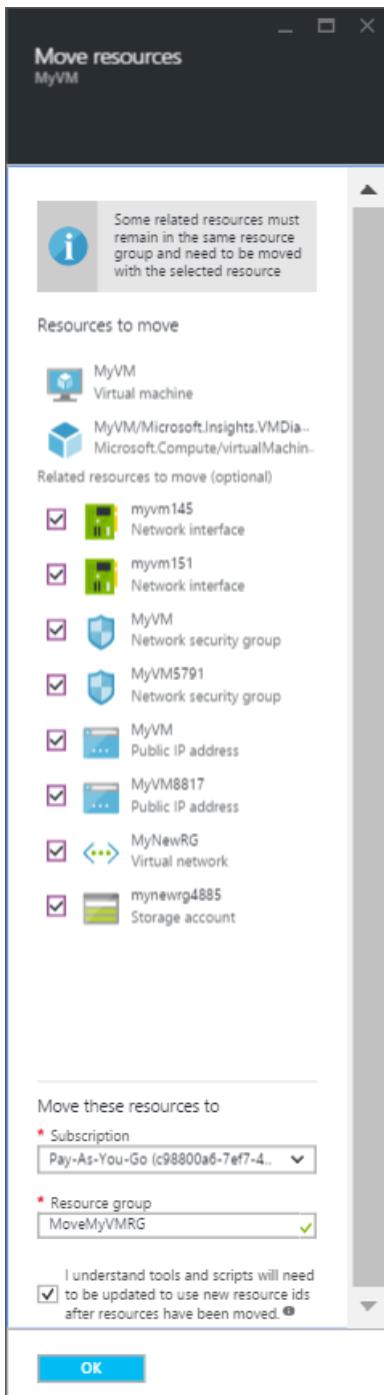
Use the portal to move a VM to a different subscription

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. In the **Essentials** section, click on the **Change subscription** pencil icon next to the subscription name. The **Move resources** blade will open.



4. Select each of the resources to move. In most cases, you should move all of the listed optional resources.
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 will be created and those need to be used with the VM once it is moved, then click **OK**.

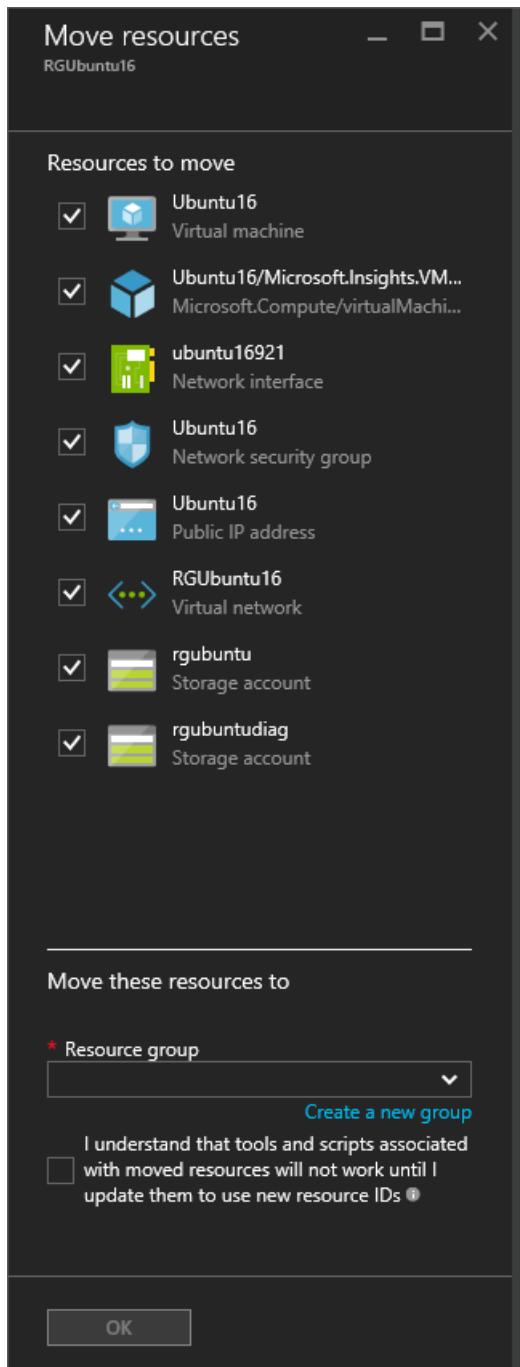
Use the portal to move a VM to another resource group

You can move a VM and its associated resources to another resource group using the portal.

1. Open the [Azure portal](#).
2. Click **Browse > Resource groups** and select the resource group that contains the VM.
3. In the **Resource group** blade, select **Move** from the menu.

→ Move

4. In the **Move resources** blade, select the resources to be moved and then either type an existing resource group name or choose to create a new resource group. When you are done, select that you understand that new resource IDs will be 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 resource name and the type of resource. You can get both from the `Find-AzureRMResource` cmdlet.

```
Find-AzureRMResource -ResourceGroupNameContains "<sourceResourceGroupName>"
```

To move a VM we need to move multiple resources. We can just create separate variables for each resource and then list them. This example includes most of the basic resources for a VM, but you can add more as needed.

```

$sourceRG = "<sourceResourceGroupName>"
$destinationRG = "<destinationResourceGroupName>

$vm = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType "Microsoft.Compute/virtualMachines" -
    ResourceName "<vmName>"
$storageAccount = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType
    "Microsoft.Storage/storageAccounts" -ResourceName "<storageAccountName>"
$diagStorageAccount = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType
    "Microsoft.Storage/storageAccounts" -ResourceName "<diagnosticStorageAccountName>"
$vNet = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType "Microsoft.Network/virtualNetworks" -
    ResourceName "<vNetName>"
$nic = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType "Microsoft.Network/networkInterfaces" -
    ResourceName "<nictName>"
$ip = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType "Microsoft.Network/publicIPAddresses" -
    ResourceName "<ipName>"
$nsg = Get-AzureRmResource -ResourceGroupName $sourceRG -ResourceType "Microsoft.Network/networkSecurityGroups" -
    ResourceName "<nsgName>"

Move-AzureRmResource -DestinationResourceGroupName $destinationRG -ResourceId $vm.ResourceId,
    $storageAccount.ResourceId, $diagStorageAccount.ResourceId, $vNet.ResourceId, $nic.ResourceId, $ip.ResourceId,
    $nsg.ResourceId

```

To move the resources to different subscription, include the **-DestinationSubscriptionId** parameter.

```

Move-AzureRmResource -DestinationSubscriptionId "<destinationSubscriptionID>" -DestinationResourceGroupName
    $destinationRG -ResourceId $vm.ResourceId, $storageAccount.ResourceId, $diagStorageAccount.ResourceId,
    $vNet.ResourceId, $nic.ResourceId, $ip.ResourceId, $nsg.ResourceId

```

You will be asked to confirm that you want to move the specified resources. Type **Y** to confirm that you want to move the 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

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 reset local Windows password for Azure VM

1/17/2017 • 4 min to read • [Edit on GitHub](#)

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:

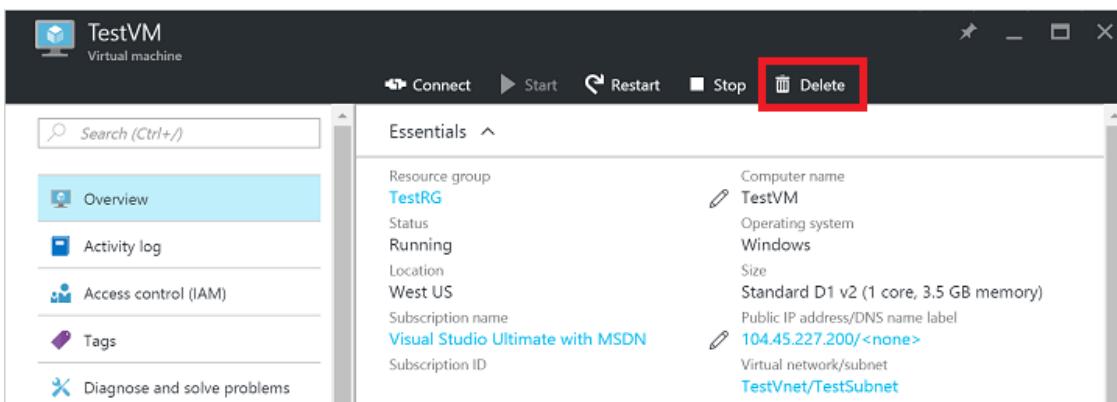
- Delete the source VM. The virtual disks are retained.
- Attach the source VM's OS disk to another VM within your Azure subscription. This VM is referred to as the troubleshooting VM.
- Using the troubleshooting VM, create some config files on the source VM's OS disk.
- Detach the VM's OS disk from 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.

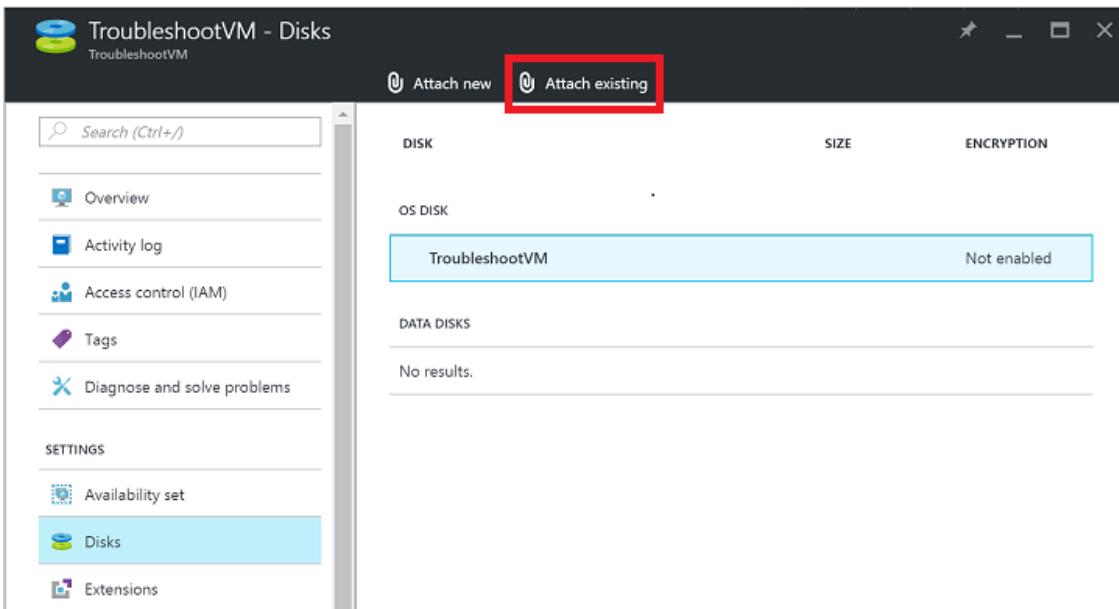
1. Delete the affected VM in Azure portal. Deleting the VM only deletes the metadata, the reference of the VM within Azure. The virtual disks are retained when the VM is deleted:

- Select the VM in the Azure portal, click *Delete*:

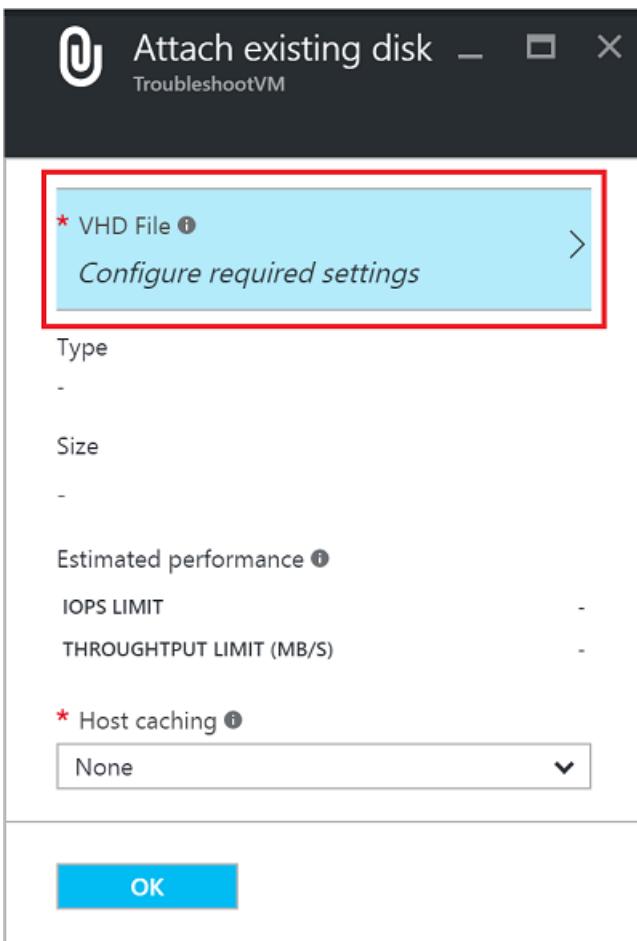


2. Attach the source VM's OS disk to the troubleshooting VM. The troubleshooting VM must be in the same region as the source VM's OS disk (such as `West US`):

- 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*:

Containers

testrgvmsstorage

+ Container Refresh

Search containers by prefix

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:

vhds

https://testrgvmsstorage.blob.core.windows.net/vhds

Refresh

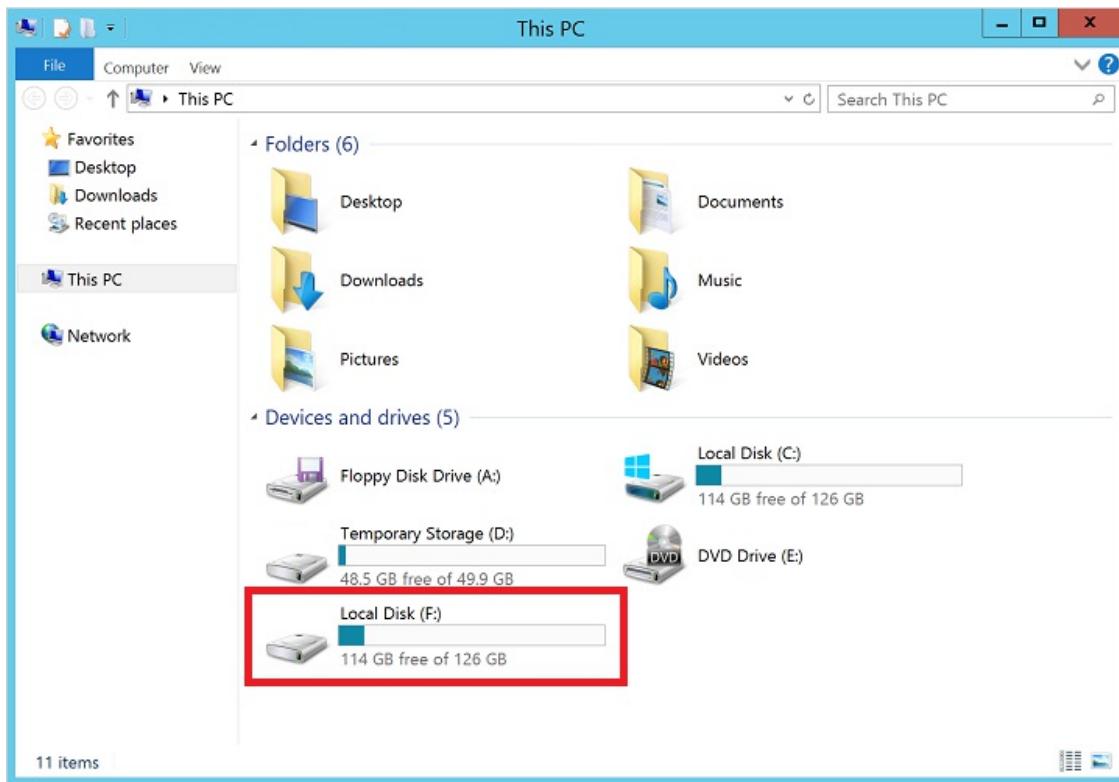
Search blobs by prefix (case-sensitive)

NAME	SIZE
TestVM201682142612.vhd	136.37 GB
TroubleshootVM.f3d1d6c8-9f2a-4e88-9644-4be3243f4bb0.status	242 B
TroubleshootVM20168214305.vhd	136.37 GB

Select

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:



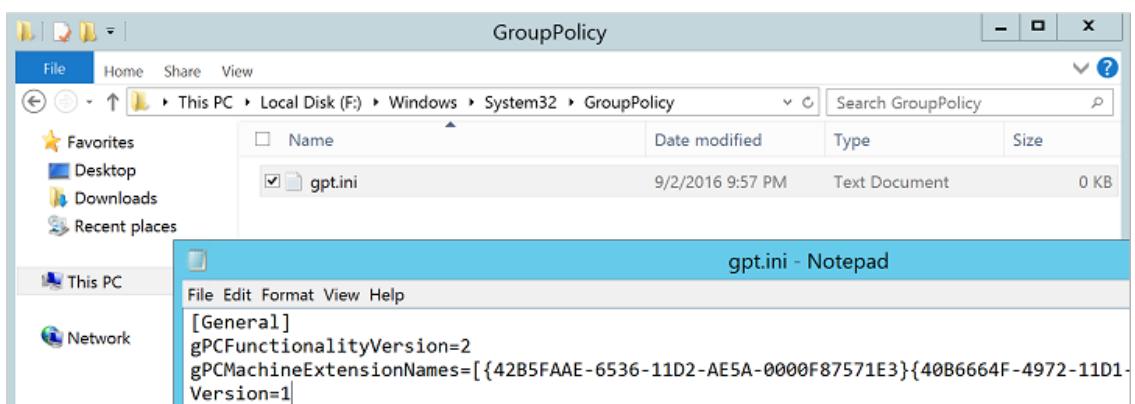
4. Create `gpt.ini` in `\Windows\System32\GroupPolicy` on the source VM's drive (if `gpt.ini` exists, rename to `gpt.ini.bak`):

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:

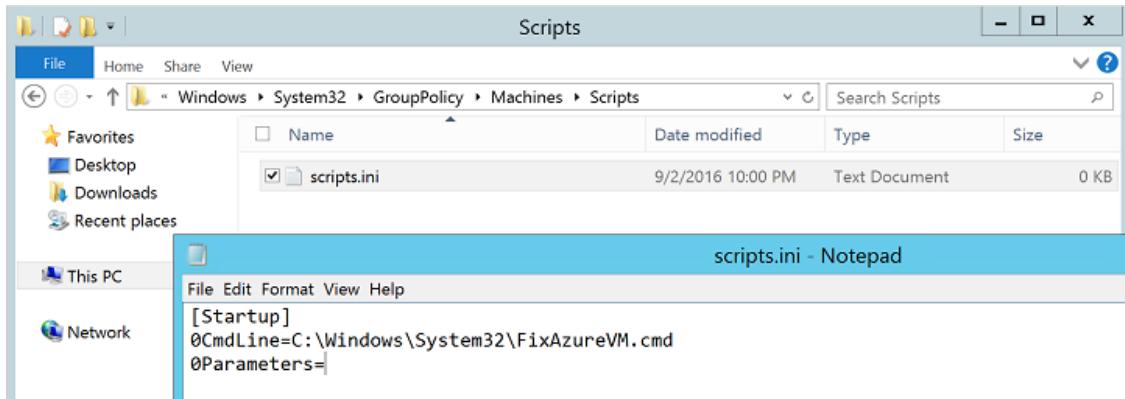
```
[General]
gPCFunctionalityVersion=2
gPCMchineExtensionNames=[{42B5FAAE-6536-11D2-AE5A-0000F87571E3}{40B6664F-4972-11D1-A7CA-
0000F87571E3}]
Version=1
```



5. Create `scripts.ini` in `\Windows\System32\GroupPolicy\Machine\Scripts`. Make sure hidden folders are shown. If needed, create the `Machine` or `Scripts` folders.

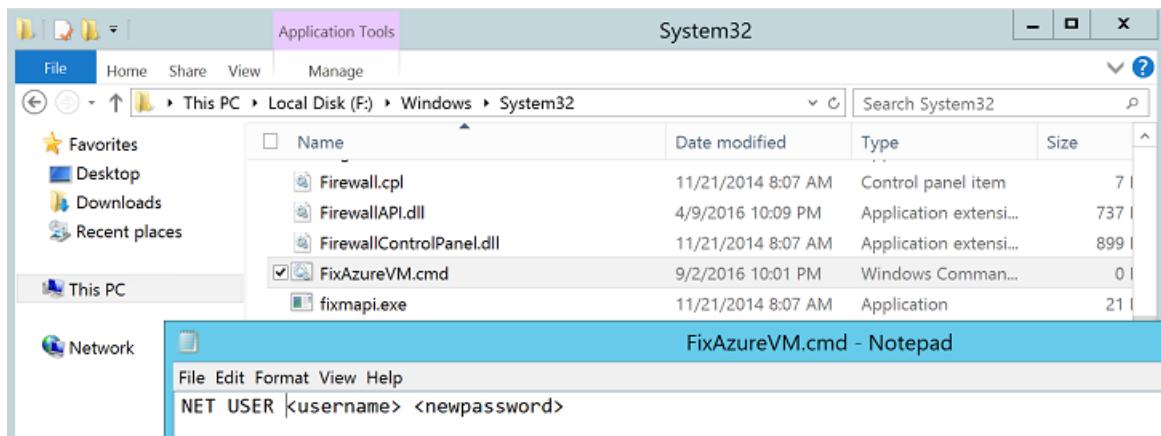
- Add the following lines the `scripts.ini` file you created:

```
[Startup]
0CmdLine=C:\Windows\System32\FixAzureVM.cmd
0Parameters=
```



6. Create `FixAzureVM.cmd` in `\Windows\System32` with the following contents, replacing `<username>` and `<newpassword>` with your own values:

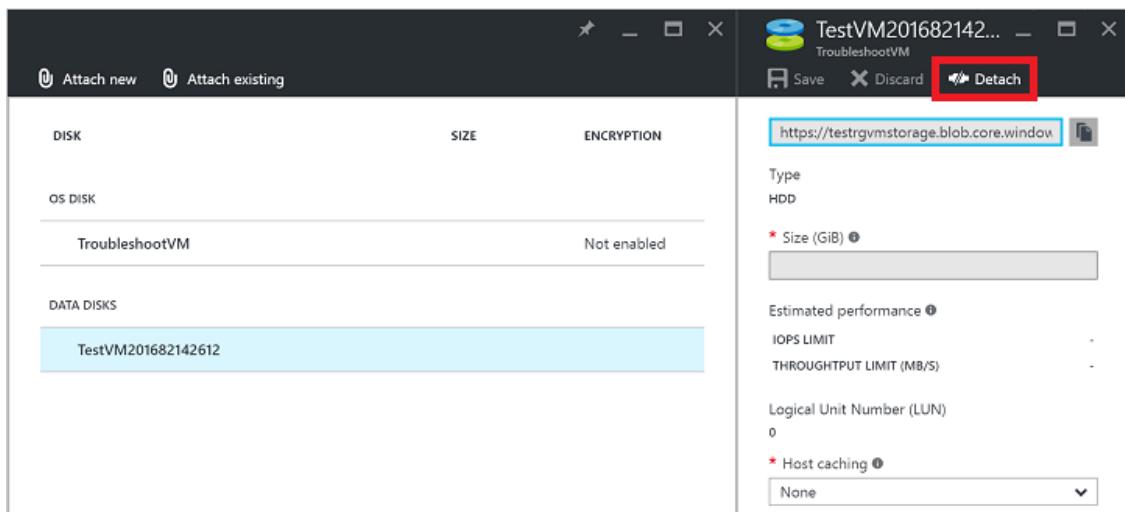
```
NET USER <username> <newpassword>
```



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 *Disk*.
- 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*:

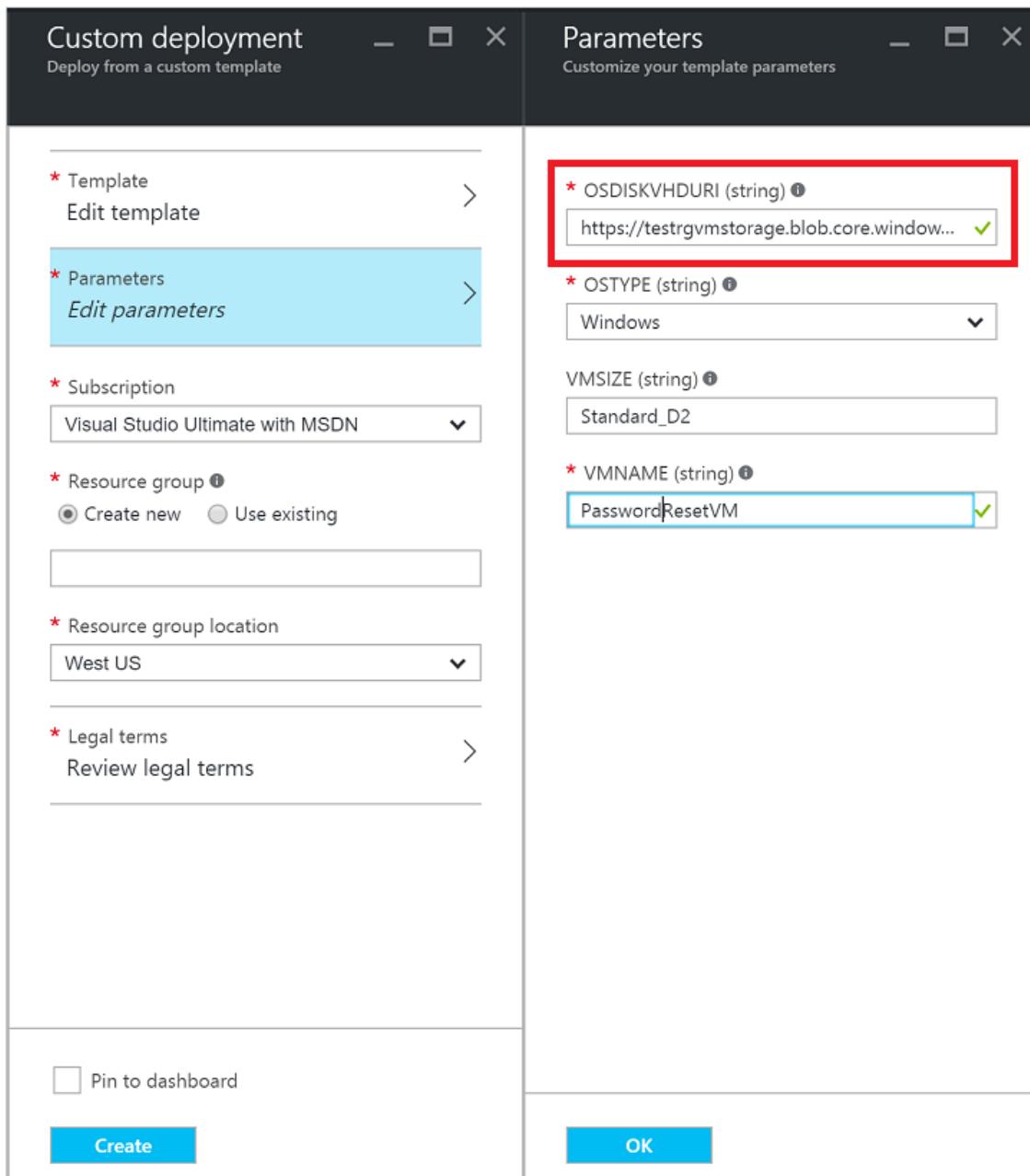
The screenshot shows two side-by-side Azure portal windows. The left window is titled 'Blob service' and shows a storage account named 'testrgvmstorage'. The right window shows a container named 'vhds' within the same storage account. Both windows have a red box highlighting the 'Blobs' service icon and the 'vhds' container name respectively.

Select your source VM OS VHD and click the *Copy* button next to the *URL* name:

The screenshot shows two windows. The left window is titled 'vhds' and lists blobs within the container. The right window is titled 'Blob properties' and shows details for a blob named 'TestVM201682142612.vhd'. A red box highlights the 'URL' field, which contains the value 'https://testrgvmstorage.blob.core.windows.net/vhds/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:



10. After the new VM is running, connect to the VM using Remote Desktop with the new password you specified in the `FixAzureVM.cmd` script.
11. 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.

1/17/2017 • 1 min to read • [Edit on GitHub](#)

How to tag a Windows virtual machine in Azure

1/17/2017 • 4 min to read • [Edit on GitHub](#)

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) ⓘ

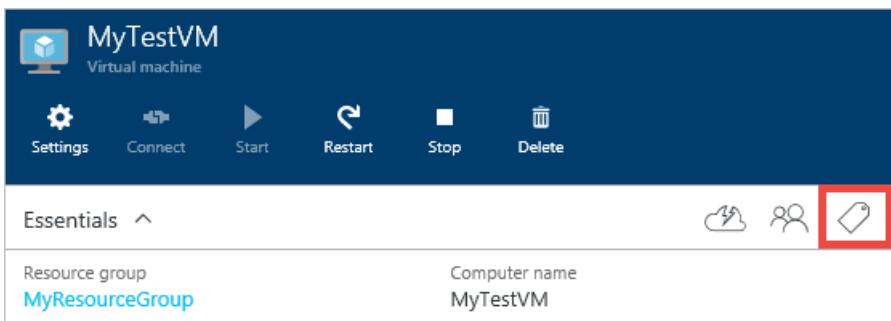
CREATEDBY (string) ⓘ

Click **Create** to deploy this template with your tag values.

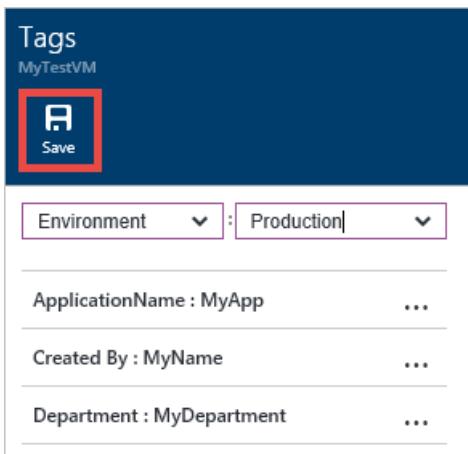
Tagging through the Portal

After creating your resources with tags, you can view, add, and delete tags in the portal.

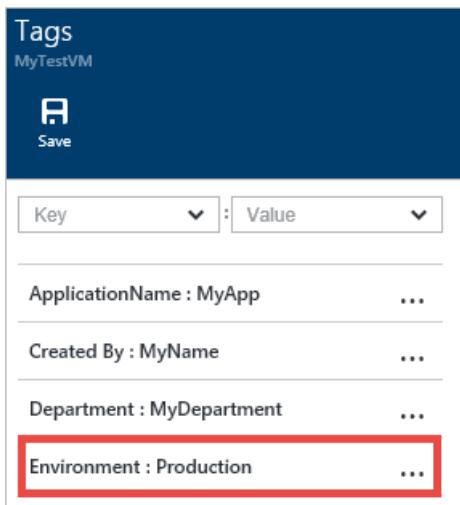
Select the tags icon to view your tags:



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 ▾

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](#).

Common PowerShell commands for creating and managing VMs

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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.

Create a VM

TASK	COMMAND
Create a VM configuration	\$vm = New-AzureRmVMConfig -VMName "vm_name" -VMSIZE "vm_size" 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 "computer_name" -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 "disk_name" -VhdUri "http://mystore1.blob.core.windows.netvhds/disk_name.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 "disk_name" -SourceImageUri "https://mystore1.blob.core.windows.net/system/Microsoft.C ompute/Images/myimages/myprefix-osDisk.{guid}.vhd" -VhdUri "https://mystore1.blob.core.windows.netvhds/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 "name_of_disk" -VhdUri "http://mystore1.blob.core.windows.netvhds/" -CreateOption Attach -Windows
Create a VM	[New-AzureRmVM]() -ResourceGroupName "resource_group_name" -Location "location_name" -VM \$vm All resources are created in a resource group . The VM must be created in the same location as the 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 "resource_group_name" To get a list of resource groups in your subscription, use Get-AzureRmResourceGroup .
Get information about a VM	Get-AzureRmVM -ResourceGroupName "resource_group_name" -Name "vm_name"

Manage VMs

TASK	COMMAND

TASK	COMMAND
Start a VM	<code>Start-AzureRmVM -ResourceGroupName "resource_group_name" -Name "vm_name"</code>
Stop a VM	<code>Stop-AzureRmVM -ResourceGroupName "resource_group_name" -Name "vm_name"</code>
Restart a running VM	<code>Restart-AzureRmVM -ResourceGroupName "resource_group_name" -Name "vm_name"</code>
Delete a VM	<code>Remove-AzureRmVM -ResourceGroupName "resource_group_name" -Name "vm_name"</code>
Generalize a VM	<p><code>Set-AzureRmVm -ResourceGroupName YourResourceGroup -Name "vm_name" -Generalized</code></p> <p>Run this command before you run <code>Save-AzureRmVMIImage</code>.</p>
Capture a VM	<p><code>Save-AzureRmVMIImage -ResourceGroupName "resource_group_name" -VMName "vm_name" -DestinationContainerName "image_container" -VHDNamePrefix "image_name_prefix" -Path "C:\filepath\filename.json"</code></p> <p>A virtual machine must be 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 "resource_group_name" -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 "disk_name" -VhdUri "https://mystore1.blob.core.windows.net/vhds/disk_name.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. For information about initializing disks as they are added, see Manage Azure Virtual Machines using Resource Manager and PowerShell.</p>
Remove a data disk from a VM	<p><code>Remove-AzureRmVMDataDisk -VM \$vm -Name "disk_name"</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>

TASK	COMMAND
Add an extension to a VM	<pre>Set-AzureRmVMExtension -ResourceGroupName "resource_group_name" -Location "azure_location" -VMName "vm_name" -Name "extension_name" -Publisher "publisher_name" -Type "extension_type" -TypeHandlerVersion "#.#" -Settings \$Settings -ProtectedSettings \$ProtectedSettings</pre> <p>Run this command with the appropriate configuration information for the extension that you want to install.</p>
Remove a VM extension	<pre>Remove-AzureRmVMExtension -ResourceGroupName "resource_group_name" -Name "extension_name" -VMName "vm_name"</pre>

Next steps

- See the basic steps for creating a virtual machine in [Create a Windows VM using Resource Manager and PowerShell](#).

Common Azure CLI commands for virtual machine tasks in the Resource Manager deployment model

1/17/2017 • 2 min to read • [Edit on GitHub](#)

This article shows common Azure Command-Line Interface (Azure CLI) commands to create and manage VMs in the Resource Manager deployment model.

Before you can use the Azure CLI with Resource Manager commands and templates to deploy Azure resources and workloads using resource groups, you will need an account with Azure. If you do not have an account, you can get a [free Azure trial here](#).

If you haven't already installed the Azure CLI and connected to your subscription, see [Install the Azure CLI](#) set the mode to `arm` with `azure config mode arm`, and connect to Azure with the `azure login` command.

Basic Azure Resource Manager commands in Azure CLI

This article covers basic commands you will want to use with Azure CLI to manage and interact with your ARM resources (primarily VMs) in your Azure subscription. For more detailed help with specific command line switches and options, you can use the online command help and options by typing `azure <command> <subcommand> --help` or `azure help <command> <subcommand>`.

NOTE

These examples don't include template-based operations which are generally recommended for VM deployments in Resource Manager. For information, see [Use the Azure CLI with Azure Resource Manager](#) and [Deploy and manage virtual machines by using Azure Resource Manager templates and the Azure CLI](#).

TASK	RESOURCE MANAGER
Create the most basic VM	<pre>azure vm quick-create [options] <resource-group> <name> <location> <os-type> <image-urn> <admin-username> <admin-password></pre> <p>(Obtain the <code>image-urn</code> from the <code>azure vm image list</code> command. See this article for examples.)</p>
Create a Linux VM	<pre>azure vm create [options] <resource-group> <name> <location> -y "Linux"</pre>
Create a Windows VM	<pre>azure vm create [options] <resource-group> <name> <location> -y "Windows"</pre>
List VMs	<pre>azure vm list [options]</pre>
Get information about a VM	<pre>azure vm show [options] <resource_group> <name></pre>
Start a VM	<pre>azure vm start [options] <resource_group> <name></pre>
Stop a VM	<pre>azure vm stop [options] <resource_group> <name></pre>

TASK	RESOURCE MANAGER
Deallocate a VM	<code>azure vm deallocate [options] <resource-group> <name></code>
Restart a VM	<code>azure vm restart [options] <resource_group> <name></code>
Delete a VM	<code>azure vm delete [options] <resource_group> <name></code>
Capture a VM	<code>azure vm capture [options] <resource_group> <name></code>
Create a VM from a user image	<code>azure vm create [options] -q <image-name> <resource-group> <name> <location> <os-type></code>
Create a VM from a specialized disk	<code>azue vm create [options] -d <os-disk-vhd> <resource-group> <name> <location> <os-type></code>
Add a data disk to a VM	<code>azure vm disk attach-new [options] <resource-group> <vm-name> <size-in-gb> [vhd-name]</code>
Remove a data disk from a VM	<code>azure vm disk detach [options] <resource-group> <vm-name> <lun></code>
Add a generic extension to a VM	<code>azure vm extension set [options] <resource-group> <vm-name> <name> <publisher-name> <version></code>
Add VM Access extension to a VM	<code>azure vm reset-access [options] <resource-group> <name></code>
Add Docker extension to a VM	<code>azure vm docker create [options] <resource-group> <name> <location> <os-type></code>
Remove a VM extension	<code>azure vm extension set [options] -u <resource-group> <vm-name> <name> <publisher-name> <version></code>
Get usage of VM resources	<code>azure vm list-usage [options] <location></code>
Get all available VM sizes	<code>azure vm sizes [options]</code>

Next steps

- For additional examples of the CLI commands going beyond basic VM management, see [Using the Azure CLI with Azure Resource Manager](#).

Creating a Work or School identity in Azure Active Directory to use with Windows VMs

1/17/2017 • 5 min to read • [Edit on GitHub](#)

If you created a personal Azure account or have a personal MSDN subscription and created the Azure account to take advantage of the MSDN Azure credits -- you used a *Microsoft account* identity to create it. Many great features of Azure -- [resource group templates](#) is one example -- require a work or school account (an identity managed by Azure Active Directory) to work. You can follow the instructions below to create a new work or school account because fortunately, one of the best things about your personal Azure account is that it comes with a default Azure Active Directory domain that you can use to create a new work or school account that you can use with Azure features that require it.

However, recent changes make it possible to manage your subscription with any type of Azure account using the `azure login` interactive login method described [here](#). You can either use that mechanism, or you can follow the instructions that follow. You can also [create a work or school identity in Azure Active Directory to use with Linux VMs](#).

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.

NOTE

If you were given a user name and password by an administrator, there's a good chance that you already have a work or school ID (also sometimes called an *organizational ID*). If so, you can immediately begin to use your Azure account to access Azure resources that require one. If you find that you cannot use those resources, you may need to return to this article for help. For more information, see [Accounts that you can use for sign in](#) and [How an Azure subscription is related to Azure AD](#).

The steps are simple. You need to locate your signed on identity in the Azure classic portal, discover your default Azure Active Directory domain, and add a new user to it as an Azure co-administrator.

Locate your default directory in the Azure classic portal

Start by logging in to the [Azure classic portal](#) with your personal Microsoft account identity. After you are logged in, scroll down the blue panel on the left side and click **ACTIVE DIRECTORY**.



Let's start by finding some information about your identity in Azure. You should see something like the following in the main pane, showing that you have one default directory.

active directory

Azure Active Directory					
NAME	STATUS	ROLE	SUBSCRIPTION	DATACENTER REGION	COUNTRY OR REGION
Default Directory	Active	Global Administrator	Shared by all Default Directories	United States	United States

Let's find out some more information about it. Click the default directory row, which brings you into the default directory properties.

default directory

Your directory is ready to use.
Here are a few options to get started.

Skip Quick Start the next time I visit

I WANT TO Set Up Directory Manage Access Develop Applications

To view the default domain name, click **DOMAINS**.

default directory

Your directory comes with a default domain,
aztrainpass[REDACTED]outlook.onmicrosoft.com. Add a custom domain to
improve user sign-on experiences!

ADD A CUSTOM DOMAIN ➔

Here you should be able to see that when the Azure account was created, Azure Active Directory created a personal default domain that is a hash value (a number generated from a string of text) of your personal ID used as a subdomain of onmicrosoft.com. That's the domain to which you will now add a new user.

Creating a new user in the default domain

Click **USERS** and look for your single personal account. You should see in the **SOURCED FROM** column that it is a **Microsoft account**. We want to create a user in your default .onmicrosoft.com Azure Active Directory domain.

default directory

DISPLAY NAME	USER NAME	SOURCED FROM
FirstName LastName	aztrainpass[REDACTED]outlook.com	Microsoft account

We're going to follow [these instructions](#) in the next few steps, but use a specific example.

At the bottom of the page, click **+ADD USER**. In the page that appears, type the new user name, and make the **Type of User** a **New user in your organization**. In this example, the new user name is **ahmet**. Select the default domain that you discovered previously as the domain for ahmet's email address. Click the next arrow when finished.

ADD USER

Tell us about this user

TYPE OF USER

New user in your organization

USER NAME ?

ahmet @ aztrainpass[REDACTED]outlook.onmicrosoft.com
aztrainpass[REDACTED]outlook.onmicrosoft.com

2 3

This screenshot shows the 'Add User' step 2 form. It includes fields for 'Type of User' (set to 'New user in your organization'), 'User Name' (set to 'ahmet'), and an email field containing 'aztrainpass[REDACTED]outlook.onmicrosoft.com'. A large blue bar on the right side contains the numbers '2' and '3' with arrows indicating a sequence.

Add more details for Ahmet, but make sure to select the appropriate **ROLE** value. It's easy to use **Global Admin** to make sure things are working, but if you can use a lesser role, that's a good idea. This example uses the **User** role. (Find out more at [Administrator permissions by role](#).) Do not enable multi-factor authentication unless you want to use multifactor authentication for each log in operation. Click the next arrow when you're finished.

ADD USER

user profile

FIRST NAME

LAST NAME

DISPLAY NAME

ROLE ?

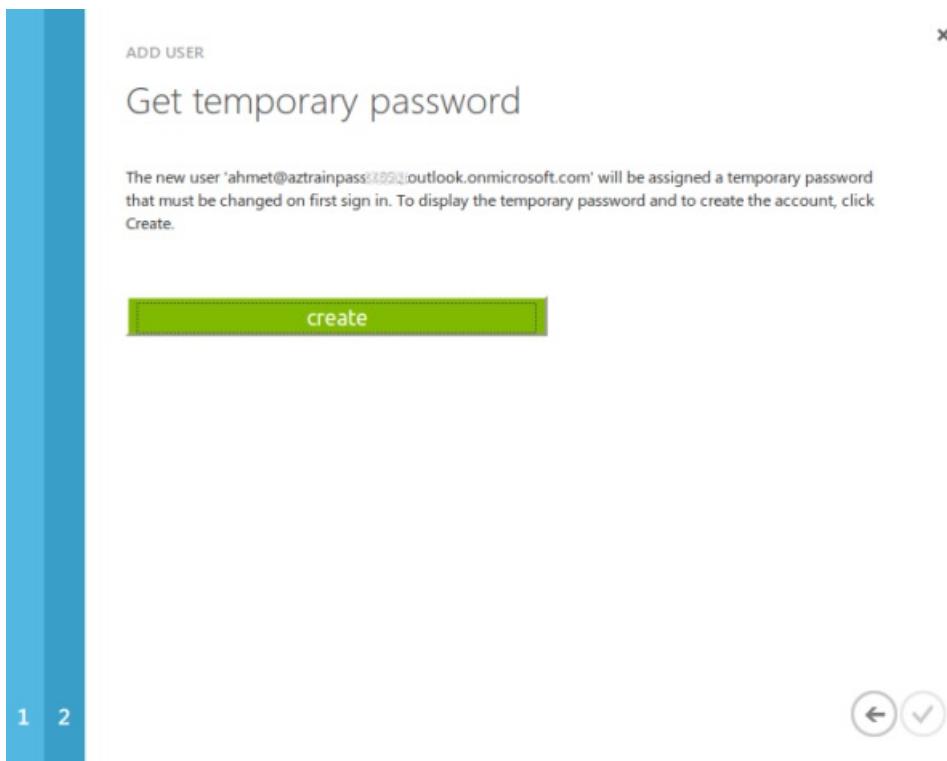
MULTI-FACTOR AUTHENTICATION ?

Enable Multi-Factor Authentication

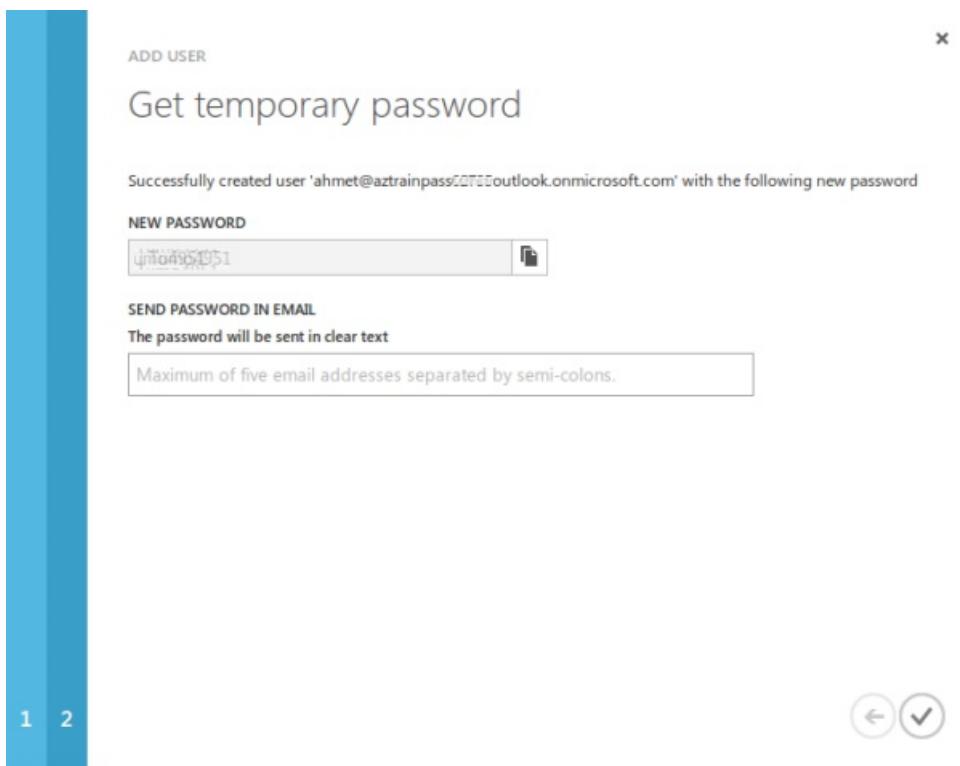
1 2 3

This screenshot shows the 'Add User' step 3 form. It includes fields for 'First Name' (Ahmet), 'Last Name' (Developer), 'Display Name' (Ahmet the Developer), 'Role' (User), and a 'Multi-Factor Authentication' checkbox. A large blue bar on the right side contains the numbers '1', '2', and '3' with arrows indicating a sequence.

Click the **create** button to generate and display a temporary password for Ahmet.



Copy the user name email address, or use **SEND PASSWORD IN EMAIL**. You'll need the information to log on shortly.

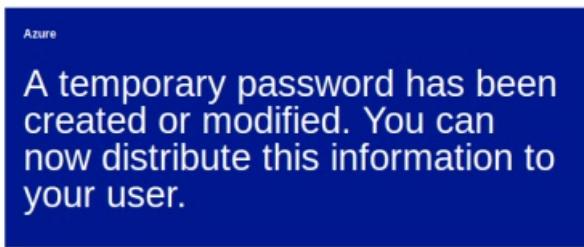


Now you should see the new user, **Ahmet the Developer**, sourced from Azure Active Directory. You've created the new work or school identity with Azure Active Directory. However, this identity does not yet have permissions to use Azure resources.

default directory

DISPLAY NAME	USER NAME	SOURCED FROM
Ahmet the Developer	ahmet@aztrainpass<REDACTED>@outlook.onmicrosoft.com	Windows Azure Active Directory
FirstName LastName	aztrainpass<REDACTED>@outlook.com	Microsoft account

If you use **SEND PASSWORD IN EMAIL**, the following kind of email is sent.



The following list contains temporary passwords for newly created or modified user accounts.

Please note:

- When distributing IDs and passwords to individual users, be sure to do so in a safe and secure manner.
- Temporary passwords are valid for 90 days.

User Name: ahmet@aztrainpass2012.outlook.onmicrosoft.com

Temporary Password: TheSecret2012

Once your end users have successfully signed in with their temporary passwords, they can create new passwords by following the steps below.

To sign in and change your temporary password:

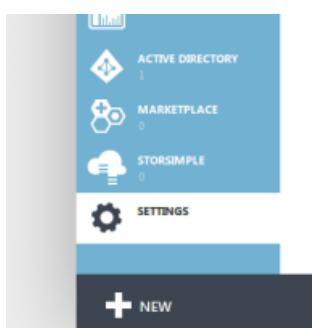
1. Go to the [sign-in page](#).
2. Enter your user name and corresponding temporary password.
3. Follow the instructions on the sign-in page to create a new password.

Thank you for choosing to host your IT solutions with Microsoft.

Thank you,
Windows Azure Active Directory Team

Adding Azure co-administrator rights for subscriptions

Now you need to add the new user as a co-administrator of your subscription so the new user can sign in to the Management Portal. To do this, in the lower-left panel click **Settings**.



In the main settings area, click **ADMINISTRATORS** at the top and you should see only your personal Microsoft account identity. At the bottom of the page, click **+ADD** to specify a co-administrator. Here, enter the email address of the new user you had created, including your default domain. As shown in the next screenshot, a green check mark appears next to the user for the default directory. Remember to select all of the subscriptions that you would like this user to be able to administer.

ADD A CO-ADMINISTRATOR

x

Specify a co-administrator for subscriptions

Co-administrators can fully manage the services within a subscription. Enter a valid email address, and then select at least one subscription.

EMAIL ADDRESS

ahmet@aztrainpass~~2025~~@outlook.onmicrosoft.com 



Default Directory

SUBSCRIPTION

SUBSCRIPTION ID



Azure Pass

fca20000-0000-0000-0000-000000000000



When you are done, you should now see two users, including your new co-administrator identity. Log out of the portal.

SUBSCRIPTIONS MANAGEMENT CERTIFICATES ADMINISTRATORS AFFINITY GROUPS USAGE REMOTEAPP

NAME	SUBSCRIPTION	SUBSCRIPTION ID	ROLE
ahmet@aztrainpass 2025 @outlook.com	Azure Pass	fca20000-0000-0000-0000-000000000000	Service administrator
ahmet@aztrainpass 2025 @outlook.onmicrosoft.com	Azure Pass	fca20000-0000-0000-0000-000000000000	Co-administrator

Logging in and changing the new user's password

Log in as the new user you created.

[Microsoft Azure](#)

Sign in with your work or school account

ahmet@aztrainpass~~2025~~@outlook.onmicrosoft.com

.....

Keep me signed in

[Sign in](#)

[Cancel](#)

[Can't access your account?](#)

You will immediately be prompted to create a new password.

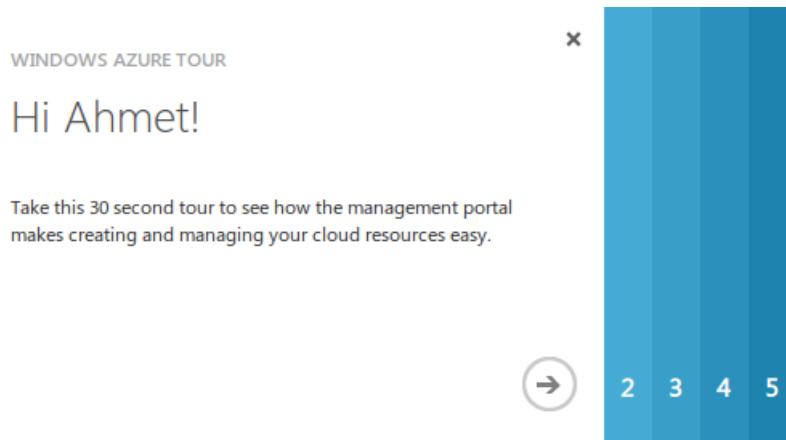
Update your password

You need to update your password because this is the first time you are signing in, or because your password has expired.

ahmet@aztrainpassXXXXoutlook.onmicrosoft.com
Current password
New password
Confirm password

Update password and sign in

You should be rewarded with success that looks like the following.



Next steps

You can now use your new Azure Active Directory identity to use [Azure resource group templates](#).

```
azure login
info: Executing command login
warn: Please note that currently you can login only via Microsoft organizational account or service principal.
For instructions on how to set them up, please read http://aka.ms/Dhf67j.
Username: ahmet@aztrainpassxxxxoutlook.onmicrosoft.com
Password: *****
/info: Added subscription Azure Pass
info: Setting subscription Azure Pass as default
+
info: login command OK
ralph@local:~$ azure config mode arm
info: New mode is arm
ralph@local:~$ azure group list
info: Executing command group list
+ Listing resource groups
info: No matched resource groups were found
info: group list command OK
ralph@local:~$ azure group create newgroup westus
info: Executing command group create
+ Getting resource group newgroup
+ Creating resource group newgroup
info: Created resource group newgroup
data: Id: /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx/resourceGroups/newgroup
data: Name: newgroup
data: Location: westus
data: Provisioning State: Succeeded
data: Tags:
data:
info: group create command OK
```

Using attributes to create advanced rules

1/17/2017 • 7 min to read • [Edit on GitHub](#)

The Azure classic portal provides you with the ability to create advanced rules to enable more complex attribute-based dynamic memberships for Azure Active Directory (Azure AD) groups.

When any attributes of a user change, the system evaluates all dynamic group rules in a directory to see if the attribute change of the user would trigger any group adds or removes. If a user satisfies a rule on a group, they are added as a member to that group. If they no longer satisfy the rule of a group they are a member of, they are removed as members from that group.

NOTE

You can set up a rule for dynamic membership on security groups or Office 365 groups. Nested group memberships aren't currently supported for group-based assignment to applications.

Dynamic memberships for groups require an Azure AD Premium license to be assigned to

- The administrator who manages the rule on a group
- All members of the group

To create the advanced rule

1. In the [Azure classic portal](#), select **Active Directory**, and then open your organization's directory.
2. Select the **Groups** tab, and then open the group you want to edit.
3. Select the **Configure** tab, select the **Advanced rule** option, and then enter the advanced rule into the text box.

Constructing the body of an advanced rule

The advanced rule that you can create for the dynamic memberships for groups is essentially a binary expression that consists of three parts and results in a true or false outcome. The three parts are:

- Left parameter
- Binary operator
- Right constant

A complete advanced rule looks similar to this: (leftParameter binaryOperator "RightConstant"), where the opening and closing parenthesis are required for the entire binary expression, double quotes are required for the right constant, and the syntax for the left parameter is user.property. An advanced rule can consist of more than one binary expressions separated by the -and, -or, and -not logical operators. The following are examples of a properly constructed advanced rule:

- (user.department -eq "Sales") -or (user.department -eq "Marketing")
- (user.department -eq "Sales") -and -not (user.jobTitle -contains "SDE")

For the complete list of supported parameters and expression rule operators, see sections below.

Note that the property must be prefixed with the correct object type: user or device. The below rule will fail the validation: mail -ne null

The correct rule would be:

```
user.mail -ne null
```

The total length of the body of your advanced rule cannot exceed 2048 characters.

NOTE

String and regex operations are case insensitive. Strings containing quotes " should be escaped using 'character, for example, user.department -eq `Sales`. Only use quotes for string type values, and only use English quotes.

Supported expression rule operators

The following table lists all the supported expression rule operators and their syntax to be used in the body of the advanced rule:

OPERATOR	SYNTAX
Not Equals	-ne
Equals	-eq
Not Starts With	-notStartsWith
Starts With	-startsWith
Not Contains	-notContains
Contains	-contains
Not Match	-notMatch
Match	-match

Operator precedence

All Operators are listed below per precedence from lower to higher, operator in same line are in equal precedence - any -all -or -and -not -eq -ne -startsWith -notStartsWith -contains -notContains -match -notMatch

All operators can be used with or without hyphen prefix.

Note that parenthesis are not always needed, you only need to add parenthesis when precedence does not meet your requirements For example:

```
user.department -eq "Marketing" -and user.country -eq "US"
```

is equivalent to:

```
(user.department -eq "Marketing") -and (user.country -eq "US")
```

Query error remediation

The following table lists potential errors and how to correct them if they occur

QUERY PARSE ERROR	ERROR USAGE	CORRECTED USAGE
Error: Attribute not supported.	(user.invalidProperty -eq "Value")	(user.department -eq "value") Property should match one from the supported properties list .
Error: Operator is not supported on attribute.	(user.accountEnabled -contains true)	(user.accountEnabled -eq true) Property is of type boolean. Use the supported operators (-eq or -ne) on boolean type from the above list.
Error: Query compilation error.	(user.department -eq "Sales") -and (user.department -eq "Marketing") (user.userPrincipalName -match "*@domain.ext")	(user.department -eq "Sales") -and (user.department -eq "Marketing") Logical operator should match one from the supported properties list above. (user.userPrincipalName -match ".*@domain.ext")or(user.userPrincipalNa me -match "@domain.ext\$")Error in regular expression.
Error: Binary expression is not in right format.	(user.department -eq "Sales") (user.department -eq "Sales") (user.department -eq "Sales")	(user.accountEnabled -eq true) -and (user.userPrincipalName -contains "alias@domain") Query has multiple errors. Parenthesis not in right place.
Error: Unknown error occurred during setting up dynamic memberships.	(user.accountEnabled -eq "True" AND user.userPrincipalName -contains "alias@domain")	(user.accountEnabled -eq true) -and (user.userPrincipalName -contains "alias@domain") Query has multiple errors. Parenthesis not in right place.

Supported properties

The following are all the user properties that you can use in your advanced rule:

Properties of type boolean

Allowed operators

- -eq
- -ne

PROPERTIES	ALLOWED VALUES	USAGE
accountEnabled	true false	user.accountEnabled -eq true)
dirSyncEnabled	true false null	(user.dirSyncEnabled -eq true)

Properties of type string

Allowed operators

- -eq
- -ne
- -notStartsWith
- -StartsWith
- -contains

- -notContains
- -match
- -notMatch

PROPERTIES	ALLOWED VALUES	USAGE
city	Any string value or \$null	(user.city -eq "value")
country	Any string value or \$null	(user.country -eq "value")
department	Any string value or \$null	(user.department -eq "value")
displayName	Any string value	(user.displayName -eq "value")
facsimileTelephoneNumber	Any string value or \$null	(user.facsimileTelephoneNumber -eq "value")
givenName	Any string value or \$null	(user.givenName -eq "value")
jobTitle	Any string value or \$null	(user.jobTitle -eq "value")
mail	Any string value or \$null (SMTP address of the user)	(user.mail -eq "value")
mailNickname	Any string value (mail alias of the user)	(user.mailNickname -eq "value")
mobile	Any string value or \$null	(user.mobile -eq "value")
objectId	GUID of the user object	(user.objectId -eq "1111111-1111-1111-1111-111111111111")
passwordPolicies	None DisableStrongPassword DisablePasswordExpiration DisablePasswordExpiration, DisableStrongPassword	(user.passwordPolicies -eq "DisableStrongPassword")
physicalDeliveryOfficeName	Any string value or \$null	(user.physicalDeliveryOfficeName -eq "value")
postalCode	Any string value or \$null	(user.postalCode -eq "value")
preferredLanguage	ISO 639-1 code	(user.preferredLanguage -eq "en-US")
sipProxyAddress	Any string value or \$null	(user.sipProxyAddress -eq "value")
state	Any string value or \$null	(user.state -eq "value")
streetAddress	Any string value or \$null	(user.streetAddress -eq "value")
surname	Any string value or \$null	(user.surname -eq "value")
telephoneNumber	Any string value or \$null	(user.telephoneNumber -eq "value")

PROPERTIES	ALLOWED VALUES	USAGE
usageLocation	Two lettered country code	(user.usageLocation -eq "US")
userPrincipalName	Any string value	(user.userPrincipalName -eq "alias@domain")
userType	member guest \$null	(user.userType -eq "Member")

Properties of type string collection

Allowed operators

- -contains
- -notContains

PROPERTIES	ALLOWED VALUES	USAGE
otherMails	Any string value	(user.otherMails -contains "alias@domain")
proxyAddresses	SMTP: alias@domain smtp: alias@domain	(user.proxyAddresses -contains "SMTP: alias@domain")

Use of Null values

To specify a null value in a rule, you can use "null" or \$null. Example:

user.mail -ne null is equivalent to user.mail -ne \$null

Extension attributes and custom attributes

Extension attributes and custom attributes are supported in dynamic membership rules.

Extension attributes are synced from on premise Window Server AD and take the format of "ExtensionAttributeX", where X equals 1 - 15. An example of a rule that uses an extension attribute would be

(user.extensionAttribute15 -eq "Marketing")

Custom Attributes are synced from on premise Windows Server AD or from a connected SaaS application and the format of "user.extension_[GUID]_[Attribute]", where [GUID] is the unique identifier in AAD for the application that created the attribute in AAD and [Attribute] is the name of the attribute as it was created. An example of a rule that uses a custom attribute is

user.extension_c272a57b722d4eb29bfe327874ae79cb__OfficeNumber

The custom attribute name can be found in the directory by querying a user's attribute using Graph Explorer and searching for the attribute name.

Support for multi-value properties

To include a multi-value property in a rule, use the "-any" operator, as in

user.assignedPlans -any assignedPlan.service -startsWith "SCO"

Direct Reports Rule

You can populate members in a group based on the manager attribute of a user.

To configure a group as a “Manager” group

1. In the Azure classic portal, click **Active Directory**, and then click the name of your organization’s directory.
2. Select the **Groups** tab, and then open the group you want to edit.
3. Select the **Configure** tab, and then select **ADVANCED RULE**.
4. Type the rule with the following syntax:

Direct Reports for *Direct Reports for {objectID_of_manager}*. An example of a valid rule for Direct Reports is

```
Direct Reports for "62e19b97-8b3d-4d4a-a106-4ce66896a863"
```

where “62e19b97-8b3d-4d4a-a106-4ce66896a863” is the objectID of the manager. The object ID can be found in the Azure AD on the **Profile tab** of the user page for the user who is the manager.

5. When saving this rule, all users that satisfy the rule will be joined as members of the group. It can take some minutes for the group to initially populate.

Using attributes to create rules for device objects

You can also create a rule that selects device objects for membership in a group. The following device attributes can be used:

PROPERTIES	ALLOWED VALUES	USAGE
displayName	any string value	(device.displayName -eq "Rob Iphone")
deviceOSType	any string value	(device.deviceOSType -eq "IOS")
deviceOSVersion	any string value	(device.OSVersion -eq "9.1")
isDirSynced	true false null	(device.isDirSynced -eq true)
isManaged	true false null	(device.isManaged -eq false)
isCompliant	true false null	(device.isCompliant -eq true)
deviceCategory	any string value	(device.deviceCategory -eq "")
deviceManufacturer	any string value	(device.deviceManufacturer -eq "Microsoft")
deviceModel	any string value	(device.deviceModel -eq "IPhone 7+")
deviceOwnership	any string value	(device.deviceOwnership -eq "")
domainName	any string value	(device.domainName -eq "contoso.com")
enrollmentProfileName	any string value	(device.enrollmentProfileName -eq "")
isRooted	true false null	(device.isRooted -eq true)

PROPERTIES	ALLOWED VALUES	USAGE
managementType	any string value	(device.managementType -eq "")
organizationalUnit	any string value	(device.organizationalUnit -eq "")
deviceId	a valid deviceId	(device.deviceId -eq "d4fe7726-5966-431c-b3b8-cddc8fdb717d")

NOTE

These device rules cannot be created using the "simple rule" dropdown in the Azure classic portal.

Additional information

These articles provide additional information on Azure Active Directory.

- [Troubleshooting dynamic memberships for groups](#)
- [Managing access to resources with Azure Active Directory groups](#)
- [Azure Active Directory cmdlets for configuring group settings](#)
- [Article Index for Application Management in Azure Active Directory](#)
- [Integrating your on-premises identities with Azure Active Directory](#)

Set up Key Vault for virtual machines in Azure Resource Manager

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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.

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](#).

Setting up WinRM access for Virtual Machines in Azure Resource Manager

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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 most 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
```

First look: Protect Azure VMs with a recovery services vault

1/17/2017 • 13 min to read • [Edit on GitHub](#)

This tutorial takes you through the steps for creating a recovery services vault and backing up an Azure virtual machine (VM). Recovery services vaults protect:

- Azure Resource Manager-deployed VMs
- Classic VMs
- Standard storage VMs
- Premium storage VMs
- VMs encrypted using Azure Disk Encryption, with BEK and KEK

For more information on protecting Premium storage VMs, see [Back up and Restore Premium Storage VMs](#)

NOTE

This tutorial assumes you already have a VM in your Azure subscription and that you have taken measures to allow the backup service to access the VM.

The Azure Backup service has two types of vaults - the Backup vault and the Recovery Services vault. The Backup vault came first. Then the Recovery Services vault came along to support the expanded Resource Manager deployments. Microsoft recommends using Resource Manager deployments unless you specifically require a Classic deployment.

DEPLOYMENT	PORTAL	VAULT
Classic	Classic	Backup
Resource Manager	Azure	Recovery Services

NOTE

Backup vaults cannot protect Resource Manager-deployed solutions. However, you can use a Recovery Services vault to protect classically-deployed servers and VMs.

Based on no of VMs you want to protect, you can start from different start points - if you want to backup multiple virtual machines in one single operation, go to Recovery Services vault and start backup from vault dashboard. If you have a single VM, which you want to backup, you can backup directly from VM management blade.

Configure Backup from VM management blade

1. Sign in to the [Azure portal](#).
2. On the Hub menu, click **More Services** and in the list of resources, type **Virtual machines**. The list of virtual machines appears. From the list of virtual machines, select a virtual machine, which you want to backup. This opens virtual machine management blade.

ContosoVM

Virtual machine

Connect Start Restart Stop Delete

Search (Ctrl+ /)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

SETTINGS

Availability set

Disks

Extensions

Network interfaces

Size

Backup

Essentials

Resource group: ContosoRG

Status: Running

Location: East US

Subscription name: MAB Canary Subscription 3

Computer name: ContosoVM

Operating system: Windows

Size: Standard DS1 v2 (1 core, 3.5 GB memory)

Public IP address/DNS name label: 23.96.43.234/<none>

Virtual network/subnet: ContosoRG-vnet/default

Monitoring

CPU percentage

100%

80%

60%

40%

20%

0%

3. In the VM management blade, click "Backup" option present on the left-hand side under Settings.

ContosoVM

Virtual machine

Connect Start Restart Stop Delete

Search (Ctrl+ /)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

SETTINGS

Availability set

Disks

Extensions

Network interfaces

Size

Backup

Properties

Locks

Essentials

Resource group: ContosoRG

Status: Running

Location: East US

Subscription name: [REDACTED]

Computer name: ContosoVM

Operating system: Windows

Size: Standard DS1 v2 (1 core, 3.5 GB memory)

Public IP address/DNS name label: [REDACTED]

Virtual network/subnet: ContosoRG-vnet/default

Monitoring

CPU percentage

100%

80%

60%

40%

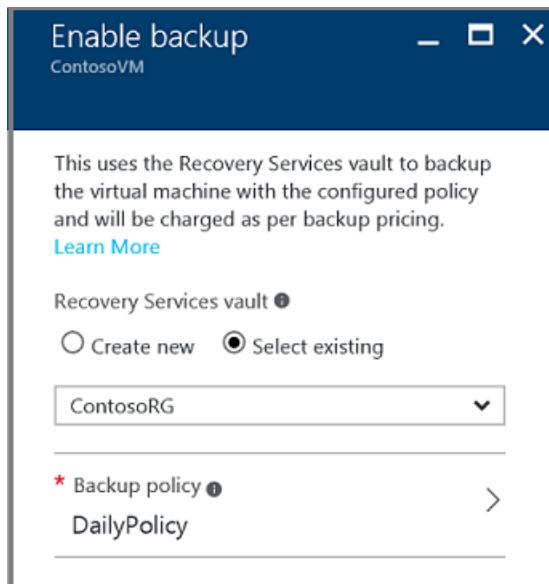
20%

0%

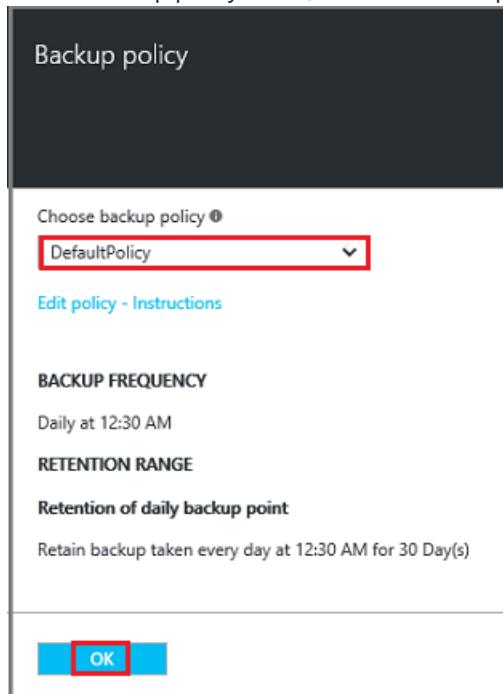
4:45 PM 5 PM 5:15 PM 5:30 PM

PERCENTAGE CPU

4. This opens Enable Backup blade. This blade expects two inputs: Recovery Services vault - an Azure Backup resource, which is used to store backups of the VMs; A backup Policy - Backup policy specifies schedule of the backups and how long to retain backup copies. This blade comes with default options. You can customize them as per backup requirements.

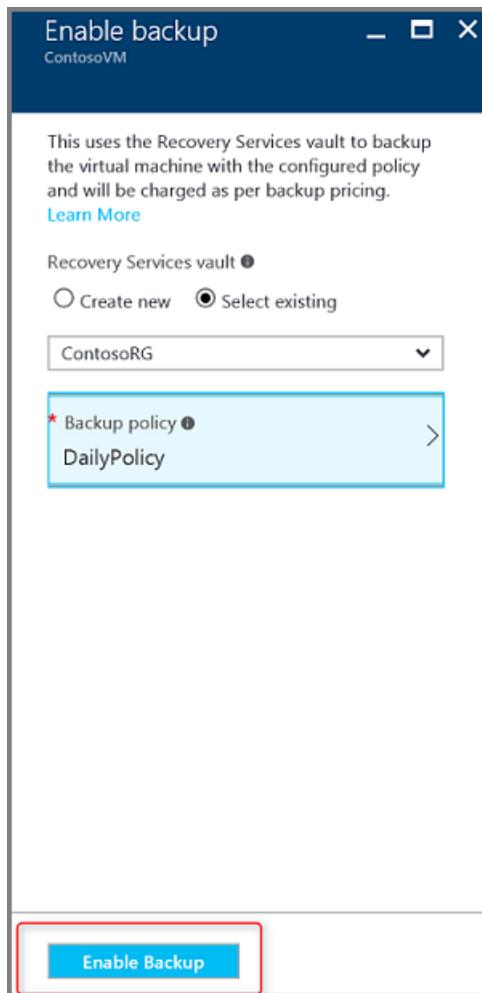


5. For Recovery Services vault, you can select an existing vault or create a new Vault. If you are creating a new vault, it gets created in the same Resource Group as virtual machine and location is same as virtual machine. If you want to create a Recovery Services vault with different values, [create a recovery services vault](#) before clicking Backup option in Step#3 and select that in this blade.
6. On the Backup policy blade, select the backup policy you want to apply to the vault and click **OK**.

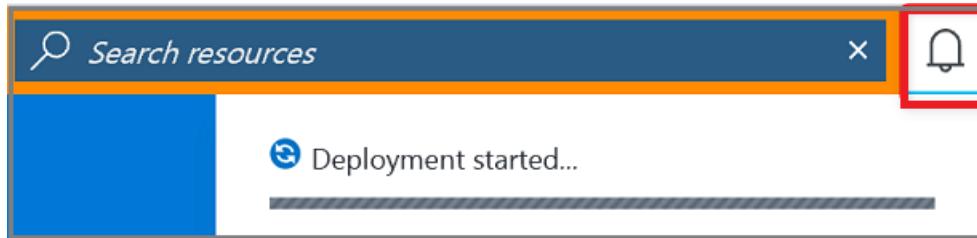


The details of the default policy are listed in the details. If you want to create a policy, select **Create New** from the drop-down menu. The drop-down menu also provides an option to switch the time when the snapshot is taken. For instructions on defining a backup policy, see [Defining a backup policy](#). Once you click **OK**, the backup policy is associated with the virtual machine.

7. Click "Enable Backup" to configure Backup on the virtual machine. This will trigger a deployment.



8. You can track the progress of configuration through notifications.



9. Once deployment for Configure backup is completed, clicking on "backup" option on VM management blade takes you to Backup Item blade corresponding to backed up VM.

The screenshot shows the Azure portal interface for a virtual machine named 'ContosoVM'. The left sidebar has a search bar at the top, followed by a list of navigation items: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, SETTINGS (Availability set, Disks, Extensions, Network interfaces, Size), and Backup. The 'Backup' item is highlighted with a red box. Below the settings is a 'Properties' link and a 'Locks' link. The main content area is titled 'Essentials' and shows details about a recovery services vault named 'ContosoRG'. It includes fields for Subscription name (redacted), Subscription ID (redacted), Item type (Azure virtual machine), and Backup policy (DailyPolicy). A 'Restore points' section shows that there are 0 restore points for both the last 30 days and the last 7 days.

Configure Backup from Recovery Services vault View

At a high level, here are the steps that you'll complete.

1. Create a recovery services vault for a VM.
2. Use the Azure portal to select a Scenario, set Policy, and identify items to protect.
3. Run the initial backup.

Create a recovery services vault for a VM

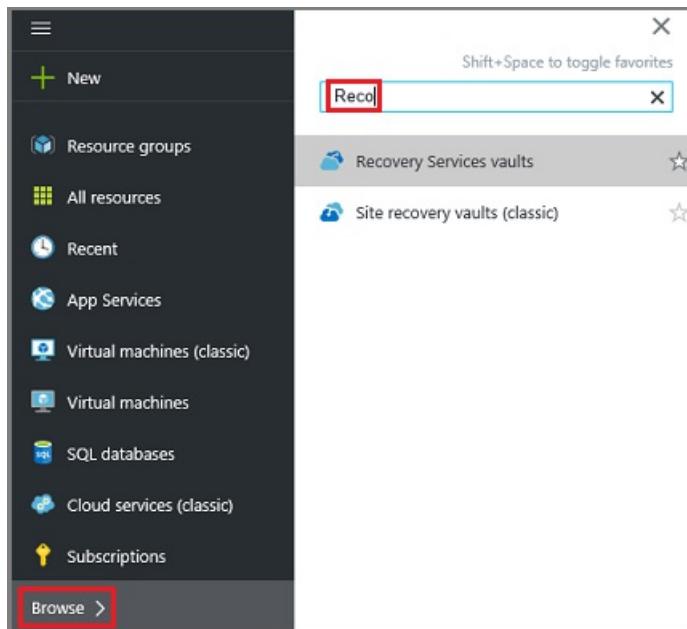
A recovery services vault is an entity that stores all the backups and recovery points that have been created over time. The recovery services vault also contains the backup policy applied to the protected VMs.

NOTE

Backing up VMs is a local process. You cannot back up VMs from one location to a recovery services vault in another location. So, for every Azure location that has VMs to be backed up, at least one recovery services vault must exist in that location.

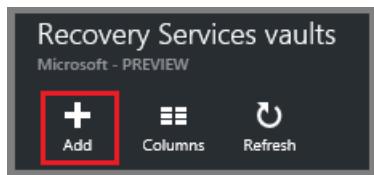
To create a recovery services vault:

1. Sign in to the [Azure portal](#).
2. On the Hub menu, click **Browse** and in the list of resources, type **Recovery Services**. As you begin typing, the list filters based on your input. Click **Recovery Services vault**.

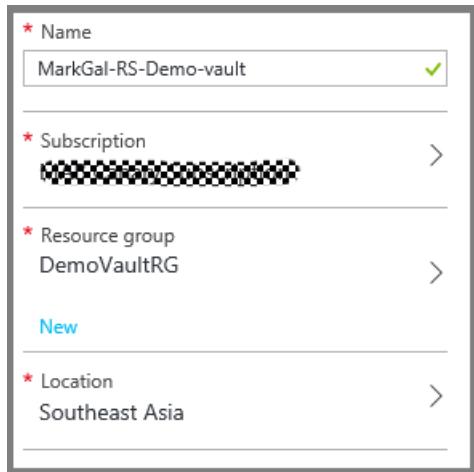


The list of recovery services vaults are displayed.

3. On the **Recovery Services vaults** menu, click **Add**.



The Recovery Services vault blade opens, prompting you to provide a **Name**, **Subscription**, **Resource group**, and **Location**.



4. For **Name**, enter a friendly name to identify the vault. The name needs to be unique for the Azure subscription. Type a name that contains between 2 and 50 characters. It must start with a letter, and can contain only letters, numbers, and hyphens.
5. Click **Subscription** to see the available list of subscriptions. If you are not sure which subscription to use, use the default (or suggested) subscription. There are multiple choices only if your organizational account is associated with multiple Azure subscriptions.
6. Click **Resource group** to see the available list of Resource groups, or click **New** to create a Resource group. For complete information on Resource groups, see [Azure Resource Manager overview](#)
7. Click **Location** to select the geographic region for the vault. The vault **must** be in the same region as the virtual machines that you want to protect.

IMPORTANT

If you are unsure of the location in which your VM exists, close out of the vault creation dialog, and go to the list of Virtual Machines in the portal. If you have virtual machines in multiple regions, create a recovery services vault in each region. Create the vault in the first location before going to the next location. There is no need to specify storage accounts to store the backup data--the recovery services vault and the Azure Backup service handle this automatically.

- Click **Create**. It can take a while for the recovery services vault to be created. Monitor the status notifications in the upper right-hand area in the portal. Once your vault is created, it appears in the list of recovery services vaults.

NAME	...
Contoso-vault	...
Jim-RS-Demo-vault	...
Jim-RS-Demo-vault2	...
LSU-RS-vault	...
test4PSvault	...

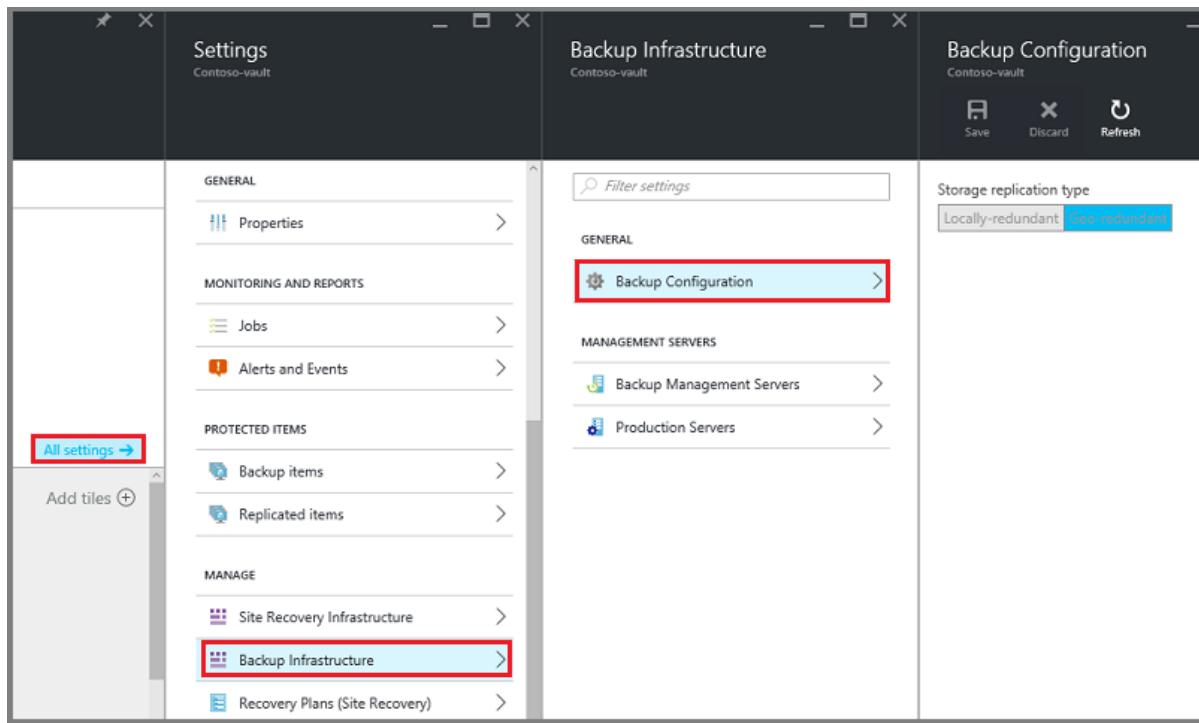
Now that you've created your vault, learn how to set the storage replication.

Set Storage Replication

The storage replication option allows you to choose between geo-redundant storage and locally redundant storage. By default, your vault has geo-redundant storage. Leave the option set to geo-redundant storage if this is your primary backup. Choose locally redundant storage if you want a cheaper option that isn't as durable. Read more about [geo-redundant](#) and [locally redundant](#) storage options in the [Azure Storage replication overview](#).

To edit the storage replication setting:

- Select your vault to open the vault dashboard and the Settings blade. If the **Settings** blade doesn't open, click **All settings** in the vault dashboard.
- On the **Settings** blade, click **Backup Infrastructure > Backup Configuration** to open the **Backup Configuration** blade. On the **Backup Configuration** blade, choose the storage replication option for your vault.

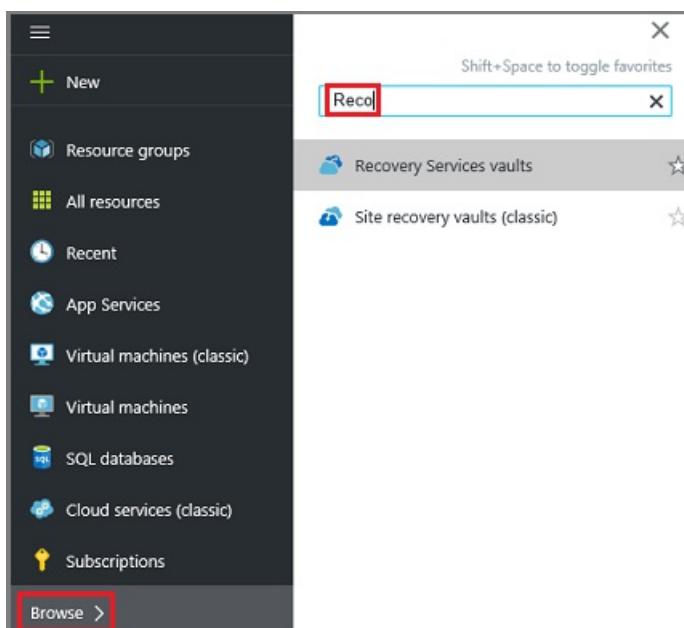


After choosing the storage option for your vault, you are ready to associate the VM with the vault. To begin the association, you should discover and register the Azure virtual machines.

Select a backup goal, set policy and define items to protect

Before registering a VM with a vault, run the discovery process to ensure that any new virtual machines that have been added to the subscription are identified. The process queries Azure for the list of virtual machines in the subscription, along with additional information like the cloud service name and the region. In the Azure portal, scenario refers to what you are going to put into the recovery services vault. Policy is the schedule for how often and when recovery points are taken. Policy also includes the retention range for the recovery points.

1. If you already have a recovery services vault open, proceed to step 2. If you do not have a recovery services vault open, but are in the Azure portal, on the Hub menu, click **Browse**.
 - In the list of resources, type **Recovery Services**.
 - As you begin typing, the list filters based on your input. When you see **Recovery Services vaults**, click it.



The list of recovery services vaults appears.

- From the list of recovery services vaults, select a vault.

The selected vault dashboard opens.

DemoVault
Recovery Services vault - PREVIEW

Settings + + Delete

Essentials ^

Resource group DemoVaultRG	Backup items/Azure VM Backup 0
Status Active	Backup management servers 0
Location Southeast Asia	Replicated items 0
Subscription name [REDACTED]	Subscription ID [REDACTED]

All settings →

Monitoring Add tiles +

Site Recovery Health	
Unhealthy serv...	0
Events	0

Backup Add tiles +

Backup Items	Backup Jobs	Backup Usage
Azure Virtual Machines	Azure virtual mach... 0	Cloud - LRS 0 B
File-Folders		Cloud - GRS 0 B

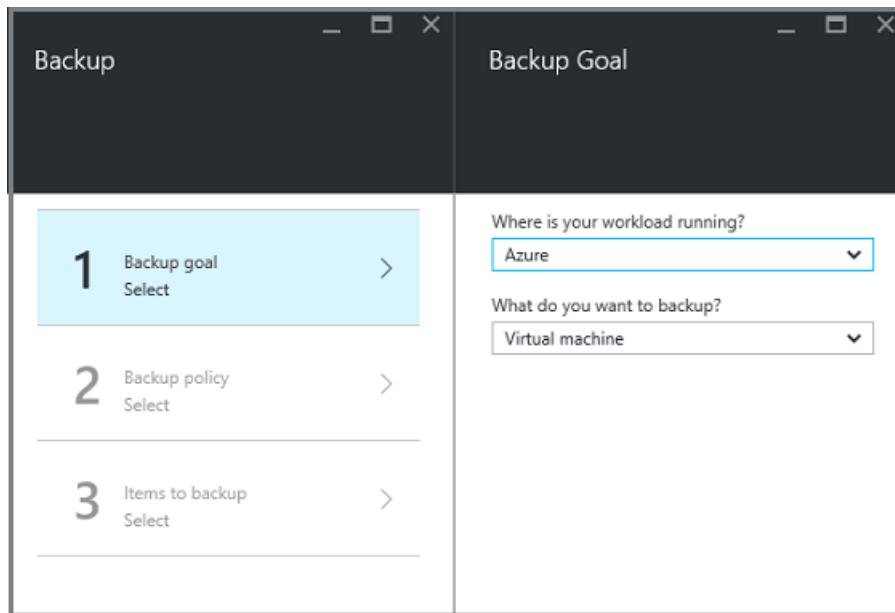
- From the vault dashboard menu, click **Backup** to open the Backup blade.



When the blade opens, the Backup service searches for any new VMs in the subscription.

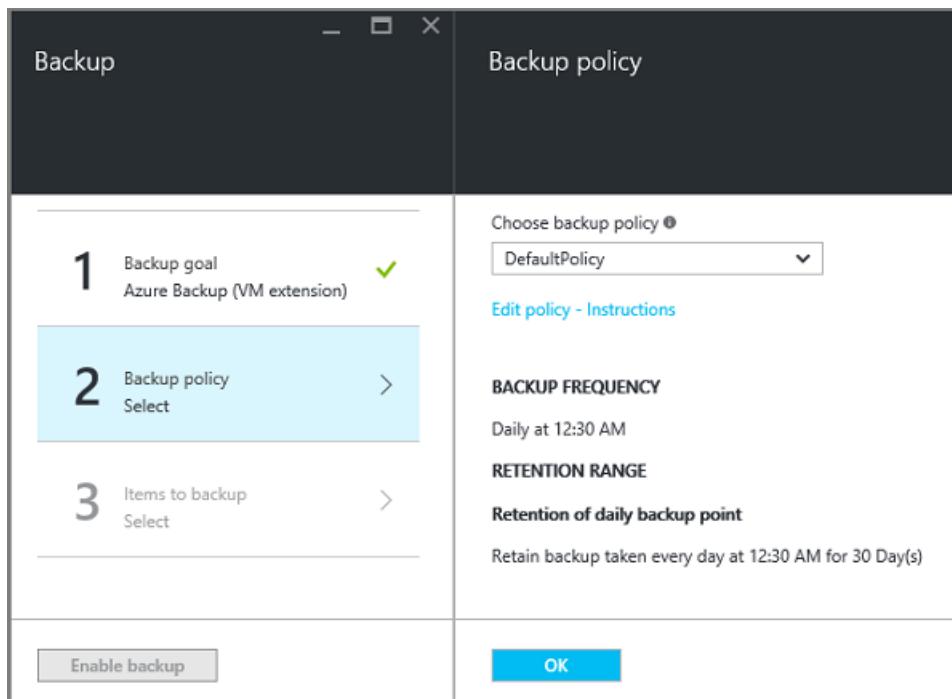


- On the Backup blade, click **Backup goal** to open the Backup Goal blade.

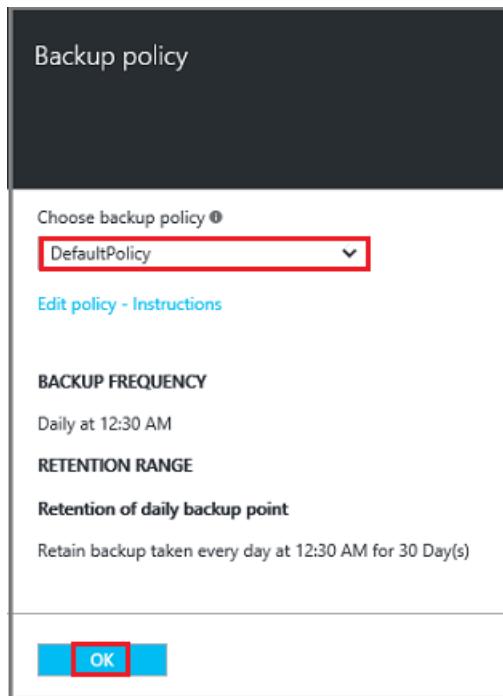


4. On the Backup Goal blade, set **Where is your workload running** to Azure and **What do you want to backup** to Virtual machine, then click **OK**.

The Backup Goal blade closes and the Backup policy blade opens.



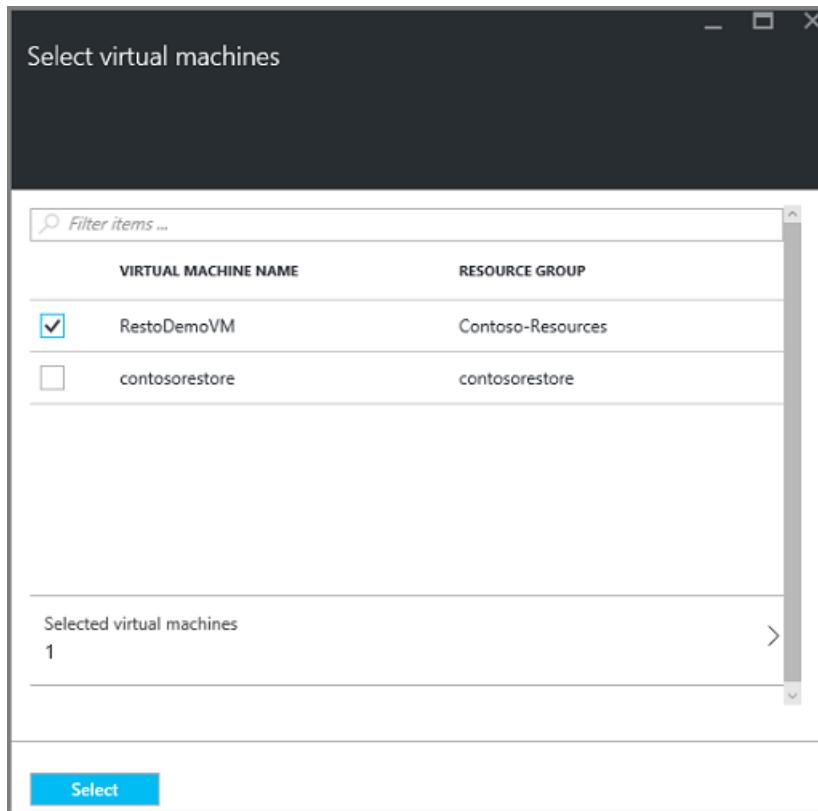
5. On the Backup policy blade, select the backup policy you want to apply to the vault and click **OK**.



The details of the default policy are listed in the details. If you want to create a policy, select **Create New** from the drop-down menu. The drop-down menu also provides an option to switch the time when the snapshot is taken, to 7PM. For instructions on defining a backup policy, see [Defining a backup policy](#). Once you click **OK**, the backup policy is associated with the vault.

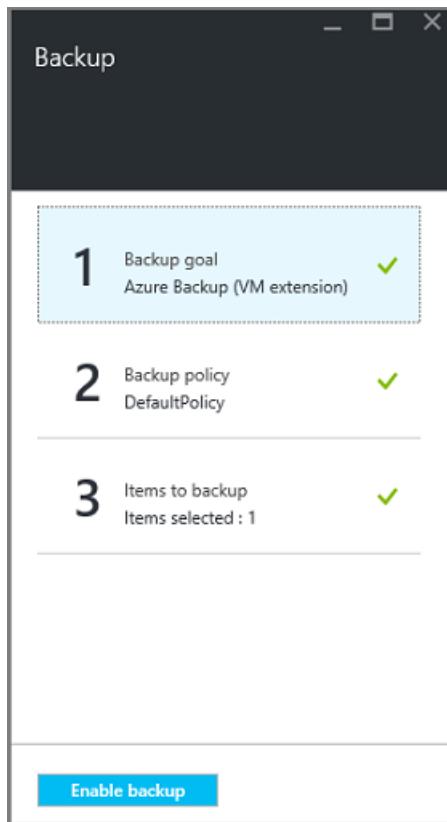
Next choose the VMs to associate with the vault.

6. Choose the virtual machines to associate with the specified policy and click **Select**.



If you do not see the desired VM, check that it exists in the same Azure location as the Recovery Services vault.

7. Now that you have defined all settings for the vault, in the Backup blade click **Enable Backup** at the bottom of the page. This deploys the policy to the vault and the VMs.



Initial backup

Once a backup policy has been deployed on the virtual machine, that does not mean the data has been backed up. By default, the first scheduled backup (as defined in the backup policy) is the initial backup. Until the initial backup occurs, the Last Backup Status on the **Backup Jobs** blade shows as **Warning(initial backup pending)**.

NAME	ITEM TYPE	LAST BACKUP STATUS	BACKUP POLICY	LATEST RECOVERY POI...
cpubdemovm-arm	Azure Virtual Machines	⚠ Warning(Initial backu...)	DefaultPolicy	

Unless your initial backup is due to begin soon, it is recommended that you run **Back up Now**.

To run **Back up Now**:

1. On the vault dashboard, on the **Backup** tile, click **Azure Virtual Machines**

The screenshot shows the Azure Recovery Services vault dashboard for 'MarkGal-RS-Demo-vault'. The 'Backup Items' section is highlighted, showing categories for Azure Virtual Machines, File-Folders, Backup Jobs, and Backup Usage. A context menu for 'Azure Virtual Machines' is open, with the 'Backup now' option highlighted.

The **Backup Items** blade opens.

- On the **Backup Items** blade, right-click the vault you want to back up, and click **Backup now**.

The screenshot shows the 'Backup Items' blade for 'cpudemovm-arm'. A context menu is open over the 'Azure Virtual Machines' item, with the 'Backup now' option highlighted. The menu also includes options for Pin to dashboard, Restore, Stop backup, and Delete backup data.

The Backup job is triggered.

The screenshot shows a notification message: 'Backup triggered(cpudemovm-arm) 1:01 PM'.

- To view that your initial backup has completed, on the vault dashboard, on the **Backup Jobs** tile, click **Azure virtual machines**.

The screenshot shows the Azure Recovery Services vault interface. At the top, there are four main navigation buttons: Settings, Backup, Replicate, and Delete. Below this, the 'Essentials' section provides basic information about the resource group, status, location, and subscription. The 'Backup' section contains three tiles: 'Backup Items', 'Backup Jobs', and 'Backup Usage'. The 'Backup Items' tile has two items listed: 'Azure Virtual Machines' and 'File-Folders'. The 'Backup Jobs' tile shows a list of backup jobs for 'Azure virtual mach...', with one job highlighted by a red box. The 'Backup Usage' tile displays storage details for Cloud - LRS and Cloud - GRS.

The Backup Jobs blade opens.

4. In the Backup jobs blade, you can see the status of all jobs.

The screenshot shows the 'Backup jobs' blade. It lists three backup operations for the workload 'cpubdemovm-arm':

- Backup (Status: In progress)
- Backup (Status: Completed)
- Configure backup (Status: Completed)

Each row includes columns for Workload Name, Operation, Status, Type, Start Time, Duration, and a 'More' (ellipsis) button.

NOTE

As a part of the backup operation, the Azure Backup service issues a command to the backup extension in each VM to flush all writes and take a consistent snapshot.

When the backup job is finished, the status is *Completed*.

Defining a backup policy

A backup policy defines a matrix of when the data snapshots are taken, and how long those snapshots are retained. When defining a policy for backing up a VM, you can trigger a backup job *once a day*. When you create a new policy, it is applied to the vault. The backup policy interface looks like this:

* Policy name

Backup frequency

Daily 5:30 AM Local Time (UTC-07:00)

Retention range

Retention of daily backup point.

* At For Day(s)
5:30 AM 180

Retention of weekly backup point.

* On * At For Week(s)
Sunday 5:30 AM 104

Retention of monthly backup point.

Week Based **Day Based**

* On * Day * At For Month(s)
First Sunday 5:30 AM 60

Retention of yearly backup point.

Week Based **Day Based**

* In * On * Day * At For Year(s)
January First Sunday 5:30 AM 10

To create a policy:

1. Enter a name for the **Policy name**.
2. Snapshots of your data can be taken at Daily or Weekly intervals. Use the **Backup Frequency** drop-down menu to choose whether data snapshots are taken Daily or Weekly.
 - If you choose a Daily interval, use the highlighted control to select the time of the day for the snapshot. To change the hour, de-select the hour, and select the new hour.

BACKUP FREQUENCY

Daily 5:30 AM Local Time (UTC-07:00)

- If you choose a Weekly interval, use the highlighted controls to select the day(s) of the week, and the time of day to take the snapshot. In the day menu, select one or multiple days. In the hour menu, select one hour. To change the hour, de-select the selected hour, and select the new hour.

BACKUP FREQUENCY

Weekly Sunday 5:30 AM Local Time (UTC-07:00)

3. By default, all **Retention Range** options are selected. Uncheck any retention range limit you do not want to use. Then, specify the interval(s) to use.

Monthly and Yearly retention ranges allow you to specify the snapshots based on a weekly or daily increment.

NOTE

When protecting a VM, a backup job runs once a day. The time when the backup runs is the same for each retention range.

- After setting all options for the policy, at the top of the blade click **Save**.

The new policy is immediately applied to the vault.

Install the VM Agent on the virtual machine

This information is provided in case it is needed. The Azure VM Agent must be installed on the Azure virtual machine for the Backup extension to work. However, if your VM was created from the Azure gallery, then the VM Agent is already present on the virtual machine. VMs that are migrated from on-premises datacenters would not have the VM Agent installed. In such a case, the VM Agent needs to be installed. If you have problems backing up the Azure VM, check that the Azure VM Agent is correctly installed on the virtual machine (see the table below). If you create a custom VM, [ensure the Install the VM Agent check box is selected](#) before the virtual machine is provisioned.

Learn about the [VM Agent](#) and [how to install it](#).

The following table provides additional information about the VM Agent for Windows and Linux VMs.

OPERATION	WINDOWS	LINUX
Installing the VM Agent	<ul style="list-style-type: none">Download and install the agent MSI. You need Administrator privileges to complete the installation.Update the VM property to indicate that the agent is installed.	<ul style="list-style-type: none">Install the latest Linux agent from GitHub. You need Administrator privileges to complete the installation.Update the VM property to indicate that the agent is installed.
Updating the VM Agent	Updating the VM Agent is as simple as reinstalling the VM Agent binaries . Ensure that no backup operation is running while the VM agent is being updated.	Follow the instructions on updating the Linux VM Agent . Ensure that no backup operation is running while the VM Agent is being updated.
Validating the VM Agent installation	<ul style="list-style-type: none">Navigate to the <code>C:\WindowsAzure\Packages</code> folder in the Azure VM.You should find the <code>WaAppAgent.exe</code> file present.Right-click the file, go to Properties, and then select the Details tab. The Product Version field should be 2.6.1198.718 or higher.	N/A

Backup extension

Once the VM Agent is installed on the virtual machine, the Azure Backup service installs the backup extension to the VM Agent. The Azure Backup service seamlessly upgrades and patches the backup extension without additional user intervention.

The backup extension is installed by the Backup service whether the VM is running. A running VM provides the greatest chance of getting an application-consistent recovery point. However, the Azure Backup service continues to back up the VM even if it is turned off, and the extension could not be installed. This is known as Offline VM. In this case, the recovery point will be *crash consistent*.

Troubleshooting information

If you have issues accomplishing some of the tasks in this article, consult the [Troubleshooting guidance](#).

Pricing

Azure VM backup will be charged based on Protected Instances model. Learn more on [Backup Pricing](#)

Questions?

If you have questions, or if there is any feature that you would like to see included, [send us feedback](#).

Deploy and manage backups for Resource Manager-deployed VMs using PowerShell

1/17/2017 • 12 min to read • [Edit on GitHub](#)

This article shows you how to use Azure PowerShell cmdlets to back up and recover an Azure virtual machine (VM) from a Recovery Services vault. A Recovery Services vault is an Azure Resource Manager resource and is used to protect data and assets in both Azure Backup and Azure Site Recovery services. You can use a Recovery Services vault to protect Azure Service Manager-deployed VMs, as well as Azure Resource Manager-deployed VMs.

NOTE

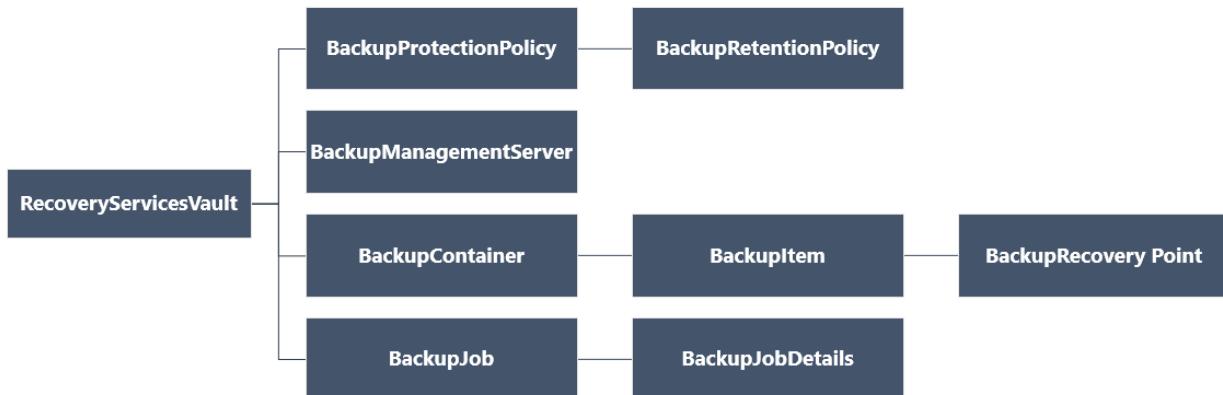
Azure has two deployment models for creating and working with resources: [Resource Manager](#) and [Classic](#). This article is for use with VMs created using the Resource Manager model.

This article walks you through using PowerShell to protect a VM, and restore data from a recovery point.

Concepts

If you are not familiar with the Azure Backup service, for an overview of the service, check out [What is Azure Backup?](#) Before you start, ensure that you cover the essentials about the prerequisites needed to work with Azure Backup, and the limitations of the current VM backup solution.

In order to use PowerShell effectively, it is necessary to understand the hierarchy of objects and from where to start.



To view the `AzureRmRecoveryServicesBackup` PowerShell cmdlet reference, see the [Azure Backup - Recovery Services Cmdlets](#) in the Azure library. To view the `AzureRmRecoveryServicesVault` PowerShell cmdlet reference, see the [Azure Recovery Service Cmdlets](#).

Setup and Registration

To begin:

1. [Download the latest version of PowerShell](#) (the minimum version required is : 1.4.0)
2. Find the Azure Backup PowerShell cmdlets available by typing the following command:

```
PS C:\> Get-Command *azurermrecoveryservices*
```

CommandType	Name	Version	Source
Cmdlet	Backup-AzureRmRecoveryServicesBackupItem	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Disable-AzureRmRecoveryServicesBackupProtection	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Enable-AzureRmRecoveryServicesBackupProtection	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupContainer	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupItem	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupJob	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupJobDetails	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupManagementServer	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupProperties	1.4.0	AzureRM.RecoveryServices
Cmdlet	Get-AzureRmRecoveryServicesBackupProtectionPolicy	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupRecoveryPoint	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupRetentionPolic...	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesBackupSchedulePolicy...	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Get-AzureRmRecoveryServicesVault	1.4.0	AzureRM.RecoveryServices
Cmdlet	Get-AzureRmRecoveryServicesVaultSettingsFile	1.4.0	AzureRM.RecoveryServices
Cmdlet	New-AzureRmRecoveryServicesBackupProtectionPolicy	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	New-AzureRmRecoveryServicesVault	1.4.0	AzureRM.RecoveryServices
Cmdlet	Remove-AzureRmRecoveryServicesProtectionPolicy	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Remove-AzureRmRecoveryServicesVault	1.4.0	AzureRM.RecoveryServices
Cmdlet	Restore-AzureRMRecoveryServicesBackupItem	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Set-AzureRmRecoveryServicesBackupProperties	1.4.0	AzureRM.RecoveryServices
Cmdlet	Set-AzureRmRecoveryServicesBackupProtectionPolicy	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Set-AzureRmRecoveryServicesVaultContext	1.4.0	AzureRM.RecoveryServices
Cmdlet	Stop-AzureRmRecoveryServicesBackupJob	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Unregister-AzureRmRecoveryServicesBackupContainer	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Unregister-AzureRmRecoveryServicesBackupManagem...	1.4.0	AzureRM.RecoveryServices.Backup
Cmdlet	Wait-AzureRmRecoveryServicesBackupJob	1.4.0	AzureRM.RecoveryServices.Backup

The following tasks can be automated with PowerShell:

- Create a Recovery Services vault
- Backup or protect Azure VMs
- Trigger a backup job
- Monitor a backup job
- Restore an Azure VM

Create a recovery services vault

The following steps lead you through creating a Recovery Services vault. A Recovery Services vault is different than a Backup vault.

1. If you are using Azure Backup for the first time, you must use the [Register-AzureRMResourceProvider](#) cmdlet to register the Azure Recovery Service provider with your subscription.

```
PS C:\> Register-AzureRmResourceProvider -ProviderNamespace "Microsoft.RecoveryServices"
```

2. The Recovery Services vault is an Resource Manager resource, so you need to place it within a resource group. You can use an existing resource group, or create a new resource group with the [New-AzureRmResourceGroup](#) cmdlet. When creating a new resource group, specify the name and location for the resource group.

```
PS C:\> New-AzureRmResourceGroup -Name "test-rg" -Location "West US"
```

3. Use the [New-AzureRmRecoveryServicesVault](#) cmdlet to create the new vault. Be sure to specify the same location for the vault as was used for the resource group.

```
PS C:\> New-AzureRmRecoveryServicesVault -Name "testvault" -ResourceGroupName "test-rg" -Location "West US"
```

4. Specify the type of storage redundancy to use; you can use [Locally Redundant Storage \(LRS\)](#) or [Geo Redundant Storage \(GRS\)](#). The following example shows the -BackupStorageRedundancy option for testVault is set to GeoRedundant.

```
PS C:\> $vault1 = Get-AzureRmRecoveryServicesVault -Name "testVault"
PS C:\> Set-AzureRmRecoveryServicesBackupProperties -Vault $vault1 -BackupStorageRedundancy GeoRedundant
```

TIP

Many Azure Backup cmdlets require the Recovery Services vault object as an input. For this reason, it is convenient to store the Backup Recovery Services vault object in a variable.

View the vaults in a subscription

Use [Get-AzureRmRecoveryServicesVault](#) to view the list of all vaults in the current subscription. You can use this command to check that a new vault was created, or to see what vaults are available in the subscription.

Run the command, Get-AzureRmRecoveryServicesVault, and all vaults in the subscription are listed.

```
PS C:\> Get-AzureRmRecoveryServicesVault
Name          : Contoso-vault
ID            : /subscriptions/1234
Type          : Microsoft.RecoveryServices/vaults
Location       : WestUS
ResourceGroupName : Contoso-docs-rg
SubscriptionId   : 1234-567f-8910-abc
Properties      : Microsoft.Azure.Commands.RecoveryServices.ARSVaultProperties
```

Backup Azure VMs

Now that you have created a recovery services vault, you can use it to protect a virtual machine. However before you apply the protection, you must set the vault context and you will want to verify the protection policy. Vault context defines the type of data that is protected in the vault. The protection policy is the schedule for when the backup job is run, and how long each backup snapshot is retained.

Before enabling protection on a VM, you must set the vault context. The context is applied to all subsequent cmdlets.

```
PS C:\> Get-AzureRmRecoveryServicesVault -Name testvault | Set-AzureRmRecoveryServicesVaultContext
```

Create a protection policy

When you create a new vault, it comes with a default policy. This policy triggers a backup job each day at a specified time. Per the default policy, the backup snapshot is retained for 30 days. You can use the default policy to quickly protect your VM and edit the policy later with different details.

Use [Get-AzureRmRecoveryServicesBackupProtectionPolicy](#) to view the available list of policies in the vault :

```
PS C:\> Get-AzureRmRecoveryServicesBackupProtectionPolicy -WorkloadType AzureVM
Name          WorkloadType    BackupManagementType BackupTime           DaysOfWeek
-----        -----          -----          -----
DefaultPolicy  AzureVM        AzureVM        4/14/2016 5:00:00 PM
```

NOTE

The timezone of the BackupTime field in PowerShell is UTC. However, when the backup time is shown in the Azure portal, the time is adjusted to your local timezone.

A backup protection policy is associated with at least one retention policy. Retention policy defines how long a recovery point is kept with Azure Backup. Use **Get-AzureRmRecoveryServicesBackupRetentionPolicyObject** to view the default retention policy. Similarly you can use **Get-AzureRmRecoveryServicesBackupSchedulePolicyObject** to obtain the default schedule policy. The schedule and retention policy objects are used as inputs to the **New-AzureRmRecoveryServicesBackupProtectionPolicy** cmdlet.

A backup protection policy defines when and how often the backup of an item is done. The **New-AzureRmRecoveryServicesBackupProtectionPolicy** cmdlet creates a PowerShell object that holds backup policy information. The backup policy is used as an input to the **Enable-AzureRmRecoveryServicesBackupProtection** cmdlet.

```
PS C:\> $schPol = Get-AzureRmRecoveryServicesBackupSchedulePolicyObject -WorkloadType "AzureVM"
PS C:\> $retPol = Get-AzureRmRecoveryServicesBackupRetentionPolicyObject -WorkloadType "AzureVM"
PS C:\> New-AzureRmRecoveryServicesBackupProtectionPolicy -Name "NewPolicy" -WorkloadType AzureVM -RetentionPolicy $retPol -SchedulePolicy $schPol
Name           WorkloadType      BackupManagementType BackupTime          DaysOfWeek
----           -----          -----                -----          -----
NewPolicy      AzureVM          AzureVM            4/24/2016 1:30:00 AM
```

Enable protection

Enabling protection involves two objects - the item and the policy. Both objects are required to enable protection on the vault. Once the policy has been associated with the vault, the backup workflow is triggered at the time defined in the policy schedule.

To enable the protection on non-encrypted ARM VMs

```
PS C:\> $pol=Get-AzureRmRecoveryServicesBackupProtectionPolicy -Name "NewPolicy"
PS C:\> Enable-AzureRmRecoveryServicesBackupProtection -Policy $pol -Name "V2VM" -ResourceGroupName "RGName1"
```

To enable the protection on encrypted VMs[encrypted using BEK and KEK], you need to give permissions for Azure Backup service to read keys and secrets from key vault.

```
PS C:\> Set-AzureRmKeyVaultAccessPolicy -VaultName 'KeyVaultName' -ResourceGroupName 'RGNameOfKeyVault' -PermissionsToKeys backup,get,list -PermissionsToSecrets get,list -ServicePrincipalName 262044b1-e2ce-469f-a196-69ab7ada62d3
PS C:\> $pol=Get-AzureRmRecoveryServicesBackupProtectionPolicy -Name "NewPolicy"
PS C:\> Enable-AzureRmRecoveryServicesBackupProtection -Policy $pol -Name "V2VM" -ResourceGroupName "RGName1"
```

NOTE

If you are on Azure Government cloud, then use the value ff281ffe-705c-4f53-9f37-a40e6f2c68f3 for the parameter - **ServicePrincipalName** in **Set-AzureRmKeyVaultAccessPolicy** cmdlet.

For ASM based VMs

```
PS C:\> $pol=Get-AzureRmRecoveryServicesBackupProtectionPolicy -Name "NewPolicy"
PS C:\> Enable-AzureRmRecoveryServicesBackupProtection -Policy $pol -Name "V1VM" -ServiceName "ServiceName1"
```

Modify a protection policy

In order to modify the policy, modify the `BackupSchedulePolicyObject` or `BackupRetentionPolicy` object and modify the policy using `Set-AzureRmRecoveryServicesBackupProtectionPolicy`

The following example changes the retention count to 365.

```
PS C:\> $retPol = Get-AzureRmRecoveryServicesBackupRetentionPolicyObject -WorkloadType "AzureVM"
PS C:\> $retPol.DailySchedule.DurationCountInDays = 365
PS C:\> $pol= Get-AzureRmRecoveryServicesBackupProtectionPolicy -Name NewPolicy
PS C:\> Set-AzureRmRecoveryServicesBackupProtectionPolicy -Policy $pol -RetentionPolicy $RetPol
```

Run an initial backup

The backup schedule triggers a full back up on the initial back up for the item. On subsequent back ups, the back up is an incremental copy. If you want to force the initial backup to happen at a certain time or even immediately then use the [Backup-AzureRmRecoveryServicesBackupItem](#) cmdlet:

```
PS C:\> $namedContainer = Get-AzureRmRecoveryServicesBackupContainer -ContainerType "AzureVM" -Status "Registered" -Name "V2VM"
PS C:\> $item = Get-AzureRmRecoveryServicesBackupItem -Container $namedContainer -WorkloadType "AzureVM"
PS C:\> $job = Backup-AzureRmRecoveryServicesBackupItem -Item $item
WorkloadName      Operation          Status           StartTime          EndTime
JobID
-----
-----
V2VM            Backup             InProgress       4/23/2016 5:00:30 PM
cf4b3ef5-2fac-4c8e-a215-d2eba4124f27
```

NOTE

The timezone of the `StartTime` and `EndTime` fields in PowerShell is UTC. However, when the time is shown in the Azure portal, the time is adjusted to your local timezone.

Monitoring a backup job

Most long-running operations in Azure Backup are modelled as a job. This makes it easy to track progress without having to keep the Azure portal open at all times.

To get the latest status of an in-progress job, use the `Get-AzureRmRecoveryServicesBackupJob` cmdlet.

```
PS C:\> $joblist = Get-AzureRmRecoveryServicesBackupJob -Status InProgress
PS C:\> $joblist[0]
WorkloadName      Operation          Status           StartTime          EndTime
JobID
-----
-----
V2VM            Backup             InProgress       4/23/2016 5:00:30 PM
cf4b3ef5-2fac-4c8e-a215-d2eba4124f27
```

Instead of polling these jobs for completion - which is unnecessary additional code - use the [Wait-AzureRmRecoveryServicesBackupJob](#) cmdlet. This cmdlet pauses the execution until either the job completes or the specified timeout value is reached.

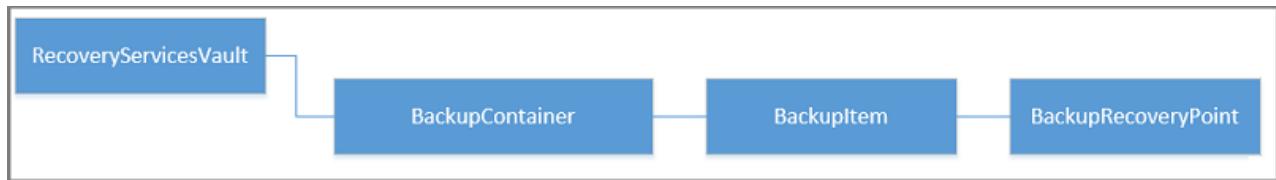
```
PS C:\> Wait-AzureRmRecoveryServicesBackupJob -Job $joblist[0] -Timeout 43200
```

Restore an Azure VM

There is a key difference between the restoring a VM using the Azure portal and restoring a VM using PowerShell. With PowerShell, the restore operation is complete once the disks and configuration information from the recovery point are created. The restore operation does not create a virtual machine. The instructions for creating the virtual machine from disks are provided. However, to fully restore a VM, you need to work through the following procedures:

- Select the VM
- Choose a recovery point
- Restore the disks
- Create the VM from stored disks

The graphic below shows the object hierarchy from the RecoveryServicesVault down to the BackupRecoveryPoint.



In order to restore backup data, identify the backed-up item and the recovery point that holds the point-in-time data. Then use the [Restore-AzureRmRecoveryServicesBackupItem](#) cmdlet to restore data from the vault to the customer's account.

Select the VM

To get the PowerShell object that identifies the right backup item, start from the container in the vault, and work your way down the object hierarchy. To select the container that represents the VM, use the [Get-AzureRmRecoveryServicesBackupContainer](#) cmdlet and pipe that to the [Get-AzureRmRecoveryServicesBackupItem](#) cmdlet.

```
PS C:\> $namedContainer = Get-AzureRmRecoveryServicesBackupContainer -ContainerType AzureVM -Status Registered -Name 'V2VM'  
PS C:\> $backupitem = Get-AzureRmRecoveryServicesBackupItem -Container $namedContainer -WorkloadType "AzureVM"
```

Choose a recovery point

Use the [Get-AzureRmRecoveryServicesBackupRecoveryPoint](#) cmdlet to list all the recovery points for the backup item. Then choose the recovery point to restore. If you are unsure which recovery point to use, it is a good practice to choose the most recent RecoveryPointType = AppConsistent point in the list.

In the following script, the variable, **\$rp**, is an array of recovery points for the selected backup item. The array is sorted in reverse order of time with the latest recovery point at index 0. Use standard PowerShell array indexing to pick the recovery point. For example: **\$rp[0]** will select the latest recovery point.

```

PS C:\> $startDate = (Get-Date).AddDays(-7)
PS C:\> $endDate = Get-Date
PS C:\> $rp = Get-AzureRmRecoveryServicesBackupRecoveryPoint -Item $backupitem -StartDate
$startdate.ToUniversalTime() -EndDate $enddate.ToUniversalTime()
PS C:\> $rp[0]
RecoveryPointAdditionalInfo :
SourceVMStorageType      : NormalStorage
Name                      : 15260861925810
ItemName                  : VM;iaasvmcontainer;RGName1;V2VM
RecoveryPointID           : /subscriptions/XX/resourceGroups/
RGName1/providers/Microsoft.RecoveryServices/vaults/testvault/backupFabrics/Azure/protectionContainers/IaaSVMCon
tainer;iaasvmcontainer;RGName1;V2VM/protectedItems/VM;iaasvmcontainer; RGName1;V2VM
                                         /recoveryPoints/15260861925810
RecoveryPointType         : AppConsistent
RecoveryPointTime          : 4/23/2016 5:02:04 PM
WorkloadType               : AzureVM
ContainerName              : IaaSVMContainer;iaasvmcontainer; RGName1;V2VM
ContainerType               : AzureVM
BackupManagementType       : AzureVM

```

Restore the disks

Use the [Restore-AzureRmRecoveryServicesBackupItem](#) cmdlet to restore data and configuration for a Backup item, to a recovery point. Once you have identified a recovery point use it as the value for the **-RecoveryPoint** parameter. In the previous example code, **\$rp[0]** was chosen as the recovery point to use. In the sample code below, **\$rp[0]** is specified as the recovery point to use for restoring to disk.

To restore the disks and configuration information

```

PS C:\> $restorejob = Restore-AzureRmRecoveryServicesBackupItem -RecoveryPoint $rp[0] -StorageAccountName
DestAccount -StorageAccountResourceGroupName DestRG
PS C:\> $restorejob
WorkloadName   Operation     Status        StartTime            EndTime       JobID
-----        -----        -----        -----
V2VM          Restore       InProgress    4/23/2016 5:00:30 PM          cf4b3ef5-
2fac-4c8e-a215-d2eba4124f27

```

Once the Restore job has completed, use the [Get-AzureRmRecoveryServicesBackupJobDetails](#) cmdlet to get the details of the restore operation. The JobDetails property has the information needed to rebuild the VM.

```

PS C:\> $restorejob = Get-AzureRmRecoveryServicesBackupJob -Job $restorejob
PS C:\> $details = Get-AzureRmRecoveryServicesBackupJobDetails

```

Once you restore the disks, go to the next section for information on creating the VM.

Create a VM from restored disks

After you have restored the disks, use these steps to create and configure the virtual machine from disk.

NOTE

If you are creating encrypted VMs using restored disks, your role should be allowed to perform **Microsoft.KeyVault/vaults/deploy/action**. If your role does not have this permission, create a custom role with this action. Refer [Custom Roles in Azure RBAC](#) to get more details.

1. Query the restored disk properties for the job details.

```

PS C:\> $properties = $details.properties
PS C:\> $storageAccountName = $properties["Target Storage Account Name"]
PS C:\> $containerName = $properties["Config Blob Container Name"]
PS C:\> $blobName = $properties["Config Blob Name"]

```

2. Set the Azure storage context and restore the JSON configuration file.

```

PS C:\> Set-AzureRmCurrentStorageAccount -Name $storageaccountname -ResourceGroupName testvault
PS C:\> $destination_path = "C:\vmconfig.json"
PS C:\> Get-AzureStorageBlobContent -Container $containerName -Blob $blobName -Destination
$destination_path
PS C:\> $obj = ((Get-Content -Path $destination_path -Encoding Unicode)).TrimEnd([char]0x00) |
ConvertFrom-Json

```

3. Use the JSON configuration file to create the VM configuration.

```

PS C:\> $vm = New-AzureRmVMConfig -VMSize $obj.HardwareProfile.VirtualMachineSize -VMName "testrestore"

```

4. Attach the OS disk and data disks.

For non-encrypted VMs,

```

PS C:\> Set-AzureRmVMOSDisk -VM $vm -Name "osdisk" -VhdUri
$obj.StorageProfile.OSDisk.VirtualHardDisk.Uri -CreateOption "Attach"
PS C:\> $vm.StorageProfile.OsDisk.OsType = $obj.StorageProfile.OSDisk.OperatingSystemType foreach($dd
in $obj.StorageProfile.DataDisks)
{
    $vm = Add-AzureRmVMDataDisk -VM $vm -Name "datadisk1" -VhdUri $dd.VirtualHardDisk.Uri -DiskSizeInGB
127 -Lun $dd.Lun -CreateOption Attach
}

```

For encrypted VMs, you need to specify [Key vault information](#) before you can attach disks.

```

PS C:\> Set-AzureRmVMOSDisk -VM $vm -Name "osdisk" -VhdUri
$obj.StorageProfile.OSDisk.VirtualHardDisk.Uri -DiskEncryptionKeyUrl
"https://ContosoKeyVault.vault.azure.net:443/secrets/ContosoSecret007" -DiskEncryptionKeyVaultId
"/subscriptions/abcdedf007-4xyz-1a2b-0000-
12a2b345675c/resourceGroups/ContosoRG108/providers/Microsoft.KeyVault/vaults/ContosoKeyVault" -
KeyEncryptionKeyUrl "https://ContosoKeyVault.vault.azure.net:443/keys/ContosoKey007" -
KeyEncryptionKeyVaultId "/subscriptions/abcdedf007-4xyz-1a2b-0000-
12a2b345675c/resourceGroups/ContosoRG108/providers/Microsoft.KeyVault/vaults/ContosoKeyVault" -
CreateOption "Attach" -Windows
PS C:\> $vm.StorageProfile.OsDisk.OsType = $obj.StorageProfile.OSDisk.OperatingSystemType foreach($dd
in $obj.StorageProfile.DataDisks)
{
    $vm = Add-AzureRmVMDataDisk -VM $vm -Name "datadisk1" -VhdUri $dd.VirtualHardDisk.Uri -DiskSizeInGB
127 -Lun $dd.Lun -CreateOption Attach
}

```

5. Set the Network settings.

```

PS C:\> $nicName="p1234"
PS C:\> $pip = New-AzureRmPublicIpAddress -Name $nicName -ResourceGroupName "test" -Location "WestUS" -
AllocationMethod Dynamic
PS C:\> $vnet = Get-AzureRmVirtualNetwork -Name "testvNET" -ResourceGroupName "test"
PS C:\> $nic = New-AzureRmNetworkInterface -Name $nicName -ResourceGroupName "test" -Location "WestUS" -
SubnetId $vnet.Subnets[$subnetindex].Id -PublicIpAddressId $pip.Id
PS C:\> $vm=Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id

```

6. Create the virtual machine.

```
PS C:\> $vm.StorageProfile.OsDisk.OsType = $obj.StorageProfile.OSDisk.OperatingSystemType  
PS C:\> New-AzureRmVM -ResourceGroupName "test" -Location "WestUS" -VM $vm
```

Next steps

If you prefer using PowerShell to engage with your Azure resources, check out the PowerShell article for protecting Windows Server, [Deploy and Manage Backup for Windows Server](#). There is also a PowerShell article for managing DPM backups, [Deploy and Manage Backup for DPM](#). Both of these articles have a version for Resource Manager deployments as well as Classic deployments.

Apply policies to Azure Resource Manager Virtual Machines

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 will describe how you can use Azure Resource Manager policies to define the desired behavior for your organization's Virtual Machines.

The outline for the steps to accomplish this is as below

1. Azure Resource Manager Policy 101
2. Define a policy for your Virtual Machine
3. Create the policy
4. Apply the policy

Azure Resource Manager Policy 101

For getting started with Azure Resource Manager policies, we recommend reading the article below and then continuing with the steps in this article. The article below describes the basic definition and structure of a policy, how policies get evaluated and gives various examples of policy definitions.

- [Use Policy to manage resources and control access](#)

Define a policy for your Virtual Machine

One of the common scenarios for an enterprise might be to only allow their users to create Virtual Machines from specific operating systems that have been tested to be compatible with a LOB application. Using an Azure Resource Manager policy this task can be accomplished in a few steps. In this policy example, we are going to allow only Windows Server 2012 R2 Datacenter Virtual Machines to be created. The policy definition looks like below

```

"if": {
  "allof": [
    {
      "field": "type",
      "equals": "Microsoft.Compute/virtualMachines"
    },
    {
      "not": {
        "allof": [
          {
            "field": "Microsoft.Compute/virtualMachines/imagePublisher",
            "equals": "MicrosoftWindowsServer"
          },
          {
            "field": "Microsoft.Compute/virtualMachines/imageOffer",
            "equals": "WindowsServer"
          },
          {
            "field": "Microsoft.Compute/virtualMachines/imageSku",
            "equals": "2012-R2-Datacenter"
          }
        ]
      }
    }
  ],
  "then": {
    "effect": "deny"
  }
}

```

The above policy can easily be modified to a scenario where you might want to allow any Windows Server Datacenter image to be used for a Virtual Machine deployment with the below change

```

{
  "field": "Microsoft.Compute/virtualMachines/imageSku",
  "like": "*Datacenter"
}

```

Virtual Machine Property Fields

The table below describes the Virtual Machine properties that can be used as fields in your policy definition. For more on policy fields, see the article below:

- [Fields and Sources](#)

FIELD NAME	DESCRIPTION
imagePublisher	Specifies the publisher of the image
imageOffer	Specifies the offer for the chosen image publisher
imageSku	Specifies the SKU for the chosen offer
imageVersion	Specifies the image version for the chosen SKU

Create the Policy

A policy can easily be created using the REST API directly or the PowerShell cmdlets. For creating the policy, see the article below:

- [Creating a Policy](#)

Apply the Policy

After creating the policy you'll need to apply it on a defined scope. The scope can be a subscription, resource group or even the resource. For applying the policy, see the article below:

- [Creating a Policy](#)

Manage Azure Virtual Machines using Resource Manager and PowerShell

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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.

Set variables

All the commands in the article require the name of the resource group where the virtual machine is located and the name of the virtual machine to manage. Replace the value of **\$rgName** with the name of the resource group that contains the virtual machine. Replace the value of **\$vmName** with the name of the VM. Create the variables.

```
$rgName = "resource-group-name"  
$vmName = "VM-name"
```

Display information about a virtual machine

Get the virtual machine information.

```
Get-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

It returns something like this example:

```

ResourceGroupName      : rg1
Id                   : /subscriptions/{subscription-id}/resourceGroups/
                      rg1/providers/Microsoft.Compute/virtualMachines/vm1
Name                 : vm1
Type                : Microsoft.Compute/virtualMachines
Location             : centralus
Tags                : {}
AvailabilitySetReference : {
    "id": "/subscriptions/{subscription-id}/resourceGroups/
          rg1/providers/Microsoft.Compute/availabilitySets/av1"
}
Extensions           : []
HardwareProfile      : {
    "vmSize": "Standard_A0"
}
InstanceView          : null
NetworkProfile        : {
    "networkInterfaces": [
        {
            "properties.primary": null,
            "id": "/subscriptions/{subscription-id}/resourceGroups/
                  rg1/providers/Microsoft.Network/networkInterfaces/nc1"
        }
    ]
}
OSProfile             : {
    "computerName": "vm1",
    "adminUsername": "myaccount1",
    "adminPassword": null,
    "customData": null,
    "windowsConfiguration": {
        "provisionVMAgent": true,
        "enableAutomaticUpdates": true,
        "timeZone": null,
        "additionalUnattendContents": [],
        "winRM": null
    },
    "linuxConfiguration": null,
    "secrets": []
}
Plan                 : null
ProvisioningState     : Succeeded
StorageProfile        : {
    "imageReference": {
        "publisher": "MicrosoftWindowsServer",
        "offer": "WindowsServer",
        "sku": "2012-R2-Datacenter",
        "version": "latest"
    },
    "osDisk": {
        "osType": "Windows",
        "encryptionSettings": null,
        "name": "osdisk",
        "vhd": {
            "Uri": "http://sa1.blob.core.windows.net/vhds/osdisk1.vhd"
        }
        "image": null,
        "caching": "ReadWrite",
        "createOption": "FromImage"
    },
    "dataDisks": []
}
DataDiskNames          : {}
NetworkInterfaceIDs   : {/subscriptions/{subscription-id}/resourceGroups/
                      rg1/providers/Microsoft.Network/networkInterfaces/nc1}

```

Stop a virtual machine

Stop the running virtual machine.

```
Stop-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

You're asked for confirmation:

```
Virtual machine stopping operation
This cmdlet will stop the specified virtual machine. Do you want to continue?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"):
```

Enter **Y** to stop the virtual machine.

After a few minutes, it returns something like this example:

```
StatusCode : Succeeded
StartTime   : 9/13/2016 12:11:57 PM
EndTime    : 9/13/2016 12:14:40 PM
```

Start a virtual machine

Start the virtual machine if it's stopped.

```
Start-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

After a few minutes, it returns something like this example:

```
StatusCode : Succeeded
StartTime   : 9/13/2016 12:32:55 PM
EndTime    : 9/13/2016 12:35:09 PM
```

If you want to restart a virtual machine that is already running, use **Restart-AzureRmVM** described next.

Restart a virtual machine

Restart the running virtual machine.

```
Restart-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

It returns something like this example:

```
StatusCode : Succeeded
StartTime   : 9/13/2016 12:54:40 PM
EndTime    : 9/13/2016 12:55:54 PM
```

Delete a virtual machine

Delete the virtual machine.

```
Remove-AzureRmVM -ResourceGroupName $rgName -Name $vmName
```

NOTE

You can use the **-Force** parameter to skip the confirmation prompt.

If you didn't use the -Force parameter, you're asked for confirmation:

```
Virtual machine removal operation
This cmdlet will remove the specified virtual machine. Do you want to continue?
[Y] Yes [N] No [S] Suspend [?] Help (default is "Y"):
```

It returns something like this example:

RequestId	IsSuccessStatusCode	StatusCode	ReasonPhrase
	True	OK	OK

Update a virtual machine

This example shows how to update the size of the virtual machine.

```
$vmSize = "Standard_A1"
$vm = Get-AzureRmVM -ResourceGroupName $rgName -Name $vmName
$vm.HardwareProfile.vmSize = $vmSize
Update-AzureRmVM -ResourceGroupName $rgName -VM $vm
```

It returns something like this example:

RequestId	IsSuccessStatusCode	StatusCode	ReasonPhrase
	True	OK	OK

See [Sizes for virtual machines in Azure](#) for a list of available sizes for a virtual machine.

Add a data disk to a virtual machine

This example shows how to add a data disk to an existing virtual machine.

```
$vm = Get-AzureRmVM -ResourceGroupName $rgName -Name $vmName
Add-AzureRmVMDataDisk -VM $vm -Name "disk-name" -VhdUri
"https://mystore1.blob.core.windows.net/vhds/datadisk1.vhd" -LUN 0 -Caching ReadWrite -DiskSizeInGB 1 -
CreateOption Empty
Update-AzureRmVM -ResourceGroupName $rgName -VM $vm
```

The disk that you add is not initialized. To initialize the disk, you can log in to it and use disk management. If you installed WinRM and a certificate on it when you created it, you can use remote PowerShell to initialize the disk. You can also use a custom script extension:

```
$location = "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

$letters = 70..89 | ForEach-Object { ([char]$_) }

$count = 0
$labels = @("data1","data2")

foreach($d in $disks) {
    $driveLetter = $letters[$count].ToString()
    $d |
    Initialize-Disk -PartitionStyle MBR -PassThru |
    New-Partition -UseMaximumSize -DriveLetter $driveLetter |
    Format-Volume -FileSystem NTFS -NewFileSystemLabel $labels[$count] `

        -Confirm:$false -Force
    $count++
}
```

Next Steps

If there were issues with a deployment, you might look at [Troubleshoot common Azure deployment errors with Azure Resource Manager](#)

Manage Azure Virtual Machines using Azure Resource Manager and C#

1/17/2017 • 9 min to read • [Edit on GitHub](#)

The tasks in this article show you how to manage virtual machines, such as starting, stopping, and updating. A virtual machine must exist in a resource group to complete the tasks in this article.

To complete the tasks in this article, you need:

- [Visual Studio](#)
- [An authentication token](#)

Create a Visual Studio project and install packages

NuGet packages are the easiest ways to install the libraries that you need to finish the tasks in this article. The libraries that you install for this article are the Azure Active Directory Authentication Library and the Compute Resource Provider Library. Complete these steps to get the libraries in Visual Studio:

1. Click **File > New > Project**.
2. In **Templates > Visual C#**, select **Console Application**, enter the name and location of the project, and then click **OK**.
3. Right-click the project name in the Solution Explorer, and then click **Manage NuGet Packages**.
4. Type *Active Directory* in the search box, click **Install** for the Active Directory Authentication Library package, and then follow the instructions to install the package.
5. At the top of the page, select **Include Prerelease**. Type *Microsoft.Azure.Management.Compute* in the search box, click **Install** for the Compute .NET Libraries, and then follow the instructions to install the package.

Now you're ready to start using the libraries to manage your virtual machines.

Set up the project

Now that the application is created and the libraries are installed, you create a token using the application information. This token is used to authenticate requests to Azure Resource Manager.

1. Open the Program.cs file for the project that you created, and then add these using statements to the top of the file:

```
using Microsoft.Azure;
using Microsoft.IdentityModel.Clients.ActiveDirectory;
using Microsoft.Azure.Management.Compute;
using Microsoft.Azure.Management.Compute.Models;
using Microsoft.Rest;
```

2. Add variables to the Main method of the Program class to specify the name of the resource group, and the name of the virtual machine, and your subscription identifier:

```
var groupName = "resource group name";
var vmName = "virtual machine name";
var subscriptionId = "subscription id";
```

You can find the subscription identifier by running Get-AzureRmSubscription.

- To get the token that is needed to create the credentials, add this method to the Program class:

```
private static async Task<AuthenticationResult> GetAccessTokenAsync()
{
    var cc = new ClientCredential("{client-id}", "{client-secret}");
    var context = new AuthenticationContext("https://login.windows.net/{tenant-id}");
    var token = await context.AcquireTokenAsync("https://management.azure.com/", cc);
    if (token == null)
    {
        throw new InvalidOperationException("Could not get the token");
    }
    return token;
}
```

Replace {client-id} with the identifier of the Azure Active Directory application, {client-secret} with the access key of the AD application, and {tenant-id} with the tenant identifier for your subscription. You can find the tenant id by running Get-AzureRmSubscription. You can find the access key by using the Azure portal.

- To create the credentials, add this code to the Main method in Program.cs:

```
var token = GetAccessTokenAsync();
var credential = new TokenCredentials(token.Result.AccessToken);
```

- Save the Program.cs file.

Display information about a virtual machine

- Add this method to the Program class in the project that you previously created:

```
public static async void GetVirtualMachineAsync(
    TokenCredentials credential,
    string groupName,
    string vmName,
    string subscriptionId)
{
    Console.WriteLine("Getting information about the virtual machine...");

    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    var vmResult = await computeManagementClient.VirtualMachines.GetAsync(
        groupName,
        vmName,
        InstanceViewTypes.InstanceView);

    Console.WriteLine("hardwareProfile");
    Console.WriteLine("    vmSize: " + vmResult.HardwareProfile.VmSize);

    Console.WriteLine("\nstorageProfile");
    Console.WriteLine("    imageReference");
    Console.WriteLine("        publisher: " + vmResult.StorageProfile.ImageReference.Publisher);
    Console.WriteLine("        offer: " + vmResult.StorageProfile.ImageReference.Offer);
    Console.WriteLine("        sku: " + vmResult.StorageProfile.ImageReference.Sku);
    Console.WriteLine("        version: " + vmResult.StorageProfile.ImageReference.Version);
    Console.WriteLine("    osDisk");
    Console.WriteLine("        osType: " + vmResult.StorageProfile.OsDisk.OsType);
    Console.WriteLine("        name: " + vmResult.StorageProfile.OsDisk.Name);
    Console.WriteLine("        createOption: " + vmResult.StorageProfile.OsDisk.CreateOption);
    Console.WriteLine("        uri: " + vmResult.StorageProfile.OsDisk.Vhd.Uri);
    Console.WriteLine("        caching: " + vmResult.StorageProfile.OsDisk.Caching);

    Console.WriteLine("\nosProfile");
    Console.WriteLine("    computerName: " + vmResult.OsProfile.ComputerName);
    Console.WriteLine("    adminUsername: " + vmResult.OsProfile.AdminUsername);
    Console.WriteLine("    provisionVMAgent: " +
```

```

vmResult.OsProfile.WindowsConfiguration.ProvisionVMAgent.Value);
Console.WriteLine(" enableAutomaticUpdates: " +
vmResult.OsProfile.WindowsConfiguration.EnableAutomaticUpdates.Value);

Console.WriteLine("\nnetworkProfile");
foreach (NetworkInterfaceReference nic in vmResult.NetworkProfile.NetworkInterfaces)
{
    Console.WriteLine(" networkInterface id: " + nic.Id);
}

Console.WriteLine("\nvmAgent");
Console.WriteLine(" vmAgentVersion" + vmResult.InstanceView.VmAgent.VmAgentVersion);
Console.WriteLine(" statuses");
foreach (InstanceViewStatus stat in vmResult.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("\ndisks");
foreach (DiskInstanceView idisk in vmResult.InstanceView.Disks)
{
    Console.WriteLine(" name: " + idisk.Name);
    Console.WriteLine(" statuses");
    foreach (InstanceViewStatus istat in idisk.Statuses)
    {
        Console.WriteLine(" code: " + istat.Code);
        Console.WriteLine(" level: " + istat.Level);
        Console.WriteLine(" displayStatus: " + istat.DisplayStatus);
        Console.WriteLine(" time: " + istat.Time);
    }
}

Console.WriteLine("\nVM general status");
Console.WriteLine(" provisioningStatus: " + vmResult.ProvisioningState);
Console.WriteLine(" id: " + vmResult.Id);
Console.WriteLine(" name: " + vmResult.Name);
Console.WriteLine(" type: " + vmResult.Type);
Console.WriteLine(" location: " + vmResult.Location);
Console.WriteLine("\nVM instance status");
foreach (InstanceViewStatus istat in vmResult.InstanceView.Statuses)
{
    Console.WriteLine("\n code: " + istat.Code);
    Console.WriteLine(" level: " + istat.Level);
    Console.WriteLine(" displayStatus: " + istat.DisplayStatus);
}
}

```

2. To call the method that you just added, add this code to the Main method:

```

GetVirtualMachineAsync(
    credential,
    groupName,
    vmName,
    subscriptionId);
Console.WriteLine("\nPress enter to continue...");
Console.ReadLine();

```

3. Save the Program.cs file.
4. Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

When you run this method, you should see something like this example:

```
Getting information about the virtual machine...
hardwareProfile
  vmSize: Standard_A0

storageProfile
  imageReference
    publisher: MicrosoftWindowsServer
    offer: WindowsServer
    sku: 2012-R2-Datacenter
    version: latest
  osDisk
    osType: Windows
    name: myosdisk
    createOption: FromImage
    uri: http://store1.blob.core.windows.net/vhds/myosdisk.vhd
    caching: ReadWrite

osProfile
  computerName: vm1
  adminUsername: account1
  provisionVMAgent: True
  enableAutomaticUpdates: True

networkProfile
  networkInterface
    id: /subscriptions/{subscription-id}
      /resourceGroups/rg1/providers/Microsoft.Network/networkInterfaces/nc1

vmAgent
  vmAgentVersion2.7.1198.766
  statuses
    code: ProvisioningState/succeeded
    level: Info
    displayStatus: Ready
    message: GuestAgent is running and accepting new configurations.
    time: 4/13/2016 8:35:32 PM

disks
  name: myosdisk
  statuses
    code: ProvisioningState/succeeded
    level: Info
    displayStatus: Provisioning succeeded
    time: 4/13/2016 8:04:36 PM

VM general status
  provisioningStatus: Succeeded
  id: /subscriptions/{subscription-id}
    /resourceGroups/rg1/providers/Microsoft.Compute/virtualMachines/vm1
  name: vm1
  type: Microsoft.Compute/virtualMachines
  location: centralus

VM instance status
  code: ProvisioningState/succeeded
  level: Info
  displayStatus: Provisioning succeeded
  code: PowerState/running
  level: Info
  displayStatus: VM running
```

Stop a virtual machine

You can stop a virtual machine in two ways. 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. Comment out any code that you previously added to the Main method, except the code to get the credentials.
2. Add this method to the Program class:

```
public static async void StopVirtualMachineAsync(
    TokenCredentials credential,
    string groupName,
    string vmName,
    string subscriptionId)
{
    Console.WriteLine("Stopping the virtual machine...");
    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    await computeManagementClient.VirtualMachines.PowerOffAsync(groupName, vmName);
}
```

If you want to deallocate the virtual machine, change the PowerOff call to this code:

```
computeManagementClient.VirtualMachines.Deallocate(groupName, vmName);
```

3. To call the method that you just added, add this code to the Main method:

```
StopVirtualMachineAsync(
    credential,
    groupName,
    vmName,
    subscriptionId);
Console.WriteLine("\nPress enter to continue...");
Console.ReadLine();
```

4. Save the Program.cs file.
5. Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

You should see the status of the virtual machine change to Stopped. If you ran the method calling Deallocate, the status is Stopped (deallocated).

Start a virtual machine

1. Comment out any code that you previously added to the Main method, except the code to get the credentials.
2. Add this method to the Program class:

```
public static async void StartVirtualMachineAsync(
    TokenCredentials credential,
    string groupName,
    string vmName,
    string subscriptionId)
{
    Console.WriteLine("Starting the virtual machine...");
    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    await computeManagementClient.VirtualMachines.StartAsync(groupName, vmName);
}
```

3. To call the method that you just added, add this code to the Main method:

```

StartVirtualMachineAsync(
    credential,
    groupName,
    vmName,
    subscriptionId);
Console.WriteLine("\nPress enter to continue...");
Console.ReadLine();

```

4. Save the Program.cs file.
5. Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

You should see the status of the virtual machine change to Running.

Restart a running virtual machine

1. Comment out any code that you previously added to the Main method, except the code to get the credentials.
2. Add this method to the Program class:

```

public static async void RestartVirtualMachineAsync(
    TokenCredentials credential,
    string groupName,
    string vmName,
    string subscriptionId)
{
    Console.WriteLine("Restarting the virtual machine...");
    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    await computeManagementClient.VirtualMachines.RestartAsync(groupName, vmName);
}

```

3. To call the method that you just added, add this code to the Main method:

```

RestartVirtualMachineAsync(
    credential,
    groupName,
    vmName,
    subscriptionId);
Console.WriteLine("\nPress enter to continue...");
Console.ReadLine();

```

4. Save the Program.cs file.
5. Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

Resize a virtual machine

This example shows you how to change the size of a running virtual machine.

1. Comment out any code that you previously added to the Main method, except the code to get the credentials.
2. Add this method to the Program class:

```

public static async void UpdateVirtualMachineAsync(
    TokenCredentials credential,
    string groupName,
    string vmName,
    string subscriptionId)
{
    Console.WriteLine("Updating the virtual machine...");
    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    var vmResult = await computeManagementClient.VirtualMachines.GetAsync(groupName, vmName);
    vmResult.HardwareProfile.VmSize = "Standard_A1";
    await computeManagementClient.VirtualMachines.CreateOrUpdateAsync(groupName, vmName, vmResult);
}

```

- To call the method that you just added, add this code to the Main method:

```

UpdateVirtualMachineAsync(
    credential,
    groupName,
    vmName,
    subscriptionId);
Console.WriteLine("\nPress enter to continue...");
Console.ReadLine();

```

- Save the Program.cs file.
- Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

You should see the size of the virtual machine change to Standard_A1.

Add a data disk to a virtual machine

This example shows you how to add a data disk to a running virtual machine.

- Comment out any code that you previously added to the Main method, except the code to get the credentials.
- Add this method to the Program class:

```

public static async void AddDataDiskAsync(
    TokenCredentials credential,
    string groupName,
    string vmName,
    string subscriptionId)
{
    Console.WriteLine("Adding the disk to the virtual machine...");
    var computeManagementClient = new ComputeManagementClient(credential)
        { SubscriptionId = subscriptionId };
    var vmResult = await computeManagementClient.VirtualMachines.GetAsync(groupName, vmName);
    vmResult.StorageProfile.DataDisks.Add(
        new DataDisk
        {
            Lun = 0,
            Name = "mydatadisk1",
            Vhd = new VirtualHardDisk
            {
                Uri = "https://mystorage1.blob.core.windows.net/vhds/mydatadisk1.vhd"
            },
            CreateOption = DiskCreateOptionTypes.Empty,
            DiskSizeGB = 2,
            Caching = CachingTypes.ReadWrite
        });
    await computeManagementClient.VirtualMachines.CreateOrUpdateAsync(groupName, vmName, vmResult);
}

```

- To call the method that you just added, add this code to the Main method:

```
AddDataDiskAsync(  
    credential,  
    groupName,  
    vmName,  
    subscriptionId);  
Console.WriteLine("\nPress enter to continue...");  
Console.ReadLine();
```

- Save the Program.cs file.
- Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

Delete a virtual machine

- Comment out any code that you previously added to the Main method, except the code to get the credentials.
- Add this method to the Program class:

```
public static async void DeleteVirtualMachineAsync(  
    TokenCredentials credential,  
    string groupName,  
    string vmName,  
    string subscriptionId)  
{  
    Console.WriteLine("Deleting the virtual machine...");  
    var computeManagementClient = new ComputeManagementClient(credential)  
    {  
        SubscriptionId = subscriptionId  
    };  
    await computeManagementClient.VirtualMachines.DeleteAsync(groupName, vmName);  
}
```

- To call the method that you just added, add this code to the Main method:

```
DeleteVirtualMachineAsync(  
    credential,  
    groupName,  
    vmName,  
    subscriptionId);  
Console.WriteLine("\nPress enter to continue...");  
Console.ReadLine();
```

- Save the Program.cs file.
- Click **Start** in Visual Studio, and then sign in to Azure AD using the same username and password that you use with your subscription.

Next Steps

If there were issues with a deployment, you might look at [Troubleshoot common Azure deployment errors with Azure Resource Manager](#)

Deploy and manage virtual machines by using Azure Resource Manager templates and the Azure CLI

1/17/2017 • 24 min to read • [Edit on GitHub](#)

- [Quick-create a virtual machine in Azure](#)
- [Deploy a virtual machine in Azure from a template](#)
- [Create a virtual machine from a custom image](#)
- [Deploy a virtual machine that uses a virtual network and a load balancer](#)
- [Remove a resource group](#)
- [Show the log for a resource group deployment](#)
- [Display information about a virtual machine](#)
- [Connect to a Linux-based virtual machine](#)
- [Stop a virtual machine](#)
- [Start a virtual machine](#)
- [Attach a data disk](#)

Getting ready

Before you can use the Azure CLI with Azure resource groups, you need to have the right Azure CLI version and an Azure account. If you don't have the Azure CLI, [install it](#).

Update your Azure CLI version to 0.9.0 or later

Type `azure --version` to see whether you have already installed version 0.9.0 or later.

```
azure --version
0.9.0 (node: 0.10.25)
```

If your version is not 0.9.0 or later, you need to update it by using one of the native installers or through **npm** by typing `npm update -g azure-cli`.

You can also run Azure CLI as a Docker container by using the following [Docker image](#). From a Docker host, run the following command:

```
docker run -it microsoft/azure-cli
```

Set your Azure account and subscription

If you don't already have an Azure subscription but you do have an MSDN subscription, you can activate your [MSDN subscriber benefits](#). Or you can sign up for a [free trial](#).

Now [log in to your Azure account interactively](#) by typing `azure login` and following the prompts for an interactive login experience to your Azure account.

NOTE

If you have a work or school ID and you know you do not have two-factor authentication enabled, you can **also** use `azure login -u` along with the work or school ID to log in *without* an interactive session. If you don't have a work or school ID, you can [create a work or school id from your personal Microsoft account](#) to log in the same way.

Your account may have more than one subscription. You can list your subscriptions by typing `azure account list`, which might look something like this:

```
```azure cli
azure account list info: Executing command account list
data: Contoso Admin
----- data: Fabrikam dev
xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx true
data: Fabrikam test xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx false
data: Contoso production xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx false
```

You can set the current Azure subscription by typing the following. Use the subscription name or the ID that has the resources you want to manage.

```
```azurecli
azure account set <subscription name or ID> true
```

Switch to the Azure CLI resource group mode

By default, the Azure CLI starts in the service management mode (**asm** mode). Type the following to switch to resource group mode.

```
azure config mode arm
```

Understanding Azure resource templates and resource groups

Most applications are built from a combination of different resource types (such as one or more VMs and storage accounts, a SQL database, a virtual network, or a content delivery network). The default Azure service management API and the Azure classic portal represented these items by using a service-by-service approach. This approach requires you to deploy and manage the individual services individually (or find other tools that do so), and not as a single logical unit of deployment.

Azure Resource Manager templates, however, make it possible for you to deploy and manage these different resources as one logical deployment unit in a declarative fashion. Instead of imperatively telling Azure what to deploy one command after another, you describe your entire deployment in a JSON file -- all of the resources and associated configuration and deployment parameters -- and tell Azure to deploy those resources as one group.

You can then manage the overall life cycle of the group's resources by using Azure CLI resource management commands to:

- Stop, start, or delete all of the resources within the group at once.
- Apply Role-Based Access Control (RBAC) rules to lock down security permissions on them.
- Audit operations.
- Tag resources with additional metadata for better tracking.

You can learn lots more about Azure resource groups and what they can do for you in the [Azure Resource Manager overview](#). If you're interested in authoring templates, see [Authoring Azure Resource Manager templates](#).

Task: Quick-create a VM in Azure

Sometimes you know what image you need, and you need a VM from that image right now and you don't care too much about the infrastructure -- maybe you have to test something on a clean VM. That's when you want to use the `azure vm quick-create` command, and pass the arguments necessary to create a VM and its infrastructure.

First, create your resource group.

```
azure group create coreos-quick westus
info:  Executing command group create
+ Getting resource group coreos-quick
+ Creating resource group coreos-quick
info:  Created resource group coreos-quick
data:  Id:          /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/coreos-quick
data:  Name:        coreos-quick
data:  Location:   westus
data:  Provisioning State: Succeeded
data:  Tags:
data:
info:  group create command OK
```

Second, you'll need an image. To find an image with the Azure CLI, see [Navigating and selecting Azure virtual machine images with PowerShell and the Azure CLI](#). But for this article, here's a short list of popular images. We'll use CoreOS's Stable image for this quick-create.

NOTE

For `ComputeImageVersion`, you can also simply supply 'latest' as the parameter in both the template language and in the Azure CLI. This will allow you to always use the latest and patched version of the image without having to modify your scripts or templates. This is shown below.

PUBLISHERNAME	OFFER	SKU	VERSION
OpenLogic	CentOS	7	7.0.201503
OpenLogic	CentOS	7.1	7.1.201504
CoreOS	CoreOS	Beta	647.0.0
CoreOS	CoreOS	Stable	633.1.0
MicrosoftDynamicsNAV	DynamicsNAV	2015	8.0.40459
MicrosoftSharePoint	MicrosoftSharePointServer	2013	1.0.0
msopentech	Oracle-Database-12c-Weblogic-Server-12c	Standard	1.0.0
msopentech	Oracle-Database-12c-Weblogic-Server-12c	Enterprise	1.0.0
MicrosoftSQLServer	SQL2014-WS2012R2	Enterprise-Optimized-for-DW	12.0.2430

PUBLISHERNAME	OFFER	SKU	VERSION
MicrosoftSQLServer	SQL2014-WS2012R2	Enterprise-Optimized-for-OLTP	12.0.2430
Canonical	UbuntuServer	12.04.5-LTS	12.04.201504230
Canonical	UbuntuServer	14.04.2-LTS	14.04.201503090
MicrosoftWindowsServer	WindowsServer	2012-Datacenter	3.0.201503
MicrosoftWindowsServer	WindowsServer	2012-R2-Datacenter	4.0.201503
MicrosoftWindowsServer	WindowsServer	Windows-Server-Technical-Preview	5.0.201504
MicrosoftWindowsServerEssentials	WindowsServerEssentials	WindowsServerEssentials	1.0.141204
MicrosoftWindowsServerHPCPack	WindowsServerHPCPack	2012R2	4.3.4665

Just create your VM by entering the `azure vm quick-create` command and being ready for the prompts. It should look something like this:

```
azure vm quick-create
info: Executing command vm quick-create
Resource group name: coreos-quick
Virtual machine name: coreos
Location name: westus
Operating system Type [Windows, Linux]: linux
ImageURN (format: "publisherName:offer:skus:version"): coreos:coreos:stable:latest
User name: ops
Password: *****
Confirm password: *****
+ Looking up the VM "coreos"
info: Using the VM Size "Standard_A1"
info: The [OS, Data] Disk or image configuration requires storage account
+ Retrieving storage accounts
info: Could not find any storage accounts in the region "westus", trying to create new one
+ Creating storage account "cli9fd3fce49e9a9b3d14302" in "westus"
+ Looking up the storage account cli9fd3fce49e9a9b3d14302
+ Looking up the NIC "coreo-westu-1430261891570-nic"
info: An nic with given name "coreo-westu-1430261891570-nic" not found, creating a new one
+ Looking up the virtual network "coreo-westu-1430261891570-vnet"
info: Preparing to create new virtual network and subnet
/ Creating a new virtual network "coreo-westu-1430261891570-vnet" [address prefix: "10.0.0.0/16"] with subnet
"coreo-westu-1430261891570-snet" [address prefix: "10.0.1.0/24"]
+ Looking up the virtual network "coreo-westu-1430261891570-vnet"
+ Looking up the subnet "coreo-westu-1430261891570-snet" under the virtual network "coreo-westu-1430261891570-vnet"
info: Found public ip parameters, trying to setup PublicIP profile
+ Looking up the public ip "coreo-westu-1430261891570-pip"
info: PublicIP with given name "coreo-westu-1430261891570-pip" not found, creating a new one
+ Creating public ip "coreo-westu-1430261891570-pip"
+ Looking up the public ip "coreo-westu-1430261891570-pip"
+ Creating NIC "coreo-westu-1430261891570-nic"
+ Looking up the NIC "coreo-westu-1430261891570-nic"
+ Creating VM "coreos"
+ Looking up the VM "coreos"
+ Looking up the NIC "coreo-westu-1430261891570-nic"
+ Looking up the public ip "coreo-westu-1430261891570-pip"
```

```

data:   Id                      :/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx/resourceGroups/coreos-quick/providers/Microsoft.Compute/virtualMachines/coreos
data:   ProvisioningState        :Succeeded
data:   Name                     :coreos
data:   Location                 :westus
data:   FQDN                     :coreo-westu-1430261891570-pip.westus.cloudapp.azure.com
data:   Type                     :Microsoft.Compute/virtualMachines
data:
data:   Hardware Profile:
data:     Size                  :Standard_A1
data:
data:   Storage Profile:
data:     Image reference:
data:       Publisher           :coreos
data:       Offer                :coreos
data:       Sku                  :stable
data:       Version              :633.1.0
data:
data:   OS Disk:
data:     OSType               :Linux
data:     Name                 :cli9fd3fce49e9a9b3d-os-1430261892283
data:     Caching               :ReadWrite
data:     CreateOption          :FromImage
data:     Vhd:
data:       Uri
:https://cli9fd3fce49e9a9b3d14302.blob.core.windows.net/vhds/cli9fd3fce49e9a9b3d-os-1430261892283.vhd
data:
data:   OS Profile:
data:     Computer Name         :coreos
data:     User Name             :ops
data:   Linux Configuration:
data:     Disable Password Auth :false
data:
data:   Network Profile:
data:     Network Interfaces:
data:       Network Interface #1:
data:         Id                  :/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx/resourceGroups/coreos-quick/providers/Microsoft.Network/networkInterfaces/coreo-westu-1430261891570-
nic
data:         Primary            :true
data:         MAC Address         :00-0D-3A-30-72-E3
data:         Provisioning State :Succeeded
data:         Name                :coreo-westu-1430261891570-nic
data:         Location             :westus
data:         Private IP alloc-method :Dynamic
data:         Private IP address   :10.0.1.4
data:         Public IP address    :104.40.24.124
data:         FQDN                :coreo-westu-1430261891570-pip.westus.cloudapp.azure.com
info:   vm quick-create command OK

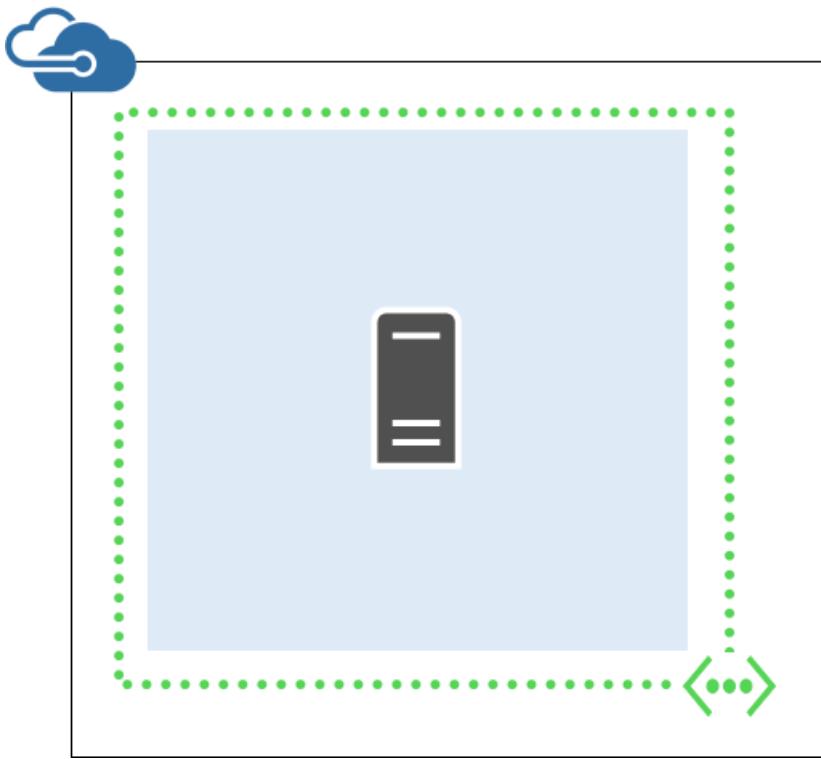
```

And away you go with your new VM.

Task: Deploy a VM in Azure from a template

Use the instructions in these sections to deploy a new Azure VM by using a template with the Azure CLI. This template creates a single virtual machine in a new virtual network with a single subnet, and unlike

`azure vm quick-create`, enables you to describe what you want precisely and repeat it without errors. Here's what this template creates:



Step 1: Examine the JSON file for the template parameters

Here are the contents of the JSON file for the template. (The template is also located in [GitHub](#).)

Templates are flexible, so the designer may have chosen to give you lots of parameters or chosen to offer only a few by creating a template that is more fixed. In order to collect the information you need to pass the template as parameters, open the template file (this topic has a template inline, below) and examine the **parameters** values.

In this case, the template below will ask for:

- A unique storage account name.
- An admin user name for the VM.
- A password.
- A domain name for the outside world to use.
- An Ubuntu Server version number -- but it will accept only one of a list.

See more about [username and password requirements](#).

Once you decide on these values, you're ready to create a group for and deploy this template into your Azure subscription.

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {  
    "newStorageAccountName": {  
      "type": "string",  
      "metadata": {  
        "description": "Unique DNS name for the storage account where the virtual machine's disks will be placed."  
      }  
    },  
    "adminUsername": {  
      "type": "string",  
      "metadata": {  
        "description": "User name for the virtual machine."  
      }  
    },  
    "adminPassword": {  
      "type": "string",  
      "minLength": 12,  
      "maxLength": 128,  
      "pattern": "^(?=.*[a-zA-Z])(?=.*[0-9]).*$"  
    }  
  }  
}
```

```

    "type": "securestring",
    "metadata": {
        "description": "Password for the virtual machine."
    },
},
{
    "dnsNameForPublicIP": {
        "type": "string",
        "metadata": {
            "description": "Unique DNS name for the public IP used to access the virtual machine."
        }
    },
},
{
    "ubuntuOSVersion": {
        "type": "string",
        "defaultValue": "14.04.2-LTS",
        "allowedValues": [
            "12.04.5-LTS",
            "14.04.2-LTS",
            "15.04"
        ],
        "metadata": {
            "description": "The Ubuntu version for the VM. This will pick a fully patched image of this given Ubuntu version. Allowed values: 12.04.5-LTS, 14.04.2-LTS, 15.04."
        }
    }
},
{
    "variables": {
        "location": "West US",
        "imagePublisher": "Canonical",
        "imageOffer": "UbuntuServer",
        "OSDiskName": "osdiskforlinuxsimple",
        "nicName": "myVMNic",
        "addressPrefix": "10.0.0.0/16",
        "subnetName": "Subnet",
        "subnetPrefix": "10.0.0.0/24",
        "storageAccountType": "Standard_LRS",
        "publicIPAddressName": "myPublicIP",
        "publicIPAddressType": "Dynamic",
        "vmStorageAccountContainerName": "vhds",
        "vmName": "MyUbuntuVM",
        "vmSize": "Standard_D1",
        "virtualNetworkName": "MyVNET",
        "vnetID": "[resourceId('Microsoft.Network/virtualNetworks',variables('virtualNetworkName'))]",
        "subnetRef": "[concat(variables('vnetID'), '/subnets/', variables('subnetName'))]"
    }
},
{
    "resources": [
        {
            "type": "Microsoft.Storage/storageAccounts",
            "name": "[parameters('newStorageAccountName')]",
            "apiVersion": "2015-05-01-preview",
            "location": "[variables('location')]",
            "properties": {
                "accountType": "[variables('storageAccountType')]"
            }
        },
        {
            "apiVersion": "2015-05-01-preview",
            "type": "Microsoft.Network/publicIPAddresses",
            "name": "[variables('publicIPAddressName')]",
            "location": "[variables('location')]",
            "properties": {
                "publicIPAllocationMethod": "[variables('publicIPAddressType')]",
                "dnsSettings": {
                    "domainNameLabel": "[parameters('dnsNameForPublicIP')]"
                }
            }
        },
        {
            "apiVersion": "2015-05-01-preview",
            "type": "Microsoft.Network/virtualNetworks"
        }
    ]
}

```

```

    "type": "Microsoft.Network/virtualNetworks",
    "name": "[variables('virtualNetworkName')]",
    "location": "[variables('location')]",
    "properties": {
        "addressSpace": {
            "addressPrefixes": [
                "[variables('addressPrefix')]"
            ]
        },
        "subnets": [
            {
                "name": "[variables('subnetName')]",
                "properties": {
                    "addressPrefix": "[variables('subnetPrefix')]"
                }
            }
        ]
    }
},
{
    "apiVersion": "2015-05-01-preview",
    "type": "Microsoft.Network/networkInterfaces",
    "name": "[variables('nicName')]",
    "location": "[variables('location')]",
    "dependsOn": [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIPAddressName'))]",
        "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'))]"
    ],
    "properties": {
        "ipConfigurations": [
            {
                "name": "ipconfig1",
                "properties": {
                    "privateIPAllocationMethod": "Dynamic",
                    "publicIPAddress": {
                        "id": "[resourceId('Microsoft.Network/publicIPAddresses', variables('publicIPAddressName'))]"
                    },
                    "subnet": {
                        "id": "[variables('subnetRef')]"
                    }
                }
            }
        ]
    }
},
{
    "apiVersion": "2015-05-01-preview",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[variables('vmName')]",
    "location": "[variables('location')]",
    "dependsOn": [
        "[concat('Microsoft.Storage/storageAccounts/', parameters('newStorageAccountName'))]",
        "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties": {
        "hardwareProfile": {
            "vmSize": "[variables('vmSize')]"
        },
        "osProfile": {
            "computerName": "[variables('vmName')]",
            "adminUsername": "[parameters('adminUsername')]",
            "adminPassword": "[parameters('adminPassword')]"
        },
        "storageProfile": {
            "imageReference": {
                "publisher": "[variables('imagePublisher')]",
                "offer": "[variables('imageOffer')]",
                "sku": "[parameters('ubuntuOSVersion')]",
                "version": "latest"
            }
        }
    }
}
]

```

```

        },
        "osDisk": {
            "name": "osdisk",
            "vhd": {
                "uri": "
[concat('http://',parameters('newStorageAccountName'),'.blob.core.windows.net/',variables('vmStorageAccountContainerName'), '/',variables('OSDiskName'), '.vhd')]"
            },
            "caching": "ReadWrite",
            "createOption": "FromImage"
        }
    },
    "networkProfile": {
        "networkInterfaces": [
            {
                "id": "[resourceId('Microsoft.Network/networkInterfaces',variables('nicName'))]"
            }
        ]
    }
}
]
}

```

Step 2: Create the virtual machine by using the template

Once you have your parameter values ready, you must create a resource group for your template deployment and then deploy the template.

To create the resource group, type `azure group create <group name> <location>` with the name of the group you want and the datacenter location into which you want to deploy. This happens quickly:

```

azure group create myResourceGroup westus
info:  Executing command group create
+ Getting resource group myResourceGroup
+ Creating resource group myResourceGroup
info:  Created resource group myResourceGroup
data:  Id:          /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/myResourceGroup
data:  Name:        myResourceGroup
data:  Location:   westus
data:  Provisioning State: Succeeded
data:  Tags:
data:
info:  group create command OK

```

Now to create the deployment, call `azure group deployment create` and pass:

- The template file (if you saved the above JSON template to a local file).
- A template URI (if you want to point at the file in GitHub or some other web address).
- The resource group into which you want to deploy.
- An optional deployment name.

You will be prompted to supply the values of parameters in the "parameters" section of the JSON file. When you have specified all the parameter values, your deployment will begin.

Here is an example:

```
azure group deployment create --template-uri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/101-vm-simple-linux/azuredeploy.json myResourceGroup firstDeployment
info: Executing command group deployment create
info: Supply values for the following parameters
newStorageAccountName: storageaccount
adminUsername: ops
adminPassword: password
dnsNameForPublicIP: newdomainname
```

You will receive the following type of information:

```
+ Initializing template configurations and parameters
+ Creating a deployment
info: Created template deployment "firstDeployment"
+ Registering providers
info: Registering provider microsoft.storage
info: Registering provider microsoft.network
info: Registering provider microsoft.compute
+ Waiting for deployment to complete
data: DeploymentName      : firstDeployment
data: ResourceGroupName   : myResourceGroup
data: ProvisioningState   : Succeeded
data: Timestamp           : 2015-04-28T07:53:55.1828878Z
data: Mode                 : Incremental
data: TemplateLink        : https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/101-simple-linux-vm/azuredeploy.json
data: ContentVersion       : 1.0.0.0
data: Name                Type      Value
data: -----
data: newStorageAccountName String    storageaccount
data: adminUsername         String    ops
data: adminPassword         SecureString undefined
data: dnsNameForPublicIP   String    newdomainname
data: ubuntuOSVersion       String    14.10
info: group deployment create command OK
```

Task: Create a custom VM image

You've seen the basic usage of templates above, so now we can use similar instructions to create a custom VM from a specific .vhf file in Azure by using a template via the Azure CLI. The difference here is that this template creates a single virtual machine from a specified virtual hard disk (VHD).

Step 1: Examine the JSON file for the template

Here are the contents of the JSON file for the template that this section uses as an example. (The template is also located in [GitHub](#).)

Again, you will need to find the values you want to enter for the parameters that do not have default values. When you run the `azure group deployment create` command, the Azure CLI will prompt you to enter those values.

```
{
  "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/deploymentTemplate.json",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "userImageStorageAccountName": {
      "type": "string",
      "defaultValue" : "userImageStorageAccountName"
    },
    "userImageStorageContainerName" : {
      "type" : "string",
      "defaultValue" : "userImageStorageContainerName"
    },
    "userImageVhdName" : {
      "type" : "string"
    }
  }
}
```

```

        "type" : "string",
        "defaultValue" : "userImageVhdName"
    },
    "dnsNameForPublicIP" : {
        "type" : "string",
        "defaultValue": "uniqueDnsNameForPublicIP"
    },
    "adminUserName": {
        "type": "string"
    },
    "adminPassword": {
        "type": "securestring"
    },
    "osType" : {
        "type" : "string",
        "allowedValues" : [
            "windows",
            "linux"
        ]
    },
    "subscriptionId": {
        "type" : "string"
    },
    "location": {
        "type": "String",
        "defaultValue" : "West US"
    },
    "vmSize": {
        "type": "string",
        "defaultValue": "Standard_A2"
    },
    "publicIPAddressName": {
        "type": "string",
        "defaultValue" : "myPublicIP"
    },
    "vmName": {
        "type": "string",
        "defaultValue" : "myVM"
    },
    "virtualNetworkName": {
        "type" : "string",
        "defaultValue" : "myVNET"
    },
    "nicName": {
        "type" : "string",
        "defaultValue": "myNIC"
    }
},
"variables": {
    "addressPrefix": "10.0.0.0/16",
    "subnet1Name": "Subnet-1",
    "subnet2Name": "Subnet-2",
    "subnet1Prefix" : "10.0.0.0/24",
    "subnet2Prefix" : "10.0.1.0/24",
    "publicIPAddressType" : "Dynamic",
    "vnetID": "[resourceId('Microsoft.Network/virtualNetworks',parameters('virtualNetworkName'))]",
    "subnet1Ref" : "[concat(variables('vnetID'), '/subnets/', variables('subnet1Name'))]",
    "userImageName" : "
[concat('http://', parameters('userImageStorageAccountName'), '.blob.core.windows.net/', parameters('userImageStorageContainerName'), '/', parameters('userImageVhdName'))]",
    "osDiskVhdName" : "
[concat('http://', parameters('userImageStorageAccountName'), '.blob.core.windows.net/', parameters('userImageStorageContainerName'), '/', parameters('vmName'), 'osDisk.vhd')]"
},
"resources": [
{
    "apiVersion": "2014-12-01-preview",
    "type": "Microsoft.Network/publicIPAddresses",
    "name": "[parameters('publicIPAddressName')]",
    "dependsOn": [
        "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'), '/subnets/', variables('subnet1Name'))]"
    ],
    "properties": {
        "ipAddress": "[parameters('publicIPAddressName')]"
    }
}
]

```

```

    "location": "[parameters('location')]",
    "properties": {
        "publicIPAllocationMethod": "[variables('publicIPAddressType')]",
        "dnsSettings": {
            "domainNameLabel": "[parameters('dnsNameForPublicIP')]"
        }
    }
},
{
    "apiVersion": "2014-12-01-preview",
    "type": "Microsoft.Network/virtualNetworks",
    "name": "[parameters('virtualNetworkName')]",
    "location": "[parameters('location')]",
    "properties": {
        "addressSpace": {
            "addressPrefixes": [
                "[variables('addressPrefix')]"
            ]
        },
        "subnets": [
            {
                "name": "[variables('subnet1Name')]",
                "properties" : {
                    "addressPrefix": "[variables('subnet1Prefix')]"
                }
            },
            {
                "name": "[variables('subnet2Name')]",
                "properties" : {
                    "addressPrefix": "[variables('subnet2Prefix')]"
                }
            }
        ]
    }
},
{
    "apiVersion": "2014-12-01-preview",
    "type": "Microsoft.Network/networkInterfaces",
    "name": "[parameters('nicName')]",
    "location": "[parameters('location')]",
    "dependsOn": [
        "[concat('Microsoft.Network/publicIPAddresses/', parameters('publicIPPropertyName'))]",
        "[concat('Microsoft.Network/virtualNetworks/', parameters('virtualNetworkName'))]"
    ],
    "properties": {
        "ipConfigurations": [
            {
                "name": "ipconfig1",
                "properties": {
                    "privateIPAllocationMethod": "Dynamic",
                    "publicIPAddress": {
                        "id": "
[resourceId('Microsoft.Network/publicIPAddresses',parameters('publicIPPropertyName'))]"
                    },
                    "subnet": {
                        "id": "[variables('subnet1Ref')]"
                    }
                }
            }
        ]
    }
},
{
    "apiVersion": "2014-12-01-preview",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[parameters('vmName')]",
    "location": "[parameters('location')]",
    "dependsOn": [
        "[concat('Microsoft.Network/networkInterfaces/', parameters('nicName'))]"
    ]
}

```

```

        ],
        "properties": {
            "hardwareProfile": {
                "vmSize": "[parameters('vmSize')]"
            },
            "osProfile": {
                "computername": "[parameters('vmName')]",
                "adminUsername": "[parameters('adminUsername')]",
                "adminPassword": "[parameters('adminPassword')]"
            },
            "storageProfile": {
                "osDisk" : {
                    "name" : "[concat(parameters('vmName'),'-osDisk')]",
                    "osType" : "[parameters('osType')]",
                    "caching" : "ReadWrite",
                    "image" : {
                        "uri" : "[variables('userImageName')]"
                    },
                    "vhd" : {
                        "uri" : "[variables('osDiskVhdName')]"
                    }
                }
            },
            "networkProfile": {
                "networkInterfaces" : [
                    {
                        "id": "[resourceId('Microsoft.Network/networkInterfaces',parameters('nicName'))]"
                    }
                ]
            }
        }
    }
]
}

```

Step 2: Obtain the VHD

Obviously, you'll need a .vhd for this. You can use one you already have in Azure, or you can upload one.

For a Windows-based virtual machine, see [Create and upload a Windows Server VHD to Azure](#).

For a Linux-based virtual machine, see [Creating and uploading a virtual hard disk that contains the Linux operating system](#).

Step 3: Create the virtual machine by using the template

Now you're ready to create a new virtual machine based on the .vhd. Create a group to deploy into, by using

```
azure group create <location> :
```

```

azure group create myResourceGroupUser eastus
info:   Executing command group create
+ Getting resource group myResourceGroupUser
+ Creating resource group myResourceGroupUser
info:   Created resource group myResourceGroupUser
data:   Id:          /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxx/resourceGroups/myResourceGroupUser
data:   Name:        myResourceGroupUser
data:   Location:    eastus
data:   Provisioning State: Succeeded
data:   Tags:
data:
info:   group create command OK

```

Then create the deployment by using the `--template-uri` option to call in the template directly (or you can use the `--template-file` option to use a file that you have saved locally). Note that because the template has defaults specified, you are prompted for only a few things. If you deploy the template in different places, you may find that

some naming collisions occur with the default values (particularly the DNS name you create).

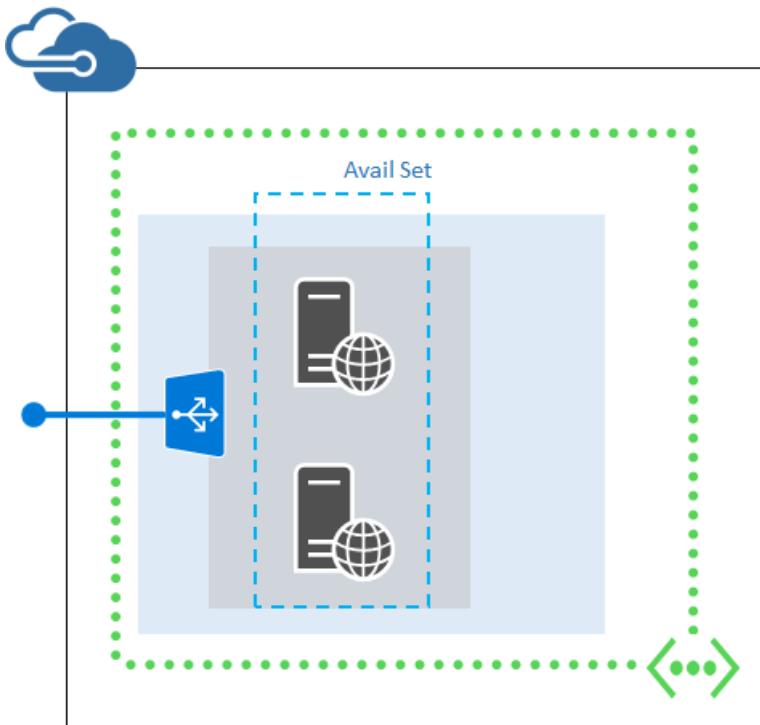
```
azure group deployment create \
> --template-uri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/101-vm-from-user-
image/azuredeploy.json \
> myResourceGroup \
> customVhdDeployment
info: Executing command group deployment create
info: Supply values for the following parameters
adminUserName: ops
adminPassword: password
osType: linux
subscriptionId: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Output looks something like the following:

```
+ Initializing template configurations and parameters
+ Creating a deployment
info: Created template deployment "customVhdDeployment"
+ Registering providers
info: Registering provider microsoft.network
info: Registering provider microsoft.compute
+ Waiting for deployment to complete
error: Deployment provisioning state was not successful
data: DeploymentName : customVhdDeployment
data: ResourceGroupName : myResourceGroupUser
data: ProvisioningState : Succeeded
data: Timestamp : 2015-04-28T14:55:48.0963829Z
data: Mode : Incremental
data: TemplateLink : https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/101-vm-
from-user-image/azuredeploy.json
data: ContentVersion : 1.0.0.0
data: Name Type Value
data: -----
data: userImageStorageAccountName String userImageStorageAccountName
data: userImageStorageContainerName String userImageStorageContainerName
data: userImageVhdName String userImageVhdName
data: dnsNameForPublicIP String uniqueDnsNameForPublicIP
data: adminUserName String ops
data: adminPassword SecureString undefined
data: osType String linux
data: subscriptionId String xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
data: location String West US
data: vmSize String Standard_A2
data: publicIPAddressName String myPublicIP
data: vmName String myVM
data: virtualNetworkName String myVNET
data: nicName String myNIC
info: group deployment create command OK
```

Task: Deploy a multi-VM application that uses a virtual network and an external load balancer

This template allows you to create two virtual machines under a load balancer and configure a load-balancing rule on Port 80. This template also deploys a storage account, virtual network, public IP address, availability set, and network interfaces.



Follow these steps to deploy a multi-VM application that uses a virtual network and a load balancer by using a Resource Manager template in the GitHub template repository via Azure PowerShell commands.

Step 1: Examine the JSON file for the template

Here are the contents of the JSON file for the template. If you want the most recent version, it's located [at the Github repository for templates](#). This topic uses the `--template-uri` switch to call in the template, but you can also use the `--template-file` switch to pass a local version.

```
{
  "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/deploymentTemplate.json",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "location": {
      "type": "string",
      "metadata": {
        "description": "Location of resources"
      }
    },
    "storageAccountName": {
      "type": "string",
      "metadata": {
        "description": "Name of storage account"
      }
    },
    "adminUsername": {
      "type": "string",
      "metadata": {
        "description": "Admin user name"
      }
    },
    "adminPassword": {
      "type": "securestring",
      "metadata": {
        "description": "Admin password"
      }
    },
    "dnsNameforLBIP": {
      "type": "string",
      "metadata": {
        "description": "DNS for load balancer IP"
      }
    }
  }
}
```

```

    },
    "backendPort": {
        "type": "int",
        "defaultValue": 3389,
        "metadata": {
            "description": "Back-end port"
        }
    },
    "vmNamePrefix": {
        "type": "string",
        "defaultValue": "myVM",
        "metadata": {
            "description": "Prefix to use for VM names"
        }
    },
    "vmSourceImageName": {
        "type": "string",
        "defaultValue": "a699494373c04fc0bc8f2bb1389d6106__Windows-Server-2012-R2-201412.01-en.us-127GB.vhd"
    },
    "lbName": {
        "type": "string",
        "defaultValue": "myLB",
        "metadata": {
            "description": "Load balancer name"
        }
    },
    "nicNamePrefix": {
        "type": "string",
        "defaultValue": "nic",
        "metadata": {
            "description": "Network interface name prefix"
        }
    },
    "publicIPAddressName": {
        "type": "string",
        "defaultValue": "myPublicIP",
        "metadata": {
            "description": "Public IP name"
        }
    },
    "vnetName": {
        "type": "string",
        "defaultValue": "myVNET",
        "metadata": {
            "description": "Virtual network name"
        }
    },
    "vmSize": {
        "type": "string",
        "defaultValue": "Standard_A1",
        "metadata": {
            "description": "Size of the VM"
        }
    }
},
"variables": {
    "storageAccountType": "Standard_LRS",
    "vmStorageAccountContainerName": "vhds",
    "availabilitySetName": "myAvSet",
    "addressPrefix": "10.0.0.0/16",
    "subnetName": "Subnet-1",
    "subnetPrefix": "10.0.0.0/24",
    "publicIPAddressType": "Dynamic",
    "vnetID": "[resourceId('Microsoft.Network/virtualNetworks',parameters('vnetName'))]",
    "subnetRef": "[concat(variables('vnetID'), '/subnets/', variables ('subnetName'))]",
    "publicIPAddressID": "
[resourceId('Microsoft.Network/publicIPAddresses',parameters('publicIPAddressName'))]",
    "lbID": "[resourceId('Microsoft.Network/loadBalancers',parameters('lbName'))]",
    "numberOfInstances": 2
}

```

```

    "nicId1": "[resourceId('Microsoft.Network/networkInterfaces', concat(parameters('nicNamePrefix'), 0))]",
    "nicId2": "[resourceId('Microsoft.Network/networkInterfaces', concat(parameters('nicNamePrefix'), 1))]",
    "frontEndIPConfigID": "[concat(variables('lbID'), '/frontendIPConfigurations/LBFE')]",
    "backEndIPConfigID1": "[concat(variables('nicId1'), '/ipConfigurations/ipconfig1')]",
    "backEndIPConfigID2": "[concat(variables('nicId2'), '/ipConfigurations/ipconfig1')]",
    "sourceImageName": "[concat('/', subscription().subscriptionId, '/services/images/', parameters('vmSourceImageName'))]",
    "lbPoolID": "[concat(variables('lbID'), '/backendAddressPools/LBBE')]",
    "lbProbeID": "[concat(variables('lbID'), '/probes/tcpProbe')]"
},
"resources": [
{
    "type": "Microsoft.Storage/storageAccounts",
    "name": "[parameters('storageAccountName')]",
    "apiVersion": "2015-05-01-preview",
    "location": "[parameters('location')]",
    "properties": {
        "accountType": "[variables('storageAccountType')]"
    }
},
{
    "type": "Microsoft.Compute/availabilitySets",
    "name": "[variables('availabilitySetName')]",
    "apiVersion": "2015-05-01-preview",
    "location": "[parameters('location')]",
    "properties": { }
},
{
    "apiVersion": "2015-05-01-preview",
    "type": "Microsoft.Network/publicIPAddresses",
    "name": "[parameters('publicIPPropertyName')]",
    "location": "[parameters('location')]",
    "properties": {
        "publicIPAllocationMethod": "[variables('publicIPAddressType')]",
        "dnsSettings": {
            "domainNameLabel": "[parameters('dnsNameforLBIP')]"
        }
    }
},
{
    "apiVersion": "2015-05-01-preview",
    "type": "Microsoft.Network/virtualNetworks",
    "name": "[parameters('vnetName')]",
    "location": "[parameters('location')]",
    "properties": {
        "addressSpace": {
            "addressPrefixes": [
                "[variables('addressPrefix')]"
            ]
        },
        "subnets": [
            {
                "name": "[variables('subnetName')]",
                "properties": {
                    "addressPrefix": "[variables('subnetPrefix')]"
                }
            }
        ]
    }
},
{
    "apiVersion": "2015-05-01-preview",
    "type": "Microsoft.Network/networkInterfaces",
    "name": "[concat(parameters('nicNamePrefix'), copyindex())]",
    "location": "[parameters('location')]",
    "copy": {
        "name": "nicLoop",
        "count": "[variables('numberOfInstances')]"
    }
}
]

```

```

        ],
        "dependsOn": [
            "[concat('Microsoft.Network/virtualNetworks/', parameters('vnetName'))]"
        ],
        "properties": {
            "ipConfigurations": [
                {
                    "name": "ipconfig1",
                    "properties": {
                        "privateIPAllocationMethod": "Dynamic",
                        "subnet": {
                            "id": "[variables('subnetRef')]"
                        }
                    }
                },
                "loadBalancerBackendAddressPools": [
                    {
                        "id": ""
                    }
                ],
                "loadBalancerInboundNatRules": [
                    {
                        "id": ""
                    }
                ]
            ]
        }
    },
    {
        "apiVersion": "2015-05-01-preview",
        "name": "[parameters('lbName')]",
        "type": "Microsoft.Network/loadBalancers",
        "location": "[parameters('location')]",
        "dependsOn": [
            "nicLoop",
            "[concat('Microsoft.Network/publicIPAddresses/', parameters('publicIPAddressName'))]"
        ],
        "properties": {
            "frontendIPConfigurations": [
                {
                    "name": "LBFE",
                    "properties": {
                        "publicIPAddress": {
                            "id": "[variables('publicIPAddressID')]"
                        }
                    }
                }
            ],
            "backendAddressPools": [
                {
                    "name": "LBBE"
                }
            ],
            "inboundNatRules": [
                {
                    "name": "RDP-VM1",
                    "properties": {
                        "frontendIPConfiguration": {
                            "id": "[variables('frontEndIPConfigID')]"
                        },
                        "protocol": "tcp",
                        "frontendPort": 50001,
                        "backendPort": 2222
                    }
                }
            ]
        }
    }
]

```

```

        "backendPort": 3389,
        "enableFloatingIP": false
    }
},
{
    "name": "RDP-VM2",
    "properties": {
        "frontendIPConfiguration":
        {
            "id": "[variables('frontEndIPConfigID')]"
        },
        "protocol": "tcp",
        "frontendPort": 50002,
        "backendPort": 3389,
        "enableFloatingIP": false
    }
}
],
"loadBalancingRules": [
{
    "name": "LBRule",
    "properties": {
        "frontendIPConfiguration": {
            "id": "[variables('frontEndIPConfigID')]"
        },
        "backendAddressPool": {
            "id": "[variables('lbPoolID')]"
        },
        "protocol": "tcp",
        "frontendPort": 80,
        "backendPort": 80,
        "enableFloatingIP": false,
        "idleTimeoutInMinutes": 5,
        "probe": {
            "id": "[variables('lbProbeID')]"
        }
    }
}
],
"probes": [
{
    "name": "tcpProbe",
    "properties": {
        "protocol": "tcp",
        "port": 80,
        "intervalInSeconds": "5",
        "numberOfProbes": "2"
    }
}
]
}
},
{
    "apiVersion": "2015-05-01-preview",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[concat(parameters('vmNamePrefix'), copyindex())]",
    "copy": {
        "name": "virtualMachineLoop",
        "count": "[variables('numberOfInstances')]"
    },
    "location": "[parameters('location')]",
    "dependsOn": [
        "[concat('Microsoft.Storage/storageAccounts/', parameters('storageAccountName'))]",
        "[concat('Microsoft.Network/networkInterfaces/', parameters('nicNamePrefix'), copyindex())]",
        "[concat('Microsoft.Compute/availabilitySets/', variables('availabilitySetName'))]"
    ],
    "properties": {
        "availabilitySet": {
            "id": "[resourceId('Microsoft.Compute/availabilitySets',variables('availabilitySetName'))]"
        }
    }
}
]
```

```

    },
    "hardwareProfile": {
        "vmSize": "[parameters('vmSize')]"
    },
    "osProfile": {
        "computerName": "[concat(parameters('vmNamePrefix'), copyIndex())]",
        "adminUsername": "[parameters('adminUsername')]",
        "adminPassword": "[parameters('adminPassword')]"
    },
    "storageProfile": {
        "sourceImage": {
            "id": "[variables('sourceImageName')]"
        },
        "destinationVhdsContainer": "
[concat('http://',parameters('storageAccountName'),'.blob.core.windows.net/',variables('vmStorageAccountContainerName'), '/')]"
    },
    "networkProfile": {
        "networkInterfaces": [
            {
                "id": "
[resourceId('Microsoft.Network/networkInterfaces',concat(parameters('nicNamePrefix'),copyindex()))]"
            }
        ]
    }
}
]
}

```

Step 2: Create the deployment by using the template

Create a resource group for the template by using `azure group create <location>`. Then, create a deployment into that resource group by using `azure group deployment create` and passing the resource group, passing a deployment name, and answering the prompts for parameters in the template that did not have default values.

```

azure group create lbgroup westus
info:   Executing command group create
+ Getting resource group lbgroup
+ Creating resource group lbgroup
info:   Created resource group lbgroup
data:   Id:          /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/lbgroup
data:   Name:        lbgroup
data:   Location:   westus
data:   Provisioning State: Succeeded
data:   Tags:
data:
info:   group create command OK

```

Now use the `azure group deployment create` command and the `--template-uri` option to deploy the template. Be ready with your parameter values when it prompts you, as shown below.

```

azure group deployment create \
> --template-uri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/201-2-vms-
loadbalancer-lbrules/azuredeploy.json \
> lbgroup \
> newdeployment
info: Executing command group deployment create
info: Supply values for the following parameters
location: westus
newStorageAccountName: storagename
adminUsername: ops
adminPassword: password
dnsNameforLBIP: lbdomainname
+ Initializing template configurations and parameters
+ Creating a deployment
info: Created template deployment "newdeployment"
+ Registering providers
info: Registering provider microsoft.storage
info: Registering provider microsoft.compute
info: Registering provider microsoft.network
+ Waiting for deployment to complete
data: DeploymentName : newdeployment
data: ResourceGroupName : lbgroup
data: ProvisioningState : Succeeded
data: Timestamp : 2015-04-28T20:58:40.1678876Z
data: Mode : Incremental
data: TemplateLink : https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/201-2-
vms-loadbalancer-lbrules/azuredeploy.json
data: ContentVersion : 1.0.0.0
data: Name Type Value
data: -----
data: location String westus
data: newStorageAccountName String storagename
data: adminUsername String ops
data: adminPassword SecureString undefined
data: dnsNameforLBIP String lbdomainname
data: backendPort Int 3389
data: vmNamePrefix String myVM
data: imagePublisher String MicrosoftWindowsServer
data: imageOffer String WindowsServer
data: imageSKU String 2012-R2-Datacenter
data: lbName String myLB
data: nicNamePrefix String nic
data: publicIPAddressName String myPublicIP
data: vnetName String myVNET
data: vmSize String Standard_A1
info: group deployment create command OK

```

Note that this template deploys a Windows Server image; however, it could easily be replaced by any Linux image. Want to create a Docker cluster with multiple swarm managers? [You can do it.](#)

Task: Remove a resource group

Remember that you can redeploy to a resource group, but if you are done with one, you can delete it by using

```
azure group delete <group name> .
```

```

azure group delete myResourceGroup
info: Executing command group delete
Delete resource group myResourceGroup? [y/n] y
+ Deleting resource group myResourceGroup
info: group delete command OK

```

Task: Show the log for a resource group deployment

This one is common while you're creating or using templates. The call to display the deployment logs for a group is `azure group log show <groupname>`, which displays quite a bit of information that's useful for understanding why something happened -- or didn't. (For more information on troubleshooting your deployments, as well as other information about issues, see [Troubleshoot common Azure deployment errors with Azure Resource Manager](#).)

To target specific failures, for example, you might use tools like **jq** to query things a bit more precisely, such as which individual failures you need to correct. The following example uses **jq** to parse a deployment log for **lbgrou**, looking for failures.

```
azure group log show lbgrou -l --json | jq '.[] | select(.status.value == "Failed") | .properties'
```

You can discover very quickly what went wrong, fix, and retry. In the following case, the template had been creating two VMs at the same time, which created a lock on the .vhdx. (After we modified the template, the deployment succeeded quickly.)

```
{
  "statusCode": "Conflict",
  "statusMessage": "{\"status\":\"Failed\",\"error\":{\"code\":\"ResourceDeploymentFailure\",\"message\":\"The resource operation completed with terminal provisioning state 'Failed'.\"},\"details\": [{\"code\":\"AcquireDiskLeaseFailed\",\"message\":\"Failed to acquire lease while creating disk 'osdisk' using blob with URI http://storage.blob.core.windows.net/vhds/osdisk.vhd.\"]}}"
}
```

Task: Display information about a virtual machine

You can see information about specific VMs in your resource group by using the

`azure vm show <groupname> <vmname>` command. If you have more than one VM in your group, you might first need to list the VMs in a group by using `azure vm list <groupname>`.

```
azure vm list zoo
info:  Executing command vm list
+ Getting virtual machines
data:    Name      ProvisioningState  Location   Size
data:    -----  -----
data:    myVM0    Succeeded          westus     Standard_A1
data:    myVM1    Failed            westus     Standard_A1
```

And then, looking up myVM1:

```
azure vm show zoo myVM1
info: Executing command vm show
+ Looking up the VM "myVM1"
+ Looking up the NIC "nic1"
data: Id :/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxx/resourceGroups/zoo/providers/Microsoft.Compute/virtualMachines/myVM1
data: ProvisioningState :Failed
data: Name :myVM1
data: Location :westus
data: Type :Microsoft.Compute/virtualMachines
data:
data: Hardware Profile:
data: Size :Standard_A1
data:
data: Storage Profile:
data: Image reference:
data: Publisher :MicrosoftWindowsServer
data: Offer :WindowsServer
data: Sku :2012-R2-Datacenter
data: Version :latest
data:
data: OS Disk:
data: OSType :Windows
data: Name :osdisk
data: Caching :ReadWrite
data: CreateOption :FromImage
data: Vhd:
data: Uri :http://zoostorageralph.blob.core.windows.net/vhds/osdisk.vhd
data:
data: OS Profile:
data: Computer Name :myVM1
data: User Name :ops
data: Windows Configuration:
data: Provision VM Agent :true
data: Enable automatic updates :true
data:
data: Network Profile:
data: Network Interfaces:
data: Network Interface #1:
data: Id :/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxx/resourceGroups/zoo/providers/Microsoft.Network/networkInterfaces/nic1
data: Primary :false
data: Provisioning State :Succeeded
data: Name :nic1
data: Location :westus
data: Private IP alloc-method :Dynamic
data: Private IP address :10.0.0.5
data:
data: AvailabilitySet:
data: Id :/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxx/resourceGroups/zoo/providers/Microsoft.Compute/availabilitySets/MYAVSET
info: vm show command OK
```

NOTE

If you want to programmatically store and manipulate the output of your console commands, you may want to use a JSON parsing tool such as [jq](#) or [jsawk](#), or language libraries that are good for the task.

Task: Log on to a Linux-based virtual machine

Typically Linux machines are connected to through SSH. For more information, see [How to use SSH with Linux on Azure](#).

Task: Stop a VM

Run this command:

```
azure vm stop <group name> <virtual machine name>
```

IMPORTANT

Use this parameter to keep the virtual IP (VIP) of the vnet in case it's the last VM in that vnet.

If you use the `StayProvisioned` parameter, you'll still be billed for the VM.

Task: Start a VM

Run this command:

```
azure vm start <group name> <virtual machine name>
```

Task: Attach a data disk

You'll also need to decide whether to attach a new disk or one that contains data. For a new disk, the command creates the .vhd file and attaches it in the same command.

To attach a new disk, run this command:

```
azure vm disk attach-new <resource-group> <vm-name> <size-in-gb>
```

To attach an existing data disk, run this command:

```
azure vm disk attach <resource-group> <vm-name> [vhd-url]
```

Then you'll need to mount the disk, as you normally would in Linux.

Next steps

For far more examples of Azure CLI usage with the **arm** mode, see [Using the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager](#). To learn more about Azure resources and their concepts, see [Azure Resource Manager overview](#).

For more templates you can use, see [Azure Quickstart templates](#) and [Application frameworks using templates](#).

Virtual machine extensions and features for Windows

1/17/2017 • 7 min to read • [Edit on GitHub](#)

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](#).

To work through an example where a VM extension is used in an end-to-end application deployment, see [Automating application deployments to Azure virtual machines](#).

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-AzureRmVMExtensionImageType | `  
Get-AzureRmVMExtensionImage | 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](#).

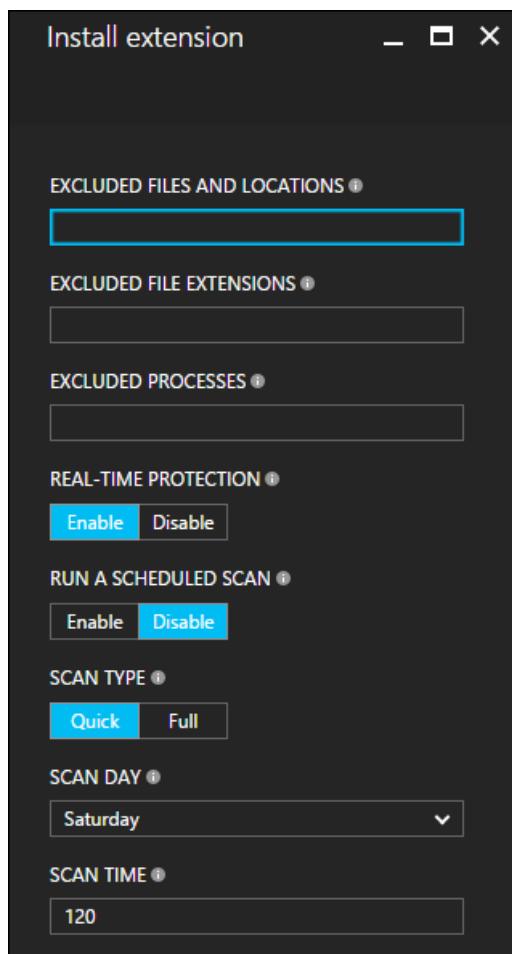
```
$cred=Get-Credential  
  
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" -VMName "myVM" -Name "myVMAccess" `  
-Location WestUS -UserName $cred.GetNetworkCredential().Username `  
-Password $cred.GetNetworkCredential().Password -typeHandlerVersion "2.0"
```

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 -user ',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 -user ',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

Extension Name	Description	More Information
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

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 Server 2008 R2, 2012, 2012 R2, and 2016 releases.

Script Location

The script needs to be stored in Azure 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": "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": "1.8",
    "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	1.8
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
storageAccountKey (e.g.)	TmJK/1N3AbAZ3q/+hOXoi/l73zOqsaxXDhqa9Y83/v5UpXQp2DQIBuv2Tifp60cE/OaHsJZmQZ7teQfczQj8hg==

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 in the following directory on the target virtual machine.

```
C:\WindowsAzure\Logs\Plugins\Microsoft.Compute.CustomScriptExtension
```

The script itself is downloaded into the following directory on the target virtual machine.

```
C:\Packages\Plugins\Microsoft.Compute.CustomScriptExtension\1.*\Downloads
```

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

1/17/2017 • 6 min to read • [Edit on GitHub](#)

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 secure 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"
.setLocation = "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

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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

1/17/2017 • 6 min to read • [Edit on GitHub](#)

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'.
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.

PROPERTY NAME	TYPE	DESCRIPTION
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.
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

1/17/2017 • 10 min to read • [Edit on GitHub](#)

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.aspx>.

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=w1",
    "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 $instances
-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 SetAzureVMLLogCollector.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 rl
-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
#


```

```

{
    $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

1/17/2017 • 6 min to read • [Edit on GitHub](#)

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-AzureRMVMDiagnosticsExtension](#) PowerShell cmdlet as shown below.

```
$vm_resourcegroup = "myvmresourcegroup"
$vm_name = "myvm"
$diagnosticsconfig_path = "DiagnosticsPubConfig.xml"

Set-AzureRmVMDiagnosticsExtension -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-AzureRMVMDiagnosticsExtension* 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 $diagnosticsstorage_name -  
StorageAccountKey $diagnosticsstorage_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 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/MyWindowsVM">
```

- 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>
    </DiagnosticMonitorConfiguration>
  </WadCfg>
</PublicConfig>
```

```

        </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>
    </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.

OMS virtual machine extension for Windows

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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.

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.

```
{
  "type": "extensions",
  "name": "Microsoft.EnterpriseCloud.Monitoring",
  "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

NAME	VALUE / EXAMPLE
publisher	Microsoft.EnterpriseCloud.Monitoring
type	MicrosoftMonitoringAgent
typeHandlerVersion	1.0
workspaceId (e.g.)	6f680a37-00c6-41c7-a93f-1437e3462574
workspaceKey (e.g.)	z4bU3p1/GrnWpQkky4gdabWXAhbWSTz70hm4m2Xt92XI+rS RgE8qVvRhsGo9TxffbrTahyrwv35W0pOqQU7uQ==

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](#).

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](#).

Authoring Azure Resource Manager templates with Windows VM extensions

1/17/2017 • 1 min to read • [Edit on GitHub](#)

Overview of Azure Resource Manager templates

Azure Resource Manager templates allow you to declaratively specify the Azure IaaS infrastructure in Json language by defining the dependencies between resources. For a detailed overview of Azure Resource Manager Templates, please refer to the article below:

[Resource Group Overview](#)

Sample template snippet for VM extensions

Deploying VM extensions as part of an Azure Resource Manager template requires you to declaratively specify the extension configuration in the template. Here is the format for specifying the extension configuration.

```
{  
  "type": "Microsoft.Compute/virtualMachines/extensions",  
  "name": "MyExtension",  
  "apiVersion": "2015-05-01-preview",  
  "location": "[parameters('location')]",  
  "dependsOn": ["[concat('Microsoft.Compute/virtualMachines/',parameters('vmName'))]"],  
  "properties":  
  {  
    "publisher": "Publisher Namespace",  
    "type": "extension Name",  
    "typeHandlerVersion": "extension version",  
    "settings": {  
      // Extension specific configuration goes in here.  
    }  
  }  
}
```

As you can see from the above, the extension template contains two main parts:

1. Extension name, publisher and version
2. Extension Configuration.

Identifying the publisher, type, and typeHandlerVersion for any extension

Azure VM extensions are published by Microsoft and trusted 3rd party publishers and each extension is uniquely identified by its publisher,type and the typeHandlerVersion. These can be determined as following:

From Azure PowerShell, run the following Azure PowerShell cmdlet:

```
Get-AzureVMAvailableExtension
```

This cmdlet returns the publisher name, extension name, and version as follows:

Publisher	: Microsoft.Azure.Extensions
ExtensionName	: DockerExtension
Version	: 1.0

These three properties map to "publisher", "type", and "typeHandlerVersion" respectively in the above template snippet.

NOTE

It's always recommended to use the latest extension version to get the most updated functionality.

Identifying the schema for the extension configuration parameters

The next step with authoring an extension template is to identify the format for providing configuration parameters. Each extension supports its own set of parameters.

To look at sample configurations for Windows extensions, see [Windows extensions samples](#).

Please refer to the following to get a fully complete template with VM extensions.

[Custom Script Extension on a Windows VM](#)

After authoring the template, you can deploy it using Azure PowerShell.

Exporting Resource Groups that contain VM extensions

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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.VMDiagnosticsSettings_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 `IaaSDiagnostic`. Once the extensions `protectedSettings` object has been located, take note of each parameter. In the example of the `IaaSDiagnostic` extension, the require 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": "IaaSDiagnostics",
        "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 `SecureObject` 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": "SecureObject"  
},  
"storageAccountKey": {  
    "defaultValue": null,  
    "type": "SecureObject"  
}
```

At this point, the template can be deployed using any template deployment method.

Azure Windows VM Extension Configuration Samples

1/17/2017 • 6 min to read • [Edit on GitHub](#)

This article provides sample configuration for configuring Azure VM Extensions for Windows VMs.

To learn more about these extensions, see [Azure VM Extensions Overview](#).

To learn more about authoring extension templates, see [Authoring Extension Templates](#).

This article lists expected configuration values for some of the Windows Extensions.

Sample template snippet for VM Extensions with IaaS VMs.

The template snippet for Deploying extensions looks as following:

```
{  
  "type": "Microsoft.Compute/virtualMachines/extensions",  
  "name": "MyExtension",  
  "apiVersion": "2015-05-01-preview",  
  "location": "[parameters('location')]",  
  "dependsOn": "[concat('Microsoft.Compute/virtualMachines/',parameters('vmName'))]",  
  "properties":  
  {  
    "publisher": "Publisher Namespace",  
    "type": "extension Name",  
    "typeHandlerVersion": "extension version",  
    "autoUpgradeMinorVersion":true,  
    "settings": {  
      // Extension specific configuration goes in here.  
    }  
  }  
}
```

Sample template snippet for VM Extensions with VM Scale Sets.

```
{  
  "type": "Microsoft.Compute/virtualMachineScaleSets",  
  ....  
  "extensionProfile":{  
    "extensions": [  
      {  
        "name": "extension Name",  
        "properties": {  
          "publisher": "Publisher Namespace",  
          "type": "extension Name",  
          "typeHandlerVersion": "extension version",  
          "autoUpgradeMinorVersion": true,  
          "settings": {  
            // Extension specific configuration goes in here.  
          }  
        }  
      }  
    ]  
  }  
}
```

Before deploying the extension please check the latest extension version and replace the "typeHandlerVersion"

with the current latest version.

Rest of the article provides sample configurations for Windows VM Extensions.

Before deploying the extension please check the latest extension version and replace the "typeHandlerVersion" with the current latest version.

CustomScript Extension 1.4.

```
{  
    "publisher": "Microsoft.Compute",  
    "type": "CustomScriptExtension",  
    "typeHandlerVersion": "1.4",  
    "settings": {  
        "fileUris": [  
            "http://Yourstorageaccount.blob.core.windows.net/customscriptfiles/start.ps1"  
        ],  
        "commandToExecute": "powershell.exe -ExecutionPolicy Unrestricted -start.ps1"  
    },  
    "protectedSettings": {  
        "storageAccountName": "yourStorageAccountName",  
        "storageAccountKey": "yourStorageAccountKey"  
    }  
}
```

Parameter description:

- fileUris : Comma seperated list of urls of the files that will be downloaded on the VM by the Extension. No files are downloaded if nothing is specified. If the files are in Azure Storage, the fileURLs can be marked private and the correspoding storageAccountName and storageAccountKey can be passed as private parameters to access these files.
- commandToExecute : [Mandatory Parameter] : This is the command that will be executed by the Extension.
- storageAccountName : [Optional Parameter] : Storage Account Name for accessing the fileURLs, if they are marked as private.
- storageAccountKey : [Optional Parameter] : Storage Account Key for accessing the fileURLs, if they are marked as private.

CustomScript Extension 1.7.

Please refer to CustomScript version 1.4 for parameter description. Version 1.7 introduces support for sending script parameters(commandToExecute) as protectedSettings, in which case they will be encrypted before sending. 'commandToExecute' parameter can be specified either in settings or protectedSettings but not in both.

```
{  
    "publisher": "Microsoft.Compute",  
    "type": "CustomScriptExtension",  
    "typeHandlerVersion": "1.7",  
    "settings": {  
        "fileUris": [  
            "http://Yourstorageaccount.blob.core.windows.net/customscriptfiles/start.ps1"  
        ],  
        "commandToExecute": "powershell.exe -ExecutionPolicy Unrestricted -start.ps1"  
    },  
    "protectedSettings": {  
        "commandToExecute": "powershell.exe -ExecutionPolicy Unrestricted -start.ps1",  
        "storageAccountName": "yourStorageAccountName",  
        "storageAccountKey": "yourStorageAccountKey"  
    }  
}
```

VMAccess Extension.

```
{
  "publisher": "Microsoft.Compute",
  "type": "VMAccessAgent",
  "typeHandlerVersion": "2.0",
  "settings": {
    "UserName" : "New User Name"
  },
  "protectedSettings": {
    "Password" : "New Password"
  }
}
```

DSC Extension.

```
{
  "publisher": "Microsoft.PowerShell",
  "type": "DSC",
  "typeHandlerVersion": "2.1(Recommendation is to use the latest version)",
  "settings": {
    "ModulesUrl": "https://UrlToZipContainingConfigurationScript.ps1.zip",
    "SasToken": "Optional : SAS Token if ModulesUrl points to Azure Blob Storage",
    "ConfigurationFunction": "ConfigurationScript.ps1\\ConfigurationFunction",
    "Properties": {
      "ParameterToConfigurationFunction1": "Value1",
      "ParameterToConfigurationFunction2": "Value2",
      "ParameterOfTypePSCredential1": {
        "UserName": "UsernameValue1",
        "Password": "PrivateSettingsRef:Key1(Value is a reference to a member of the Items object in the protected settings)"
      },
      "ParameterOfTypePSCredential2": {
        "UserName": "UsernameValue2",
        "Password": "PrivateSettingsRef:Key2"
      }
    }
  },
  "protectedSettings": {
    "Items": {
      "Key1": "PasswordValue1",
      "Key2": "PasswordValue2"
    },
    "DataBlobUri": "optional : https://UrlToConfigurationData.psd1"
  }
}
```

Symantec Endpoint Protection.

```
{
  "publisher": "SymantecEndpointProtection",
  "type": "Symantec",
  "typeHandlerVersion": "12.1",
  "settings": {}
}
```

Trend Micro Deep Security Agent.

```
{
  "publisher": "TrendMicro.DeepSecurity",
  "type": "TrendMicroDSA",
  "typeHandlerVersion": "9.6",
  "settings": {
    "ManagerAddress" : "Enter the externally accessible DNS name or IP address of the Deep Security Manager.  
Please enter \"agents.deepsecurity.trendmicro.com\" if using Deep Security as a Service",

    "ActivationPort" : "Enter the port number of the Deep Security Manager, default value - 443",

    "TenantIdentifier" : "Enter the tenant ID, which is a hyphenated, 36-character string available in the Deployment Scripts dialog box in the Deep Security console. This parameter is mandatory if using Deep Security as a Service, or a multi-tenant installation of Deep Security Manager. Type NA if using a non multi-tenant installation of Deep Security Manager.",

    "TenantActivationPassword" : "Enter the tenant activation password, which is a hyphenated, 36-character string available in the Deployment Scripts dialog box in the Deep Security console. This parameter is mandatory if using Deep Security as a Service, or a multi-tenant installation of Deep Security Manager. Type NA if using a non multi-tenant installation of Deep Security Manager.",

    "SecurityPolicy" : "Optional : Enter the name or numeric ID of the security policy defined in the Deep Security Manager which will be applied on agent activation to protect this virtual machine (recommended). No security policy will be applied to the virtual machine if this parameter is blank. This parameter is optional if using Deep Security as a Service."
  }
}
```

Vormetric Transparent Encryption Agent.

```
{
  "publisher": "Vormetric",
  "type": "VormetricTransparentEncryptionAgent",
  "typeHandlerVersion": "5.2",
  "settings": {
  }
}
```

Puppet Enterprise Agent.

```
{
  "publisher": "PuppetLabs",
  "type": "PuppetEnterpriseAgent",
  "typeHandlerVersion": "3.2",
  "settings": {
    "puppet_master_server" : "Puppet Master Server Name"
  }
}
```

Microsoft Monitoring Agent for Azure Operational Insights

```
{
  "publisher": "Microsoft.EnterpriseCloud.Monitoring",
  "type": "MicrosoftMonitoringAgent",
  "typeHandlerVersion": "1.0",
  "settings": {
    "workspaceId" : "The Workspace ID is available from within the Direct Agent Configuration section of the Azure Operational Insights portal"
  }
  "protectedSettings": {
    "workspaceKey" : "The Workspace Key is a string that is available from within the Direct Agent Configuration section of the Azure Operational Insights portal"
  }
}
```

McAfee EndpointSecurity

```
{
  "publisher": "McAfee.EndpointSecurity",
  "type": "McAfeeEndpointSecurity",
  "typeHandlerVersion": "6.0",
  "settings": {
    "entitlementKey" : "Optional : Enter a valid entitlement key or leave blank for trial version",
    "featureVS"      : "Choose whether or not to install the Virus and Spyware Protection features : true|false",
    "featureBP"      : "Choose whether or not to install the Browser Protection feature : true|false",
    "featureFW"      : "Choose whether or not to install the Firewall Protection feature :true|false",
    "relayServer"     : "Allows VMs on the local subnet to receive updates through this VM when they are not connected to the internet : true|false"
  }
}
```

Azure IaaS Antimalware

```
{
  "publisher": "Microsoft.Azure.Security",
  "type": "IaaSAntimalware",
  "typeHandlerVersion": "1.2",
  "settings": {
    "AntimalwareEnabled": "true",
    "ExclusionsPaths"      : "Optional : ExclusionsPaths",
    "ExclusionsExtensions" : "Optional : ExclusionsExtensions",
    "ExclusionsProcesses"   : "Optional : ExclusionsProcesses",
    "RealtimeProtectionEnabled" : "Optional : True|False",
    "ScheduledScanSettingsIsEnabled" : "Optional : True|False",
    "ScheduledScanSettingsScanType"   : "Optional : Quick|Full",
    "ScheduledScanSettingsDay"       : "Optional : Sunday-Saturday",
    "ScheduledScanSettingsTime"     : "Optional : When to perform the scheduled scan, measured in minutes from midnight,0-1440"
  }
}
```

ESET File Security

```
{
  "publisher": "ESET",
  "type": "FileSecurity",
  "typeHandlerVersion": "6.0",
  "settings": {
  }
}
```

Datadog Agent

```
{
  "publisher": "Datadog.Agent",
  "type": "DatadogWindowsAgent",
  "typeHandlerVersion": "0.5",
  "settings": {
    "api_key" : "API Key from https://app.datadoghq.com/account/settings#api"
  }
}
```

Confer Advanced Threat Prevention and Incident Response for Azure

```
{
  "publisher": "Confer",
  "type": "ConferForAzure",
  "typeHandlerVersion": "1.0",
  "settings": {
    "ConferRegisterCode" : "Optional : Valid product registration code or leave it blank to register later",
    "ConferRegisterCode" : "Enter a valid server name if your account requires a dedicated confer backend server or leave it blank"
  }
}
```

CloudLink SecureVM Agent

```
{
  "publisher": "CloudLinkEMC.SecureVM",
  "type": "CloudLinkSecureVMWindowsAgent",
  "typeHandlerVersion": "4.0",
  "settings": {
    "CloudLinkCenter" : "specify valid IP/FQDN to CloudLinkCenter"
  }
}
```

Barracuda VPN Connectivity Agent for Microsoft Azure

```
{
  "publisher": "Barracuda.Azure.ConnectivityAgent",
  "type": "BarracudaConnectivityAgent",
  "typeHandlerVersion": "3.5",
  "settings": {
    "ServerAddress" : "Host name or IP address of the VPN server - AES, AES256, Blowfish,CAST,DES,3DES,None",
    "EncryptionAlgorithm" : "Algorithm used to encrypt VPN traffic - MD5,SHA1,SHA256,None",
    "PKCS12File" : "Url for file containing certificate and private key used to authenticate against the VPN server",
    "PKCS12FilePassword" : "Password for the file containing certificate and private key"
  }
}
```

Alert Logic Log Manager

```
{
  "publisher": "AlertLogic.Extension",
  "type": "AlertLogicLM",
  "typeHandlerVersion": "1.9",
  "settings": {
    "registrationKey" : " Alert Logic Log Manager registration key"
  }
}
```

Chef Agent

```
{  
  "publisher": "Chef.Bootstrap.WindowsAzure",  
  "type": "ChefClient",  
  "typeHandlerVersion": "1210.12",  
  "settings": {  
    "validation_key" : " Validation key",  
    "client_rb" : "client_rb file",  
    "runlist" : "Optional runlist"  
  }  
}
```

Azure Diagnostics

For more details about how to configure diagnostics, see [Azure Diagnostics Extension](#)

```
{  
  "publisher": "Microsoft.Azure.Diagnostics",  
  "type": "IaaS.Diagnostics",  
  "typeHandlerVersion": "1.5",  
  "autoUpgradeMinorVersion": true,  
  "settings": {  
    "xmlCfg": "[base64(variables('wadcfgx'))]",  
    "storageAccount": "[parameters('diagnosticsStorageAccount')]"  
  },  
  "protectedSettings": {  
    "storageAccountName": "[parameters('diagnosticsStorageAccount')]",  
    "storageAccountKey": "[listkeys(variables('accountid'), '2015-05-01-preview').key1]",  
    "storageAccountEndPoint": "https://core.windows.net"  
  }  
}
```

In the examples above, replace the version number with the latest version number.

Here is an example of a full VM template with Custom Script Extension.

Custom Script Extension on a Windows VM

Troubleshooting Azure Windows VM extension failures

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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----          -----   -----
      9/1/2015  2:03 AM      11
      test.txt           \\\\n\\\\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 conditons, 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.

Managing Azure Virtual Machines using Azure Automation

1/17/2017 • 1 min to read • [Edit on GitHub](#)

This guide introduces you to the Azure Automation service and how it can be used to simplify managing your Azure virtual machines.

What is Azure Automation?

Azure Automation is an Azure service for simplifying cloud management through process automation. By using Azure Automation, long-running, manual, error-prone, and frequently repeated tasks can be automated to increase reliability, efficiency, and time-to-value for your organization.

Azure Automation provides a highly reliable and highly available workflow execution engine that scales to meet your needs as your organization grows. In Azure Automation, processes can be kicked off manually, by third-party systems, or at scheduled intervals so that tasks happen exactly when needed.

You can lower operational overhead and free up IT and DevOps staff to focus on work that adds business value by running your cloud management tasks automatically with Azure Automation.

How can Azure Automation help manage Azure virtual machines?

Virtual machines can be managed in Azure Automation by using [Azure PowerShell](#). Azure Automation includes the Azure PowerShell cmdlets, so you can perform all of your virtual machine management tasks within the service. You can also pair the cmdlets in Azure Automation with the cmdlets for other Azure services, to automate complex tasks across Azure services and third-party systems.

Next steps

Now that you've learned the basics of Azure Automation and how it can be used to manage Azure virtual machines, learn more:

- [Azure Automation Overview](#)
- [My first runbook](#)
- [Azure Automation learning map](#)

Vertically scale Azure virtual machines with Azure Automation

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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

Browse Gallery

 Filter

	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. **Note:** Premium database quota must be requested for your server via the Windows Azure Management Portal Tags: Capacity Management	Created by: Windows Azure Product Team Ratings: 0 of 5 1,122 downloads Last update: 10/16/2014
	Create Availability Group Listener in Windows Azure VMs (Cloud-Only) PowerShell Runbook This script configures an availability group listener for an availability group that is running in Windows Azure VMs. It automatically configures the Windows Azure settings and also configures each VM remotely using PowerShell remoting. Tags: Microsoft Azure , Powershell , High Availability	Created by: Cephas Lin Ratings: 5 of 5 1,813 downloads Last update: 11/22/2013
	Create Availability Group Listener in Hybrid IT PowerShell Runbook 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. Tags: Microsoft Azure , Powershell , High Availability	Created by: Cephas Lin Ratings: 0 of 5 1,095 downloads Last update: 11/22/2013
	Vertically scale up an Azure Resource Manager VM with Azure Automation PowerShell Runbook This Azure Automation runbook can help you vertically scale up an Azure Resource Manager VM in response to an alert. Tags:	Created by: Kay Singh Ratings: 0 of 5 0 downloads Last update: 3/29/2016
	Vertically scale down Azure Resource Manager VM with Azure Automation PowerShell Runbook This Azure Automation runbook can help you vertically scale down an Azure Resource Manager VM in response to an alert. Tags:	Created by: Kay Singh Ratings: 0 of 5 0 downloads Last update: 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.

armvm
Virtual machine

Settings Connect Start Restart Stop Delete

Essentials ^

Resource group
armrg

Status
Running

Location
Southeast Asia

Subscription name
Visual Studio Ultimate with MSDN

Subscription ID
<subscription-id>

Computer name
armvm

Operating system
Windows

Size
Basic A2 (2 cores, 3.5 GB memory)

Public IP address/DNS name label
-

Virtual network/subnet
armrg/default

1 All settings →

Monitoring Add tiles +

CPU percentage

100%

80%

60%

40%

20%

No available data.

0% Mar 29 6 AM 12 PM 6 PM

CPU PERCENTAGE GUEST...
0.41 %

Add a section +

Settings

armvm

Filter settings

SUPPORT + TROUBLESHOOTING

- Troubleshoot >
- Audit logs >
- Check health >
- Boot diagnostics >
- Reset password >
- Redeploy >
- New support request >

GENERAL

- Properties >
- Disks >
- Network interfaces >
- Availability set >
- Extensions >
- Size >

MONITORING

- 2 Alert rules >**
- Diagnostics >

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

Download the template for a VM

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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 hub menu, select **Virtual Machines**.
3. Select the virtual machine from the list.
4. Select **Automation script**.
5. Select **Download** and save the .zip file to your local computer.
6. Open the .zip file and extract the files to a folder. The .zip file will contain:
 - 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](#).

Automating Azure virtual machine deployment with Chef

1/17/2017 • 6 min to read • [Edit on GitHub](#)

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 our latest cloud-api release, Chef provides seamless integration with Azure, giving you the ability to provision and deploy configuration states through a single command.

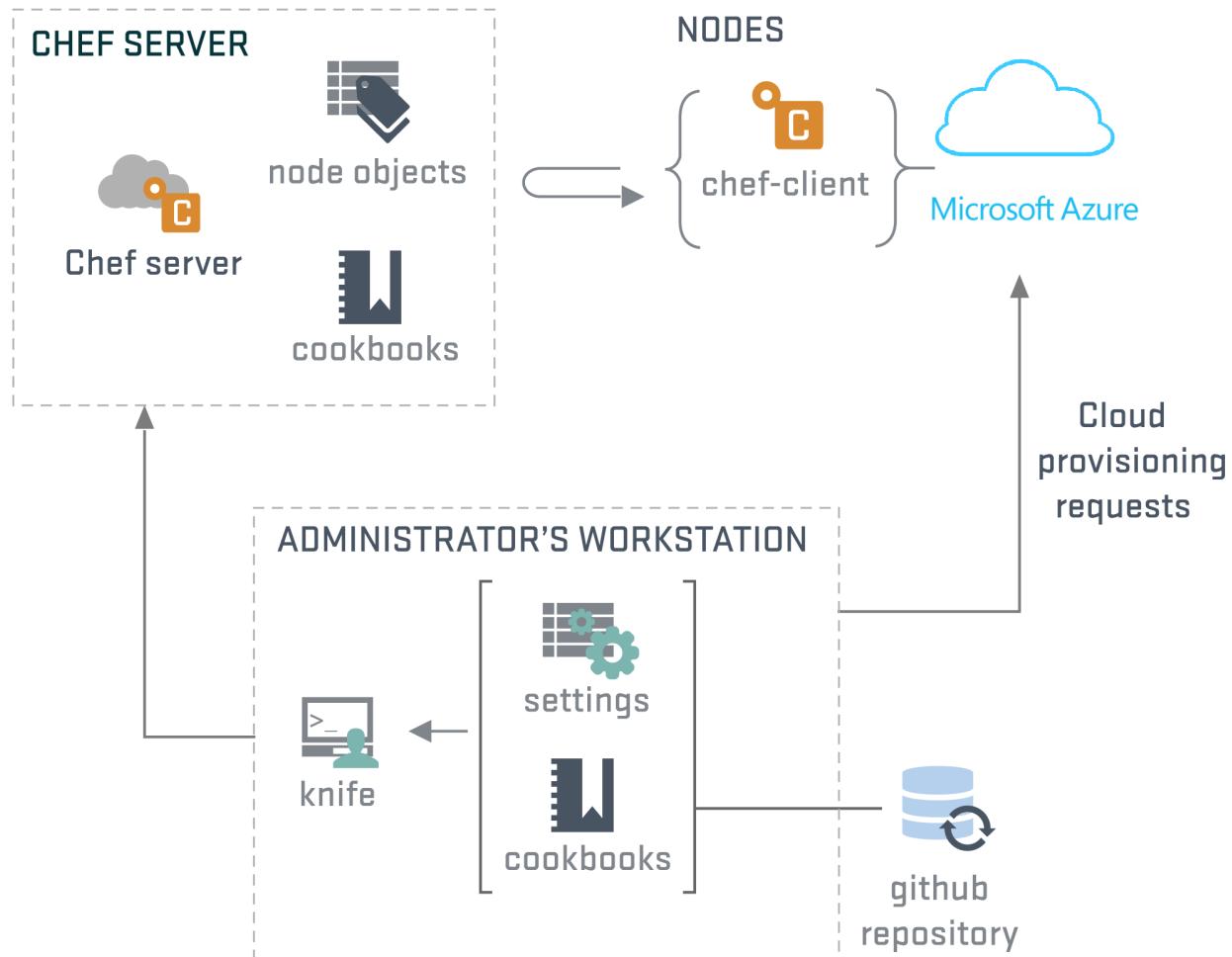
In this article, I'll show you how to set up your Chef environment to provision Azure virtual machines and walk you through creating a policy or "CookBook" and then deploying this cookbook to an Azure virtual machine.

Let's begin!

Chef basics

Before you begin, I suggest you review the basic concepts of Chef. There is great material [here](#) and I recommend you have a quick read before you attempt this walkthrough. I will however recap the basics before we get started.

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 our 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 our admin workstation where we create our policies and execute our management commands. We run the **knife** command from the Chef Workstation to manage our infrastructure.

There is also the concept of "Cookbooks" and "Recipes". These are effectively the policies we define and apply to our 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 our 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 our Azure settings file so Chef can communicate with our Azure subscription.

Download your publish settings from [here](#).

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 building 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 text 'Azure'. Below this is a field labeled 'Short Name (example: chef)'. A red error message 'Short name is required' is displayed below the short name field. At the bottom right of the dialog are two buttons: 'Cancel' and 'Create Organization' (which has a building icon).

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'. The 'Administration' tab is active. On the left, a sidebar menu lists 'Organizations' (with 'Create', 'Reset Validation Key', 'Generate Knife Config', 'Invite User', 'Leave Organization', and 'Starter Kit' options), 'Users', 'Groups', and 'Global Permissions'. 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'. Below this, a section titled 'What's next?' contains links to 'Chef Documentation' (described as 'The best place to start learning about Chef in general.') and 'Browse Community Cookbooks' (described as 'Hundreds of members of the Chef community have contributed cookbooks you can use or draw inspiration from.'). A note at the bottom 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.'

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.



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 our 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 our Chef client to execute on our 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 our 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 our 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 our 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 --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 'portalg' --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]'

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.

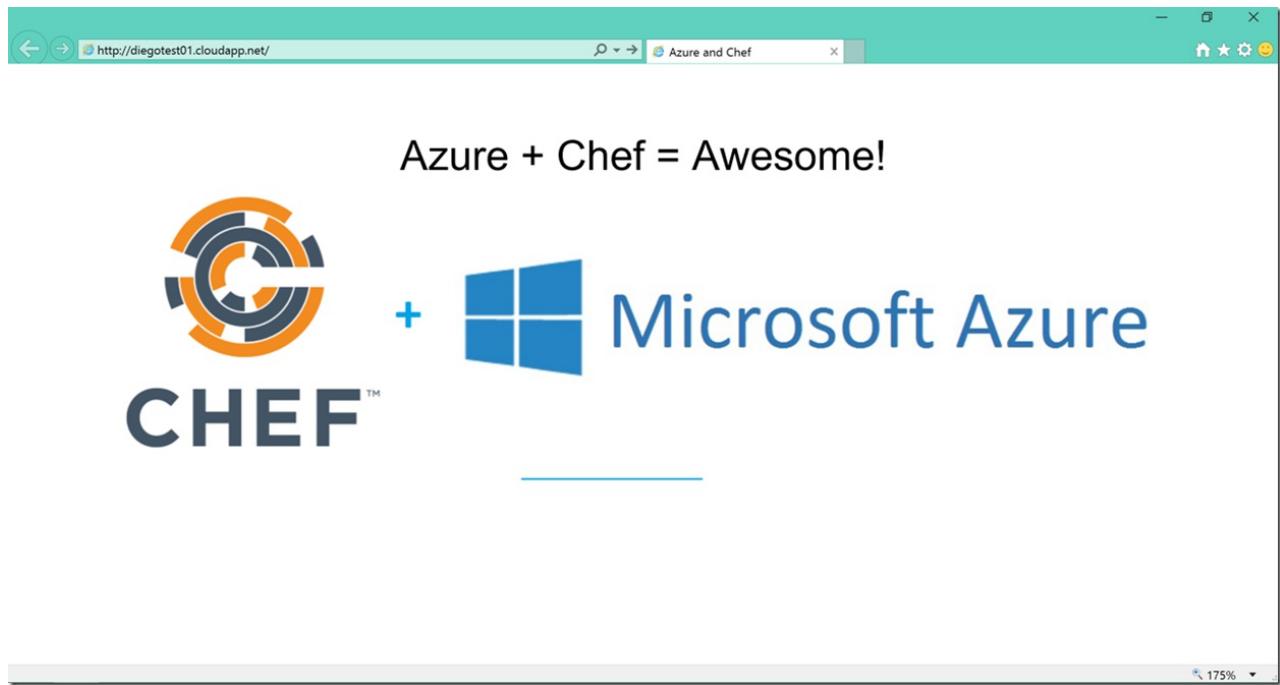
DNS Name: diegotest01.cloudapp.net
VM Name: testserver01
Size: Small
Azure Source Image: a699494373c04fc0bc8f2bb1389d6106_Windows-Server-2012-Datacenter-201411.01-en.us-127GB.vhd
Azure Service Location: Southeast Asia
Public Ip Address: 104.43.9.88
Private Ip Address: 100.72.52.27
WinRM Port: 5985
TCP Ports: [{"Name":>"tcpport_3389_testserver01", "Uip":>"104.43.9.88", "PublicPort":>"3389", "LocalPort":>"3389"}, {"Name":>"tcpport_80_testserver01", "Uip":>"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", "Uip":>"104.43.9.88", "PublicPort":>"3389", "LocalPort":>"3389"}, {"Name":>"tcpport_80_testserver01", "Uip":>"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 classic portal via port 3389.

I hope this has been helpful! Go and start your infrastructure as code journey with Azure today!

Platform-supported migration of IaaS resources from classic to Azure Resource Manager

1/17/2017 • 15 min to read • [Edit on GitHub](#)

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.

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 have to update your existing automation or tooling to ensure that it continues to work after the migration.

Meaning of migration of IaaS resources from classic to Resource Manager

Before we drill down into the details, let's look at the difference between data-plane and management-plane operations on the IaaS resources.

- *Management plane* describes the calls that come into the management 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 instances.
- *Data plane* (application) describes the runtime of the application itself and involves interaction with instances that don't go through the Azure API. Accessing your website or pulling data from a running SQL Server instance or a MongoDB server would be considered data plane or application interaction. Copying a blob from a storage account and accessing a public IP address to RDP or SSH into the virtual machine also are data plane. These operations keep the application running across compute, networking, and storage.

NOTE

In some migration scenarios, the Azure platform stops, deallocates, and restarts your virtual machines. This incurs a short data-plane downtime.

Supported scopes of migration

There are three migration scopes that primarily target compute, network, and storage.

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:

- 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.

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
Compute	Unassociated virtual machine disks.
Compute	Virtual machine images.
Network	Endpoint ACLs.
Network	Virtual network gateways (Azure ExpressRoute Gateways, Application gateway).
Network	Virtual networks using VNet Peering. (Migrate VNet to ARM, then peer) Learn more about VNet Peering .
Network	Traffic Manager profiles.

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.
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.

Service	Configuration	Recommendation
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.
Network	Virtual networks that contain virtual machines and web/worker roles	This is currently not supported.
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.
Compute	Azure Security Center extensions with a VNET that has a VPN gateway in transit connectivity or ExpressRoute gateway with on-prem DNS server	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. ExpressRoute gateway migration is not supported currently, and VPN gateways with transit connectivity loses on-premises access. Deleting ExpressRoute gateway or migrating VPN gateway with transit connectivity causes internet access to VM storage account to be lost when proceeding with committing the migration. The migration will not proceed when this happens as the guest agent status blob cannot be populated. It is recommended to disable Azure Security Center policy on the subscription 3 hours before proceeding with migration.

The migration experience

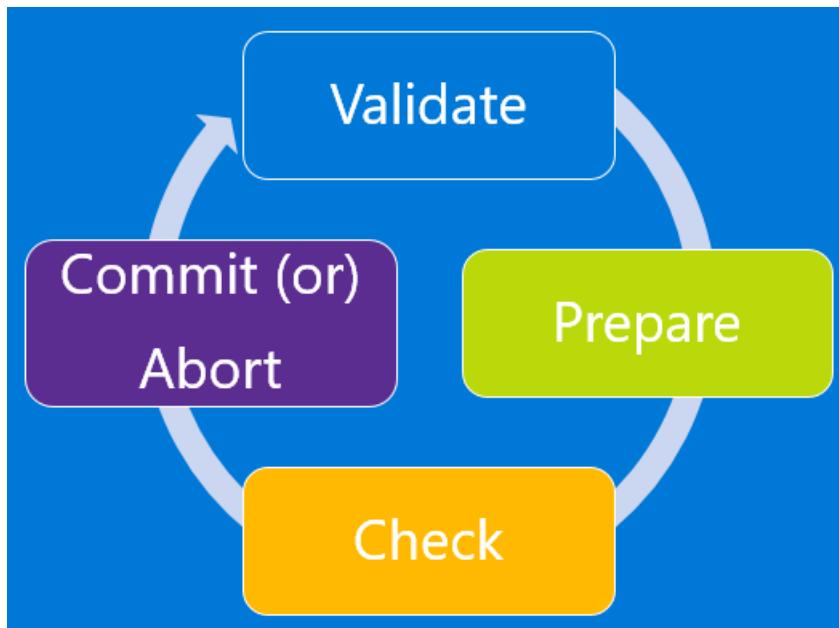
Before you start the migration experience, the following is recommended:

- 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 will be stopped and deallocated as part of the prepare operation. If you don't want to lose the public IP address, look into reserving the IP address before triggering the prepare operation. However, 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/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 RBAC policies that are configured on the classic IaaS resources, and plan for after the migration is complete.

The migration workflow is as follows



NOTE

All the operations described in the following sections are idempotent. If you have a problem other than an unsupported feature or a configuration error, it is recommended that you retry the prepare, abort, or commit operation. The Azure platform tries the action again.

Validate

The validate operation is the first step in the migration process. The goal of this step is to analyze data in the background for the resources under migration and return success/failure if the resources are capable of migration.

You select the virtual network or the hosted service (if it's not a virtual network) that you want to validate for migration.

- If the resource is not capable of migration, the Azure platform lists all the reasons for why it's not supported for migration.

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 classic to Resource Manager resources and present this side by side for you to visualize.

You select the virtual network or the hosted service (if it's not a virtual network) that you want to prepare for

migration.

- If the resource is not capable of migration, the Azure platform stops the migration process and lists the reason why the prepare operation failed.
- If the resource is capable of migration, the Azure platform first 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.

The Azure platform then starts the migration of metadata from classic to Resource Manager for the migrating resources.

After the prepare operation is complete, you have the option of visualizing the resources in both classic 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

Virtual Machines that are not in a classic Virtual Network are stopped deallocated in this phase of migration.

Check (manual or scripted)

In the check step, you can optionally 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 up and running.

You can test your monitoring/automation 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 time window 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, the Azure platform will open the management-plane operations on the resources so that you can resume normal operations on those VMs in the classic deployment model.

Abort

Abort is an optional step that you can use to revert your changes to the classic deployment model and stop the migration.

NOTE

This operation cannot be executed after you have triggered the commit operation.

Commit

After you finish the validation, you can commit the migration. Resources do not appear anymore in classic 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, it is recommended that you retry the operation. If it continues to fail, create a support ticket or create a forum post with a [ClassicaaSMigration](#) tag on our [VM forum](#).

Frequently asked questions

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.

What if I'm using Azure Site Recovery or Azure Backup today?

To migrate your Virtual Machine that are enabled for backup, see [I have backed up my classic VMs in backup vault. Now I want to migrate my VMs from classic mode to Resource Manager mode. How Can I backup them in recovery services vault?](#) i have backed up my classic VMs in backup vault. Now I want to migrate my VMs from classic mode to Resource Manager mode. How Can I backup them in 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](#).

I got a 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

Now that you understand the migration of classic IaaS resources to Resource Manager, you can start migrating resources.

- [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](#)
- [Clone a classic virtual machine to Azure Resource Manager by using community PowerShell scripts](#)
- [Review most common migration errors](#)

Technical deep dive on platform-supported migration from classic to Azure Resource Manager

1/17/2017 • 5 min to read • [Edit on GitHub](#)

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](#).

Detailed guidance on migration

You can find the classic and Resource Manager representations of the resources in the following table. Other features and resources are not currently supported.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	DETAILED 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.
Disk resources attached to VM	Implicit disks attached to VM	Disk resources 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 classic storage accounts, 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.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	DETAILED NOTES
Virtual machine certificates	Certificates in Azure Key Vault	<p>If a cloud service contains service certificates, 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.</p> <p>NOTE: Please do not delete the keyvault as it can cause the VM to go into a failed state. We're working on improving things in the backend so that Key Vaults can be deleted safely or moved along with the VM to a new subscription.</p>
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 was 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 in the Resource Manager deployment model, along with all the properties.
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.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	DETAILED NOTES
VIP address	Public IP address with DNS name	The virtual IP address becomes a public IP address and is associated with the load balancer.
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> . There are unsupported configurations .
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.
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.

CLASSIC REPRESENTATION	RESOURCE MANAGER REPRESENTATION	DETAILED NOTES
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 Local Network Gateway. This represents on-premises address prefixes and remote gateway IP.
Connections references	Connection	Connectivity references between gateway and local network site in network configuration is represented by a newly created resource called Connection in resource manager after migration. All properties of connectivity reference in network configuration files are copied unchanged to the newly created Connection resource. VNet to VNet connectivity in classic is achieved by creating two IPsec tunnels to local network sites representing the VNets. This is transformed to Vnet2Vnet connection type in resource manager model without requiring local network gateways.

Illustration of a simple migration walkthrough

The following screenshot shows an existing cloud service with a VM (not in a virtual network) after the preparation phase:

cloudguy-mig
Resource group

Resource Group containing classic resources in prepared state

Settings Add Delete

Subscription name: cloudguy-mig Subscription ID: [View](#)

Last deployment: No deployments Location: East Asia

All settings →

Summary Add tiles +

Resources

cloudguy-mig

cloudguy-mig

After the migration process has completed, the following screenshots show that the new resources have been created in a new resource group:

cloudguy-mig-Migrated
Resource group

Platform created resource group for migrated resources

Settings Add Delete

Subscription name: cloudguy-mig Subscription ID: [View](#)

Last deployment: 5/5/2016 (Succeeded) Location: East Asia

All settings →

Resources

cloudguy-mig

cloudguy-mig-PublicLoadBalancer

cloudguy-mig-PrimaryNic

cloudguy-mig-PrimaryVirtualIP

cloudguy-mig-VirtualNetwork

Next steps

Now that you understand the migration of classic IaaS resources to Resource Manager, you can start migrating resources.

- [Use PowerShell to migrate IaaS resources from classic to Azure Resource Manager](#)
- [Use CLI to migrate IaaS resources from classic to Azure Resource Manager](#)

- Platform-supported migration of IaaS resources from classic to Azure Resource Manager
- Clone a classic virtual machine to Azure Resource Manager by using community PowerShell scripts
- Review most common migration errors

Migrate IaaS resources from classic to Azure Resource Manager by using Azure PowerShell

1/17/2017 • 7 min to read • [Edit on GitHub](#)

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](#)

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

ExpressRoute gateways and Application Gateways are not currently supported for migration from classic to Resource Manager. To migrate a classic virtual network with a ExpressRoute or Application gateway, remove the gateway before running a Commit operation to move the network (you can run the Prepare step without deleting the ExpressRoute or Application gateway). After you complete the migration, reconnect the gateway in Azure Resource Manager.

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: 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 SubscriptionName | Select SubscriptionName
```

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 4: Make sure you have enough Azure Resource Manager Virtual Machine cores in the Azure region of your current deployment or VNET

You can use the following PowerShell command to check the current number of cores you have in Azure Resource Manager. To learn more about core 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 5: 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.

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 cloud service 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.

```
```powershell  
$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
```

Migrate virtual machines in a virtual network

To migrate virtual machines in a virtual network, you migrate the network. The virtual machines automatically migrate with the network. Pick the virtual network that you want to migrate.

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
```

Migrate a storage account

Once you're done migrating the virtual machines, we recommend you migrate the storage accounts.

Prepare 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 -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

- For more information about migration, see [Platform-supported migration of IaaS resources from classic to Azure Resource Manager](#).
- To migrate additional network resources to Resource Manager using Azure PowerShell, use similar steps with [Move-AzureNetworkSecurityGroup](#), [Move-AzureReservedIP](#), and [Move-AzureRouteTable](#).
- For open-source scripts you can use to migrate Azure resources from classic to Resource Manager, see [Community tools for migration to Azure Resource Manager migration](#)

Common errors during Classic to Azure Resource Manager migration

1/17/2017 • 7 min to read • [Edit on GitHub](#)

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 | 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.

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. |
| Migration is not supported for Deployment {deployment-name} in HostedService {hosted-service-name} because it is a PaaS deployment (Web/Worker). | 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. |
| Template {template-name} deployment failed. CorrelationId= {guid} | 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.

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. |
| The virtual network {virtual-network-name} does not exist. | This can happen if you created the Virtual Network in the new Azure portal. The actual Virtual Network name follows the pattern "Group * " |
| 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. | 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. |
| 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 | 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/vmbackupmigration |

| 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>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>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>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/classiciaasmigrationfaqs.</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. If you'd like to correct the data disk size before proceeding with migration, visit https://aka.ms/vmdiskresize.</p> | <p>This error happen if you've resized the VHD blob without updating the size in the VM API model. Detailed mitigation steps are outlined below.</p> |

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         :
```

```

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


```

Next steps

Here's a list of migration articles that explain the process.

- [Platform-supported 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 Azure Service Management to Azure Resource Manager migration

1/17/2017 • 1 min to read • [Edit on GitHub](#)

This article catalogs the tools that have been provided by the community to assist with migration of IaaS resources from Azure Service Management to the Azure Resource Manager stack.

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](#)

ASM2ARM

This is a PowerShell script module for migrating your **single** Virtual Machine (VM) from Azure Service Management stack to Azure Resource Manager stack.

[Link to the tool documentation](#)

migAz

migAz is an additional option to migrate a complete set of Azure Service Management 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](#)

Best Practices for running a Windows VM on Azure

1/17/2017 • 10 min to read • [Edit on GitHub](#)

patterns & practices

proven practices for predictable results

This article outlines a set of proven practices for running a Windows virtual machine (VM) on Azure, paying attention to scalability, availability, manageability, and security.

NOTE

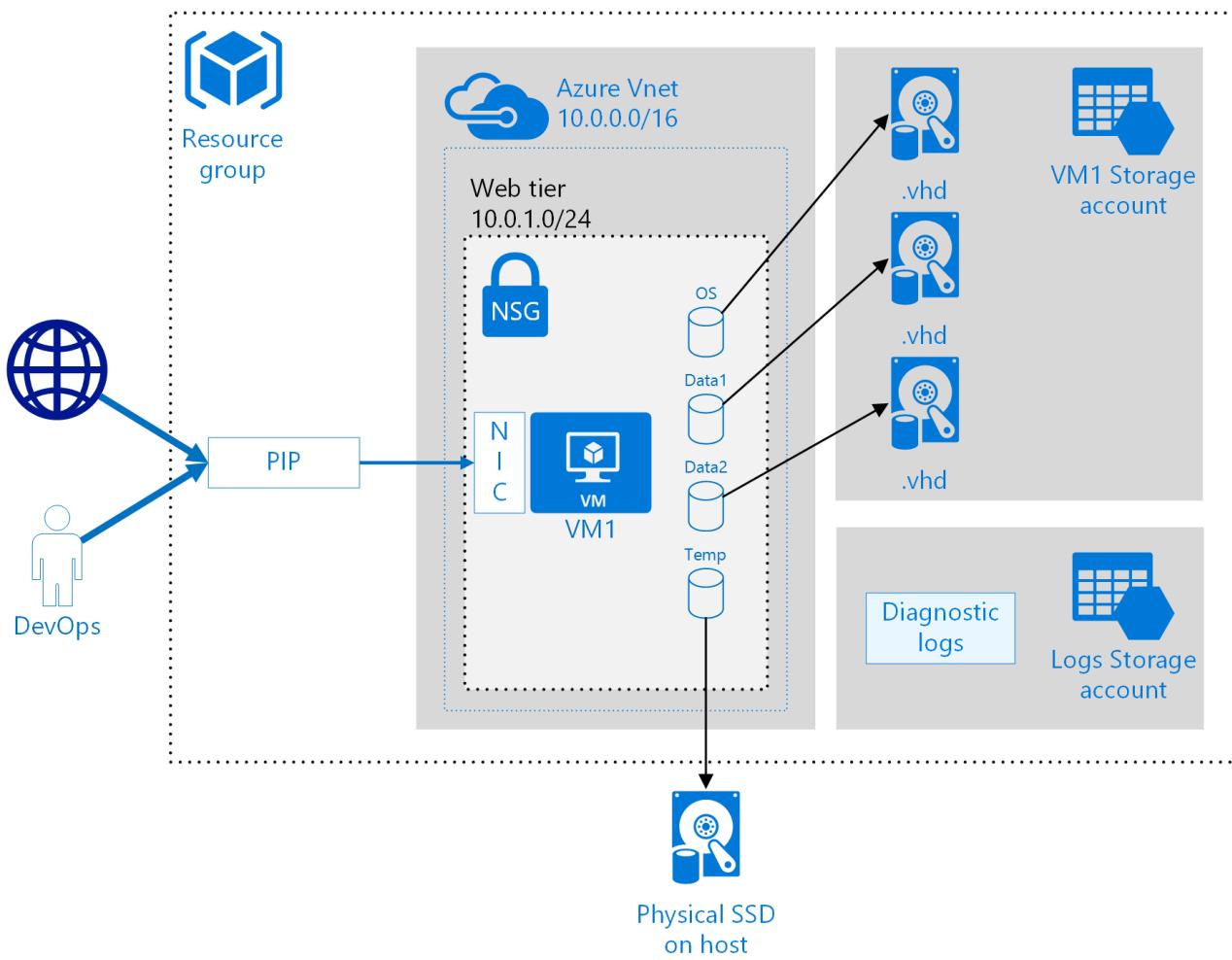
Azure has two different deployment models: [Azure Resource Manager](#) and classic. This article uses Resource Manager, which Microsoft recommends for new deployments.

We don't recommend using a single VM for mission critical workloads, because it creates a single point of failure. For higher availability, deploy multiple VMs in an [availability set](#). For more information, see [Running multiple VMs on Azure](#).

Architecture diagram

Provisioning a VM in Azure involves more moving parts than just the VM itself. There are compute, networking, and storage elements.

A Visio document that includes this architecture diagram is available for download from the [Microsoft download center](#). This diagram is on the "Compute - single VM" page.



- **Resource group.** A [resource group](#) is a container that holds related resources. Create a resource group to hold the resources for this VM.
- **VM.** You can provision a VM from a list of published images or from a virtual hard disk (VHD) file that you upload to Azure Blob storage.
- **OS disk.** The OS disk is a VHD stored in [Azure Storage](#). That means it persists even if the host machine goes down.
- **Temporary disk.** The VM is created with a temporary disk (the [D:](#) drive on Windows). This disk is stored on a physical drive on the host machine. It is *not* saved in Azure Storage, and might be deleted during reboots and other VM lifecycle events. Use this disk only for temporary data, such as page or swap files.
- **Data disks.** A [data disk](#) is a persistent VHD used for application data. Data disks are stored in Azure Storage, like the OS disk.
- **Virtual network (VNet) and subnet.** Every VM in Azure is deployed into a VNet that is further divided into subnets.
- **Public IP address.** A public IP address is needed to communicate with the VM—for example over remote desktop (RDP).
- **Network interface (NIC).** The NIC enables the VM to communicate with the virtual network.
- **Network security group (NSG).** The [NSG](#) is used to allow/deny network traffic to the subnet. You can associate an NSG with an individual NIC or with a subnet. If you associate it with a subnet, the NSG rules apply to all VMs in that subnet.
- **Diagnostics.** Diagnostic logging is crucial for managing and troubleshooting the VM.

Recommendations

The following recommendations apply for most scenarios. Follow these recommendations unless you have a specific requirement that overrides them.

VM recommendations

Azure offers many different virtual machine sizes, but we recommend the DS- and GS-series because these machine sizes support [Premium Storage](#). Select one of these machine sizes unless you have a specialized workload such as high-performance computing. For details, see [virtual machine sizes](#).

If you are moving an existing workload to Azure, start with the VM size that's the closest match to your on-premises servers. Then measure the performance of your actual workload with respect to CPU, memory, and disk input/output operations per second (IOPS), and adjust the size if needed. If you require multiple NICs for your VM, be aware that the maximum number of NICs is a function of the [VM size](#).

When you provision the VM and other resources, you must specify a region. Generally, choose a region closest to your internal users or customers. However, not all VM sizes may be available in all regions. For details, see [services by region](#). To see a list of the VM sizes available in a given region, run the following Azure command-line interface (CLI) command:

```
azure vm sizes --location <location>
```

For information about choosing a published VM image, see [Navigate and select Windows virtual machine images in Azure with Powershell or CLI](#).

Disk and storage recommendations

For best disk I/O performance, we recommend [Premium Storage](#), which stores data on solid state drives (SSDs). Cost is based on the size of the provisioned disk. IOPS and throughput also depend on disk size, so when you provision a disk, consider all three factors (capacity, IOPS, and throughput).

Create separate Azure storage accounts for each VM to hold the virtual hard disks (VHDs) in order to avoid hitting the IOPS limits for storage accounts.

Add one or more data disks. When you create a new VHD, it is unformatted. Log into the VM to format the disk. If you have a large number of data disks, be aware of the total I/O limits of the storage account. For more information, see [virtual machine disk limits](#).

When possible, install applications on a data disk, not the OS disk. However, some legacy applications might need to install components on the C: drive. In that case, you can [resize the OS disk](#) using PowerShell.

For best performance, create a separate storage account to hold diagnostic logs. A standard locally redundant storage (LRS) account is sufficient for diagnostic logs.

Network recommendations

The public IP address can be dynamic or static. The default is dynamic.

- Reserve a [static IP address](#) if you need a fixed IP address that won't change — for example, if you need to create an A record in DNS, or need the IP address to be added to a safe list.
- You can also create a fully qualified domain name (FQDN) for the IP address. You can then register a [CNAME record](#) in DNS that points to the FQDN. For more information, see [create a fully qualified domain name in the Azure portal](#).

All NSGs contain a set of [default rules](#), including a rule that blocks all inbound Internet traffic. The default rules cannot be deleted, but other rules can override them. To enable Internet traffic, create rules that allow inbound traffic to specific ports — for example, port 80 for HTTP.

To enable RDP, add an NSG rule that allows inbound traffic to TCP port 3389.

Scalability considerations

You can scale a VM up or down by [changing the VM size](#). To scale out horizontally, put two or more VMs into an

availability set behind a load balancer. For details, see [Running multiple VMs on Azure for scalability and availability](#).

Availability considerations

For higher availability, deploy multiple VMs in an availability set. This also provides a higher [service level agreement](#) (SLA).

Your VM may be affected by [planned maintenance](#) or [unplanned maintenance](#). You can use [VM reboot logs](#) to determine whether a VM reboot was caused by planned maintenance.

VHDs are stored in [Azure storage](#), and Azure storage is replicated for durability and availability.

To protect against accidental data loss during normal operations (for example, because of user error), you should also implement point-in-time backups, using [blob snapshots](#) or another tool.

Manageability considerations

Resource groups. Put tightly-coupled resources that share the same life cycle into the same [resource group](#).

Resource groups allow you to deploy and monitor resources as a group and roll up billing costs by resource group. You can also delete resources as a set, which is very useful for test deployments. Give resources meaningful names. That makes it easier to locate a specific resource and understand its role. See [Recommended Naming Conventions for Azure Resources](#).

VM diagnostics. Enable monitoring and diagnostics, including basic health metrics, diagnostics infrastructure logs, and [boot diagnostics](#). Boot diagnostics can help you diagnose a boot failure if your VM gets into a nonbootable state. For more information, see [Enable monitoring and diagnostics](#). Use the [Azure Log Collection](#) extension to collect Azure platform logs and upload them to Azure storage.

The following CLI command enables diagnostics:

```
azure vm enable-diag <resource-group> <vm-name>
```

Stopping a VM. Azure makes a distinction between "stopped" and "deallocated" states. You are charged when the VM status is stopped, but not when the VM is deallocated.

Use the following CLI command to deallocate a VM:

```
azure vm deallocate <resource-group> <vm-name>
```

In the Azure portal, the **Stop** button deallocates the VM. However, if you shut down through the OS while logged in, the VM is stopped but *not* deallocated, so you will still be charged.

Deleting a VM. If you delete a VM, the VHDs are not deleted. That means you can safely delete the VM without losing data. However, you will still be charged for storage. To delete the VHD, delete the file from [Blob storage](#).

To prevent accidental deletion, use a [resource lock](#) to lock the entire resource group or lock individual resources, such as the VM.

Security considerations

Use [Azure Security Center](#) to get a central view of the security state of your Azure resources. Security Center monitors potential security issues and provides a comprehensive picture of the security health of your deployment. Security Center is configured per Azure subscription. Enable security data collection as described in [Use Security Center](#). When data collection is enabled, Security Center automatically scans any VMs created under that subscription.

Patch management. If enabled, Security Center checks whether security and critical updates are missing. Use [Group Policy settings](#) on the VM to enable automatic system updates.

Antimalware. If enabled, Security Center checks whether antimalware software is installed. You can also use Security Center to install antimalware software from inside the Azure portal.

Operations. Use [role-based access control](#) (RBAC) to control access to the Azure resources that you deploy. RBAC lets you assign authorization roles to members of your DevOps team. For example, the Reader role can view Azure resources but not create, manage, or delete them. Some roles are specific to particular Azure resource types. For example, the Virtual Machine Contributor role can restart or deallocate a VM, reset the administrator password, create a new VM, and so forth. Other [built-in RBAC roles](#) that might be useful for this reference architecture include [DevTest Labs User](#) and [Network Contributor](#). A user can be assigned to multiple roles, and you can create custom roles for even more fine-grained permissions.

NOTE

RBAC does not limit the actions that a user logged into a VM can perform. Those permissions are determined by the account type on the guest OS.

To reset the local administrator password, run the `vm reset-access` Azure CLI command.

```
azure vm reset-access -u <user> -p <new-password> <resource-group> <vm-name>
```

Use [audit logs](#) to see provisioning actions and other VM events.

Data encryption. Consider [Azure Disk Encryption](#) if you need to encrypt the OS and data disks.

Solution deployment

A deployment for this reference architecture is available on [GitHub](#). It includes a VNet, NSG, and a single VM. To deploy the architecture, follow these steps:

1. Right click the button below and select either "Open link in new tab" or "Open link in new window."



2. Once the link has opened in the Azure portal, you must enter values for some of the settings:

- The **Resource group** name is already defined in the parameter file, so select **Create New** and enter `ra-single-vm-rg` in the text box.
 - Select the region from the **Location** drop down box.
 - Do not edit the **Template Root Uri** or the **Parameter Root Uri** text boxes.
 - Select **windows** in the **Os Type** drop down box.
 - Review the terms and conditions, then click the **I agree to the terms and conditions stated above** checkbox.
 - Click on the **Purchase** button.
3. Wait for the deployment to complete.
 4. The parameter files include a hard-coded administrator user name and password, and it is strongly recommended that you immediately change both. Click on the VM named `ra-single-vm0` in the Azure portal. Then, click on **Reset password** in the **Support + troubleshooting** blade. Select **Reset password** in the **Mode** dropdown box, then select a new **User name** and **Password**. Click the **Update** button to persist the new user name and password.

For information on additional ways to deploy this reference architecture, see the readme file in the [guidance-single-vm](#) Github folder.

Customize the deployment

If you need to change the deployment to match your needs, follow the instructions in the [readme](#).

Next steps

For higher availability, deploy two or more VMs behind a load balancer. For more information, see [Running multiple VMs on Azure](#).

Azure virtual machines guidelines for Windows

1/17/2017 • 2 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding the required planning steps for creating and managing virtual machines (VMs) within your Azure environment.

Implementation guidelines for VMs

Decisions:

- How many VMs do you require for your various application tiers and components of your infrastructure?
- What CPU and memory resources does each VM need, and what are the storage requirements?

Tasks:

- Define the workloads for your application and the resources the VMs require.
- Align the resource demands for each VM with the appropriate VM size and storage type.
- Define your resource groups for the different tiers and components of your infrastructure.
- Define your VM naming convention.
- Create your VMs using the Azure PowerShell, web portal, or with Resource Manager templates.

Virtual machines

One of the main resources within your Azure environment is likely VMs. This resource is where you run your applications, databases, authentication services, etc.

It is important to understand the [different VM sizes](#) to correctly size your environment from a performance and cost perspective. If your VMs do not have enough CPU cores or memory, performance of your application suffers regardless of how well it is designed and developed. Review the suggested workloads for each VM series as a starting point as you decide which size VM to use for each component in your infrastructure. You can [change the size of a VM after deployment](#).

Storage plays a key role in VM performance. You can use Standard storage, that uses regular spinning disks, or Premium storage for high I/O workloads and peak performance, that uses SSD disks. As with the VM size, there are cost considerations to selecting the storage medium. You can read the [storage infrastructure guidelines](#) article to understand how to design appropriate storage for optimum performance of your VMs.

Resource groups

Components such as VMs are logically grouped for ease of management and maintenance using [Azure Resource Groups](#). By using resource groups, you can create, manage, and monitor all the resources that make up a given application. You can also implement [role-based access controls](#) to grant access to others within your team to only the resources they require. Take time to plan out your resource groups and role assignments. There are different approaches to actually design and implement resource groups, so be sure to read the [resource groups guidelines article](#) to understand how best to build out your VMs.

Templates

You can build templates, defined by declarative JSON files, to create your VMs. Templates typically also build the required storage, networking, network interfaces, IP addressing, etc. along with the VMs themselves. You use templates to create consistent, reproducible environments for development and testing purposes to easily replicate production environments and vice versa. You can read more about [building and using templates](#) to understand how you can use them for creating and deploying your VMs.

Next steps

The full series of guidelines helps you understand all the different design considerations for building our your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Azure subscription and accounts guidelines

1/17/2017 • 2 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding how to approach subscription and account management as your environment and user base grows.

Implementation guidelines for subscriptions and accounts

Decisions:

- What set of subscriptions and accounts do you need to host your IT workload or infrastructure?
- How to break down the hierarchy to fit your organization?

Tasks:

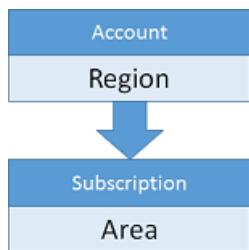
- Define your logical organization hierarchy as you would like to manage it from a subscription level.
- To match this logical hierarchy, define the accounts required and subscriptions under each account.
- Create the set of subscriptions and accounts using your naming convention.

Subscriptions and accounts

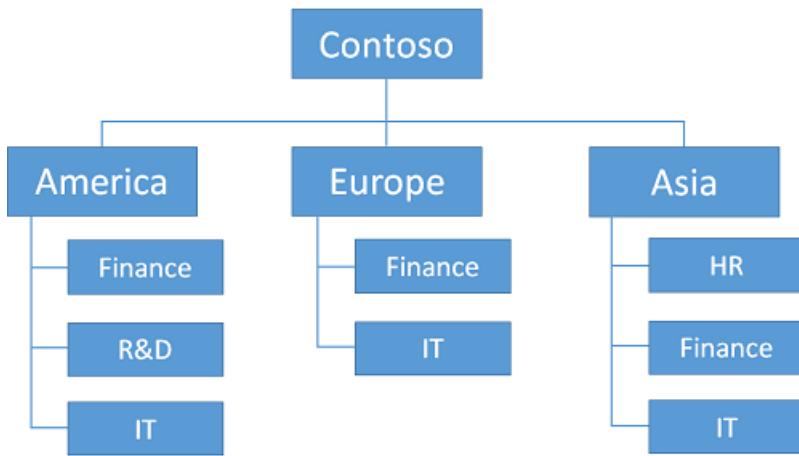
To work with Azure, you need one or more Azure subscriptions. Resources like virtual machines (VMs) or virtual networks exist in of those subscriptions.

- Enterprise customers typically have an Enterprise Enrollment, which is the top-most resource in the hierarchy, and is associated to one or more accounts.
- For consumers and customers without an Enterprise Enrollment, the top-most resource is the account.
- Subscriptions are associated to accounts, and there can be one or more subscriptions per account. Azure records billing information at the subscription level.

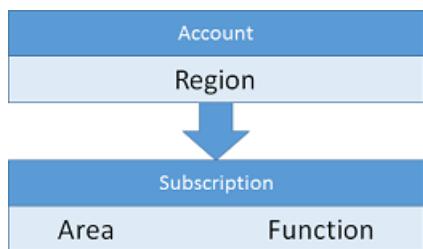
Due to the limit of two hierarchy levels on the Account/Subscription relationship, it is important to align the naming convention of accounts and subscriptions to the billing needs. For instance, if a global company uses Azure, they might choose to have one account per region, and have subscriptions managed at the region level:



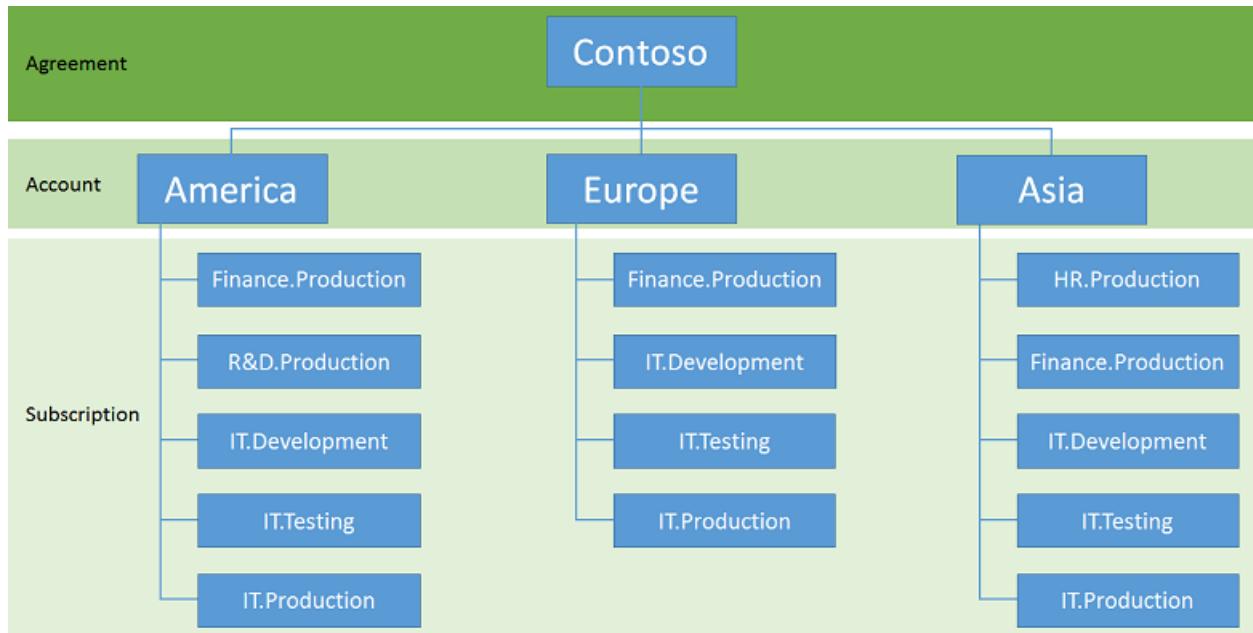
For instance, you might use the following structure:



If a region decides to have more than one subscription associated to a particular group, the naming convention should incorporate a way to encode the extra data on either the account or the subscription name. This organization allows massaging billing data to generate the new levels of hierarchy during billing reports:



The organization could look like the following example:



We provide detailed billing via a downloadable file for a single account, or for all accounts in an enterprise agreement.

Next steps

The full series of guidelines helps you understand all the different design considerations for building your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)

- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Azure infrastructure naming guidelines

1/17/2017 • 4 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding how to approach naming conventions for all your various Azure resources to build a logical and easily identifiable set of resources across your environment.

Implementation guidelines for naming conventions

Decisions:

- What are your naming conventions for Azure resources?

Tasks:

- Define the affixes to use across your resources to maintain consistency.
- Define storage account names given the requirement for them to be globally unique.
- Document the naming convention to be used and distribute to all parties involved to ensure consistency across deployments.

Naming conventions

You should have a good naming convention in place before creating anything in Azure. A naming convention ensures that all the resources have a predictable name, which helps lower the administrative burden associated with managing those resources.

You might choose to follow a specific set of naming conventions defined for your entire organization or for a specific Azure subscription or account. Although it is easy for individuals within organizations to establish implicit rules when working with Azure resources, when a team needs to work on a project on Azure, that model does not scale well.

Agree on a set of naming conventions up front. There are some considerations regarding naming conventions that cut across this sets of rules.

Affixes

As you look to define a naming convention, one decision comes whether the affix is at:

- The beginning of the name (prefix)
- The end of the name (suffix)

For instance, here are two possible names for a Resource Group using the `rg` affix:

- Rg-WebApp (prefix)
- WebApp-Rg (suffix)

Affixes can refer to different aspects that describe the particular resources. The following table shows some examples typically used.

| Aspect | Examples | Notes |
|--------------------------------------|---|--|
| Environment | dev, stg, prod | Depending on the purpose and name of each environment. |
| Location | usw (West US), use (East US 2) | Depending on the region of the datacenter or the region of the organization. |
| Azure component, service, or product | Rg for resource group, VNet for virtual network | Depending on the product for which the resource provides support. |
| Role | sql, ora, sp, iis | Depending on the role of the virtual machine. |
| Instance | 01, 02, 03, etc. | For resources that have more than one instance. For example, load balanced web servers in a cloud service. |

When establishing your naming conventions, make sure that they clearly state which affixes to use for each type of resource, and in which position (prefix vs suffix).

Dates

It is often important to determine the date of creation from the name of a resource. We recommend the YYYYMMDD date format. This format ensures that not only the full date is recorded, but also that two resources whose names differ only on the date is sorted alphabetically and chronologically at the same time.

Naming resources

Define each type of resource in the naming convention, which should have rules that define how to assign names to each resource that is created. These rules should apply to all types of resources, for example:

- Subscriptions
- Accounts
- Storage accounts
- Virtual networks
- Subnets
- Availability sets
- Resource groups
- Virtual machines
- Endpoints
- Network security groups
- Roles

To ensure that the name provides enough information to determine to which resource it refers, you should use descriptive names.

Computer names

When you create a virtual machine (VM), Microsoft Azure requires a VM name of up to 15 characters which is used for the resource name. Azure uses the same name for the operating system installed in the VM. However, these names might not always be the same.

In case a VM is created from a .vhdx image file that already contains an operating system, the VM name in Azure can differ from the VM's operating system computer name. This situation can add a degree of difficulty to VM management, which we therefore do not recommend. Assign the Azure VM resource the same name as the computer name that you assign to the operating system of that VM.

We recommend that the Azure VM name is the same as the underlying operating system computer name.

Storage account names

Storage accounts have special rules governing their names. You can only use lowercase letters and numbers. For more information, see [Create a storage account](#). Additionally, the storage account name, along with core.windows.net, should be a globally valid, unique DNS name. For instance, if the storage account is called mystorageaccount, the following resulting DNS names should be unique:

- mystorageaccount.blob.core.windows.net
- mystorageaccount.table.core.windows.net
- mystorageaccount.queue.core.windows.net

Next steps

The full series of guidelines helps you understand all the different design considerations for building our your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Azure resource group guidelines

1/17/2017 • 3 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding how to logically build out your environment and group all the components in Resource Groups.

Implementation guidelines for Resource Groups

Decisions:

- Are you going to build out Resource Groups by the core infrastructure components, or by complete application deployment?
- Do you need to restrict access to Resource Groups using Role-Based Access Controls?

Tasks:

- Define what core infrastructure components and dedicated Resource Groups you need.
- Review how to implement Resource Manager templates for consistent, reproducible deployments.
- Define what user access roles you need for controlling access to Resource Groups.
- Create the set of Resource Groups using your naming convention. You can use Azure PowerShell or the portal.

Resource Groups

In Azure, you logically group related resources such as storage accounts, virtual networks, and virtual machines (VMs) to deploy, manage, and maintain them as a single entity. This approach makes it easier to deploy applications while keeping all the related resources together from a management perspective, or to grant others access to that group of resources. For a more comprehensive understanding of Resource Groups, read the [Azure Resource Manager overview](#).

A key feature to Resource Groups is ability to build out your environment using templates. A template is simply a JSON file that declares the storage, networking, and compute resources. You can also define any related custom scripts or configurations to apply. By using these templates, you create consistent, reproducible deployments for your applications. This approach makes it easy to build out an environment in development and then use that same template to create a production deployment, or vice versa. For a better understanding using templates, read the [template walkthrough](#) that guides you through each step of the building out a template.

There are two different approaches you can take when designing your environment with Resource Groups:

- Resource Groups for each application deployment that combines the storage accounts, virtual networks, and subnets, VMs, load balancers, etc.
- Centralized Resource Groups that contain your core virtual networking and subnets or storage accounts. Your applications are then in their own Resource Groups that only contain VMs, load balancers, network interfaces, etc.

As you scale out, creating centralized Resource Groups for your virtual networking and subnets makes it easier to build cross-premises network connections for hybrid connectivity options. The alternative approach is for each application to have their own virtual network that requires configuration and maintenance. [Role-Based Access](#)

Controls provide a granular way to control access to Resource Groups. For production applications, you can control the users that may access those resources, or for the core infrastructure resources you can limit only infrastructure engineers to work with them. Your application owners only have access to the application components within their Resource Group and not the core Azure infrastructure of your environment. As you design your environment, consider the users that require access to the resources and design your Resource Groups accordingly.

Next steps

The full series of guidelines helps you understand all the different design considerations for building our your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Azure storage infrastructure guidelines

1/17/2017 • 4 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding storage needs and design considerations for achieving optimum virtual machine (VM) performance.

Implementation guidelines for storage

Decisions:

- Do you need to use Standard or Premium storage for your workload?
- Do you need disk striping to create disks larger than 1023 GB?
- Do you need disk striping to achieve optimal I/O performance for your workload?
- What set of storage accounts do you need to host your IT workload or infrastructure?

Tasks:

- Review I/O demands of the applications you are deploying and plan the appropriate number and type of storage accounts.
- Create the set of storage accounts using your naming convention. You can use Azure PowerShell or the portal.

Storage

Azure Storage is a key part of deploying and managing virtual machines (VMs) and applications. Azure Storage provides services for storing file data, unstructured data, and messages, and it is also part of the infrastructure supporting VMs.

There are two types of storage accounts available for supporting VMs:

- Standard storage accounts give you access to blob storage (used for storing Azure VM disks), table storage, queue storage, and file storage.
- [Premium storage](#) accounts deliver high-performance, low-latency disk support for I/O intensive workloads, such as SQL Servers in an AlwaysOn cluster. Premium storage currently supports Azure VM disks only.

Azure creates VMs with an operating system disk, a temporary disk, and zero or more optional data disks. The operating system disk and data disks are Azure page blobs, whereas the temporary disk is stored locally on the node where the machine lives. Take care when designing applications to only use this temporary disk for non-persistent data as the VM may be migrated between hosts during a maintenance event. Any data stored on the temporary disk would be lost.

Durability and high availability is provided by the underlying Azure Storage environment to ensure that your data remains protected against unplanned maintenance or hardware failures. As you design your Azure Storage environment, you can choose to replicate VM storage:

- locally within a given Azure datacenter
- across Azure datacenters within a given region
- across Azure datacenters across different regions

You can read [more about the replication options for high availability](#).

Operating system disks and data disks have a maximum size of 1023 gigabytes (GB). The maximum size of a blob is 1024 GB and that must contain the metadata (footer) of the VHD file (a GB is 1024^3 bytes). You can use Storage Spaces in Windows Server 2012 to surpass this limit by pooling together data disks to present logical volumes larger than 1023GB to your VM.

There are some scalability limits when designing your Azure Storage deployments - for more information, see [Microsoft Azure subscription and service limits, quotas, and constraints](#). Also see [Azure storage scalability and performance targets](#).

For application storage, you can store unstructured object data such as documents, images, backups, configuration data, logs, etc. using blob storage. Rather than your application writing to a virtual disk attached to the VM, the application can write directly to Azure blob storage. Blob storage also provides the option of [hot and cool storage tiers](#) depending on your availability needs and cost constraints.

Striped disks

Besides allowing you to create disks larger than 1023 GB, in many instances, using striping for data disks enhances performance by allowing multiple blobs to back the storage for a single volume. With striping, the I/O required to write and read data from a single logical disk proceeds in parallel.

Azure imposes limits on the number of data disks and amount of bandwidth available, depending on the VM size. For details, see [Sizes for virtual machines](#).

If you are using disk striping for Azure data disks, consider the following guidelines:

- Data disks should always be the maximum size (1023 GB).
- Attach the maximum data disks allowed for the VM size.
- Use Storage Spaces.
- Avoid using Azure data disk caching options (caching policy = None).

For more information, see [Storage spaces - designing for performance](#).

Multiple storage accounts

When designing your Azure Storage environment, you can use multiple storage accounts as the number of VMs you deploy increases. This approach helps distribute out the I/O across the underlying Azure Storage infrastructure to maintain optimum performance for your VMs and applications. As you design the applications that you are deploying, consider the I/O requirements each VM has and balance out those VMs across Azure Storage accounts. Try to avoid grouping all the high I/O demanding VMs in to just one or two storage accounts.

For more information about the I/O capabilities of the different Azure Storage options and some recommended maximums, see [Azure storage scalability and performance targets](#).

Next steps

The full series of guidelines helps you understand all the different design considerations for building our your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)

- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Azure networking infrastructure guidelines

1/17/2017 • 5 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding the required planning steps for virtual networking within Azure and connectivity between existing on-prem environments.

Implementation guidelines for virtual networks

Decisions:

- What type of virtual network do you need to host your IT workload or infrastructure (cloud-only or cross-premises)?
- For cross-premises virtual networks, how much address space do you need to host the subnets and VMs now and for reasonable expansion in the future?
- Are you going to create centralized virtual networks or create individual virtual networks for each resource group?

Tasks:

- Define the address space for the virtual networks to be created.
- Define the set of subnets and the address space for each.
- For cross-premises virtual networks, define the set of local network address spaces for the on-premises locations that the VMs in the virtual network need to reach.
- Work with on-premises networking team to ensure the appropriate routing is configured when creating cross-premises virtual networks.
- Create the virtual network using your naming convention.

Virtual networks

Virtual networks are necessary to support communications between virtual machines (VMs). You can define subnets, custom IP address, DNS settings, security filtering, and load balancing, as with physical networks. By using a [VPN gateway](#) or [Express Route circuit](#), you can connect Azure virtual networks to your on-premises networks. You can read more about [virtual networks and their components](#).

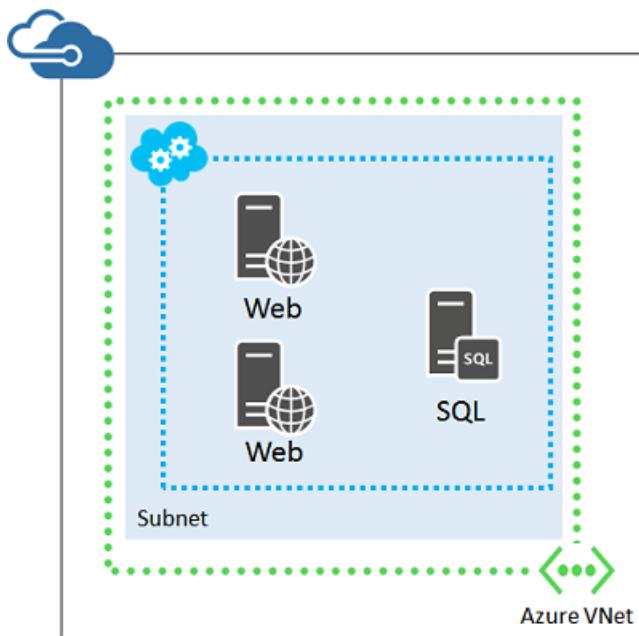
By using Resource Groups, you have flexibility in how you design your virtual networking components. VMs can connect to virtual networks outside of their own resource group. A common design approach would be to create centralized resource groups that contain your core networking infrastructure that can be managed by a common team, and then VMs and their applications deployed to separate resource groups. This approach allows application owners access to the resource group that contains their VMs without opening up access to the configuration of the wider virtual networking resources.

Site connectivity

Cloud-only virtual networks

If on-premises users and computers do not require ongoing connectivity to VMs in an Azure virtual network, your

virtual network design is straight forward:

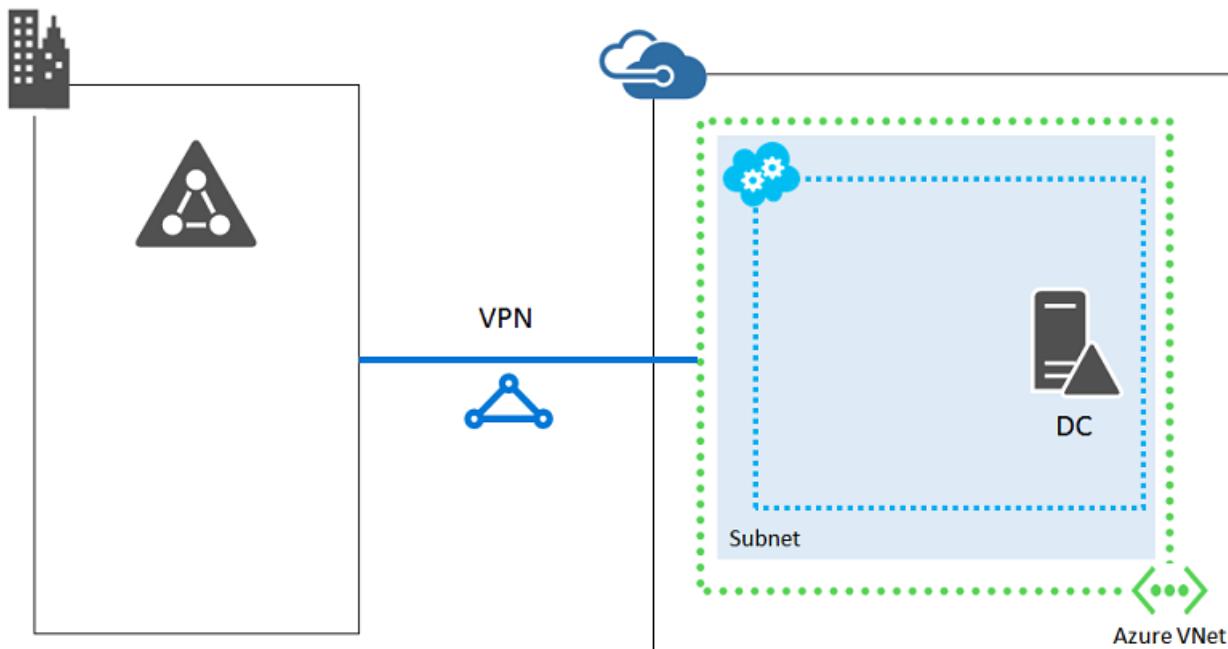


This approach is typically for Internet-facing workloads, such as an Internet-based web server. You can manage these VMs using RDP or point-to-site VPN connections.

Because they do not connect to your on-premises network, Azure-only virtual networks can use any portion of the private IP address space, even if the same private space is in use on-premises.

Cross-premises virtual networks

If on-premises users and computers require ongoing connectivity to VMs in an Azure virtual network, create a cross-premises virtual network. Connect it to your on-premises network with an ExpressRoute or site-to-site VPN connection.



In this configuration, the Azure virtual network is essentially a cloud-based extension of your on-premises network.

Because they connect to your on-premises network, cross-premises virtual networks must use a portion of the address space used by your organization that is unique. In the same way that different corporate locations are assigned a specific IP subnet, Azure becomes another location as you extend out your network.

To allow packets to travel from your cross-premises virtual network to your on-premises network, you must configure the set of relevant on-premises address prefixes as part of the local network definition for the virtual network. Depending on the address space of the virtual network and the set of relevant on-premises locations, there can be many address prefixes in the local network.

You can convert a cloud-only virtual network to a cross-premises virtual network, but it most likely requires you to re-IP your virtual network address space and Azure resources. Therefore, carefully consider if a virtual network needs to be connected to your on-premises network when you assign an IP subnet.

Subnets

Subnets allow you to organize resources that are related, either logically (for example, one subnet for VMs associated to the same application), or physically (for example, one subnet per resource group). You can also employ subnet isolation techniques for added security.

For cross-premises virtual networks, you should design subnets with the same conventions that you use for on-premises resources. **Azure always uses the first three IP addresses of the address space for each subnet.**

To determine the number of addresses needed for the subnet, start by counting the number of VMs that you need now. Estimate for future growth, and then use the following table to determine the size of the subnet.

| NUMBER OF VMs NEEDED | NUMBER OF HOST BITS NEEDED | SIZE OF THE SUBNET |
|----------------------|----------------------------|--------------------|
| 1–3 | 3 | /29 |
| 4–11 | 4 | /28 |
| 12–27 | 5 | /27 |
| 28–59 | 6 | /26 |
| 60–123 | 7 | /25 |

NOTE

For normal on-premises subnets, the maximum number of host addresses for a subnet with n host bits is $2^n - 2$. For an Azure subnet, the maximum number of host addresses for a subnet with n host bits is $2^n - 5$ (2 plus 3 for the addresses that Azure uses on each subnet).

If you choose a subnet size that is too small, you have to re-IP and redeploy the VMs in the subnet.

Network Security Groups

You can apply filtering rules to the traffic that flows through your virtual networks by using Network Security Groups. You can build granular filtering rules to secure your virtual networking environment, controlling inbound and outbound traffic, source and destination IP ranges, allowed ports, etc. Network Security Groups can be applied to subnets within a virtual network or directly to a given virtual network interface. It is recommended to have some level of Network Security Group filtering traffic on your virtual networks. You can read more about [Network Security Groups](#).

Additional network components

As with an on-premises physical networking infrastructure, Azure virtual networking can contain more than subnets and IP addressing. As you design your application infrastructure, you may want to incorporate some of

these additional components:

- [VPN gateways](#) - connect Azure virtual networks to other Azure virtual networks, or connect to on-premises networks through a Site-to-Site VPN connection. Implement Express Route connections for dedicated, secure connections. You can also provide users direct access with Point-to-Site VPN connections.
- [Load balancer](#) - provides load balancing of traffic for both external and internal traffic as desired.
- [Application Gateway](#) - HTTP load-balancing at the application layer, providing some additional benefits for web applications rather than deploying the Azure load balancer.
- [Traffic Manager](#) - DNS-based traffic distribution to direct end-users to the closest available application endpoint, allowing you to host your application out of Azure datacenters in different regions.

Next steps

The full series of guidelines helps you understand all the different design considerations for building our your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Azure availability sets guidelines

1/17/2017 • 3 min to read • [Edit on GitHub](#)

This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article focuses on understanding the required planning steps for availability sets to ensure your applications remains accessible during planned or unplanned events.

Implementation guidelines for availability sets

Decisions:

- How many availability sets do you need for the various roles and tiers in your application infrastructure?

Tasks:

- Define the number of VMs in each application tier you require.
- Determine if you need to adjust the number of fault or update domains to be used for your application.
- Define the required availability sets using your naming convention and what VMs reside in them. A VM can only reside in one availability set.

Availability sets

In Azure, virtual machines (VMs) can be placed in to a logical grouping called an availability set. When you create VMs within an availability set, the Azure platform distributes the placement of those VMs across the underlying infrastructure. Should there be a planned maintenance event to the Azure platform or an underlying hardware / infrastructure fault, the use of availability sets ensures that at least one VM remains running.

As a best practice, applications should not reside on a single VM. An availability set that contains a single VM doesn't gain any protection from planned or unplanned events within the Azure platform. The [Azure SLA](#) requires two or more VMs within an availability set to allow the distribution of VMs across the underlying infrastructure.

The underlying infrastructure in Azure is divided in to update domains and fault domains. These domains are defined by what hosts share a common update cycle, or share similar physical infrastructure such as power and networking. Azure automatically distributes your VMs within an availability set across domains to maintain availability and fault tolerance. Depending on the size of your application and the number of VMs within an availability set, you can adjust the number of domains you wish to use. You can read more about [managing availability and use of update and fault domains](#).

When designing your application infrastructure, plan the application tiers that you use. Group VMs that serve the same purpose in to availability sets, such as an availability set for your front-end VMs running IIS. Create a separate availability set for your back-end VMs running SQL Server. The goal is to ensure that each component of your application is protected by an availability set and at least once instance always remains running.

Load balancers can be utilized in front of each application tier to work alongside an availability set and ensure traffic can always be routed to a running instance. Without a load balancer, your VMs may continue running throughout planned and unplanned maintenance events, but your end users may not be able to resolve them if the primary VM is unavailable.

Design your application for high availability at storage layer. The best practice is to use separate storage account for each VM in an Availability Set. Keep all disks (OS and data) associated with a VM in the same storage account. Consider storage account [limits](#) when adding more VHDs to a storage account.

Next steps

The full series of guidelines helps you understand all the different design considerations for building our your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)
- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Example Azure infrastructure walkthrough

1/17/2017 • 3 min to read • [Edit on GitHub](#)

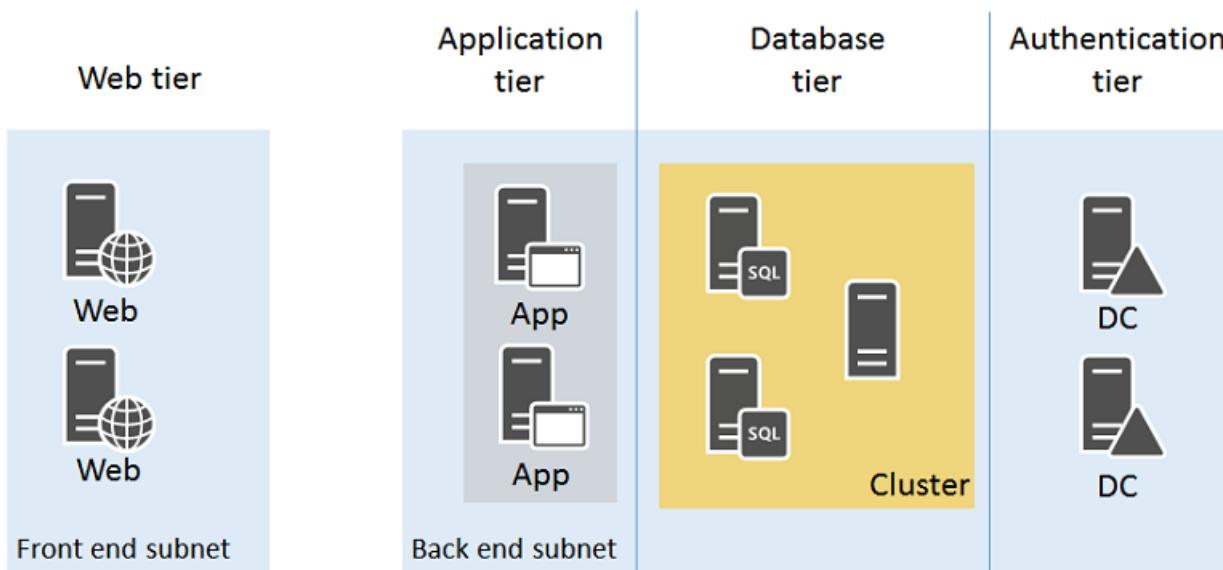
This article is part of a wider series to provide you with design considerations and guidelines as you build out an application infrastructure in Azure. You can [view the additional topics in the series](#). Although you can quickly build a dev/test environment in Azure, there are additional considerations when implementing a production-ready, highly available, and secure environment.

This article walks through building out an example application infrastructure. We detail designing an infrastructure for a simple on-line 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 on-line 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 on-line 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

- Storage accounts
- 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 On-line Store) is the IT workload name and "use" (East US 2) is the location
- Storage accounts use adventureazosusesa**[description]**
 - 'adventure' was added to the prefix to provide uniqueness, and storage account names do not support the use of hyphens.
- 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 accounts

Adventure Works Cycles determined that they needed two storage accounts:

- **adventureazosusesawebapp** for the standard storage of the web servers, application servers, and domain controllers and their data disks.
- **adventureazosusesasql** for the Premium storage of 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 on-line store, Adventure Works Cycles decided on four availability sets:

- **azos-use-as-web** for the web servers

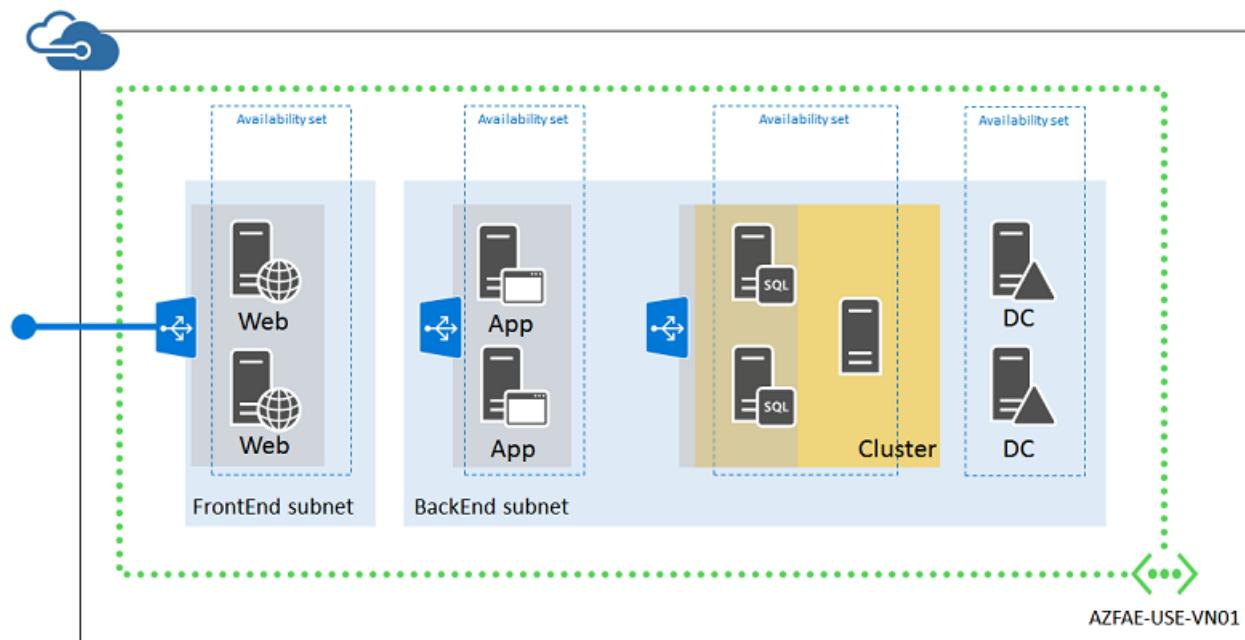
- **azos-use-as-app** for the application servers
- **azos-use-as-sql** for the SQL Servers
- **azos-use-as-dc** for the domain controllers

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)
- Two storage accounts
- Four availability sets, one for each tier of the on-line 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

Next steps

The full series of guidelines helps you understand all the different design considerations for building your application infrastructure in Azure:

- [Azure Subscription and Accounts Guidelines](#)

- [Azure Infrastructure Naming Guidelines](#)
- [Azure Resource Groups Guidelines](#)
- [Azure Storage Guidelines](#)
- [Azure Networking Guidelines](#)
- [Azure Availability Set Guidelines](#)
- [Azure Virtual Machine Guidelines](#)
- [Azure Example Infrastructure Walkthrough](#)

Planned maintenance for virtual machines in Azure

1/17/2017 • 6 min to read • [Edit on GitHub](#)

Understand what Azure planned maintenance is and how it can affect the availability of your Windows virtual machines. This article is also available for [Linux virtual machines](#).

This article provides a background as to the Azure planned maintenance process. If you are wanting to troubleshoot why your VM rebooted, you can [read this blog post detailing viewing VM reboot logs](#).

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.

Why Azure performs planned maintenance

Microsoft Azure periodically performs updates across the globe to improve the reliability, performance, and security of the host infrastructure that underlies virtual machines. Many of these updates are performed without any impact to your virtual machines or Cloud Services, including memory-preserving updates.

However, some updates do require a reboot to your virtual machines to apply the required updates to the infrastructure. The virtual machines are shut down while we patch the infrastructure, and then the virtual machines are restarted.

Please note that there are two types of maintenance that can impact the availability of your virtual machines: planned and unplanned. This page describes how Microsoft Azure performs planned maintenance. For more information about unplanned maintenance, see [Understand planned versus unplanned maintenance](#).

Memory-preserving updates

For a class of updates in Microsoft Azure, customers will not see any impact to their running virtual machines. Many of these updates are to components or services that can be updated without interfering with the running instance. Some of these updates are platform infrastructure updates on the host operating system that can be applied without requiring a full reboot of the virtual machines.

These updates are accomplished with technology that enables in-place live migration, also called a "memory-preserving" update. When updating, the virtual machine is placed into a "paused" state, preserving the memory in RAM, while the underlying host operating system receives the necessary updates and patches. The virtual machine is resumed within 30 seconds of being paused. After resuming, the clock of the virtual machine is automatically synchronized.

Not all updates can be deployed by using this mechanism, but given the short pause period, deploying updates in this way greatly reduces impact to virtual machines.

Multi-instance updates (for virtual machines in an availability set) are applied one update domain at a time.

Virtual machine configurations

There are two kinds of virtual machine configurations: multi-instance and single-instance. In a multi-instance configuration, similar virtual machines are placed in an availability set.

The multi-instance configuration provides redundancy across physical machines, power, and network, and it is

recommended to ensure the availability of your application. All virtual machines in the availability set should serve the same purpose to your application.

For more information about configuring your virtual machines for high availability, refer to [Manage the availability of your Windows virtual machines](#) or [Manage the availability of your Linux virtual machines](#).

By contrast, a single-instance configuration is used for standalone virtual machines that are not placed in an availability set. These virtual machines do not qualify for the service level agreement (SLA), which requires that two or more virtual machines are deployed under the same availability set.

For more information about SLAs, refer to the "Cloud Services, Virtual Machines and Virtual Network" section of [Service Level Agreements](#).

Multi-instance configuration updates

During planned maintenance, the Azure platform first updates the set of virtual machines that are hosted in a multi-instance configuration. This causes a reboot to these virtual machines with approximately 15 minutes of downtime.

In a multi-instance configuration update, virtual machines are updated in a way that preserves availability throughout the process, assuming that each virtual machine serves a similar function as the others in the set.

Each virtual machine in your availability set is assigned an update domain and a fault domain by the underlying Azure platform. Each update domain is a group of virtual machines that will be rebooted in the same time window. Each fault domain is a group of virtual machines that share a common power source and network switch.

For more information about update domains and fault domains, see [Configure multiple virtual machines in an availability set for redundancy](#).

To prevent update domains from going offline at the same time, the maintenance is performed by shutting down each virtual machine in an update domain, applying the update to the host machines, restarting the virtual machines, and moving on to the next update domain. The planned maintenance event ends after all update domains have been updated.

The order of the update domains that are being rebooted may not proceed sequentially during planned maintenance, but only one update domain is rebooted at a time. Today, Azure offers 1-week advanced notification for planned maintenance of virtual machines in the multi-instance configuration.

After a virtual machine is restored, here is an example of what your Windows Event Viewer might display:

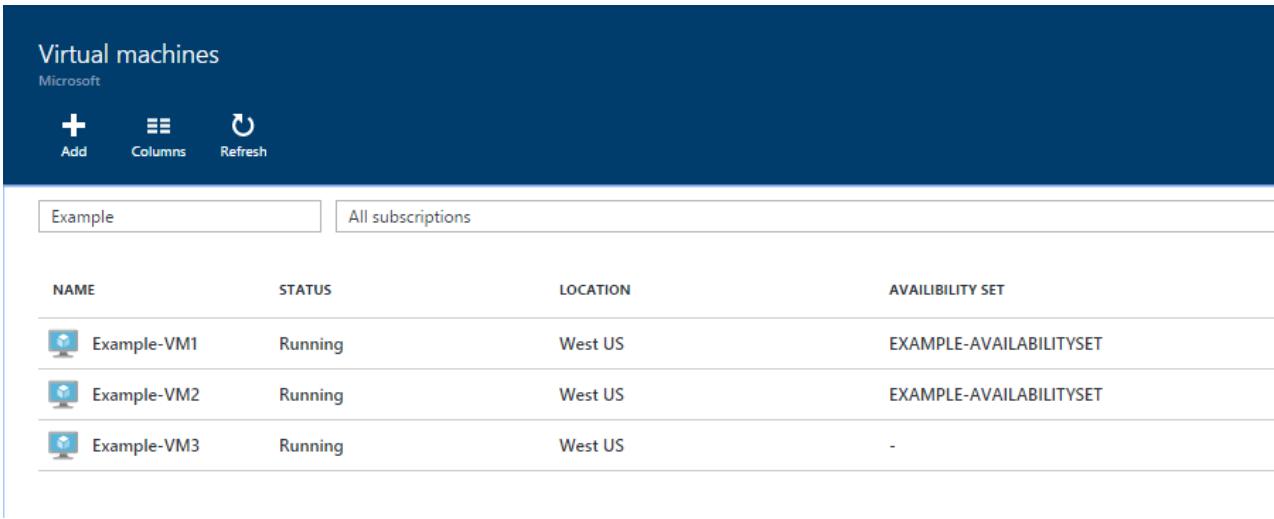
| System Number of events: 4,067 | | | | | |
|--|-----------------------|----------------|----------|---------------|--|
| Filtered: Log: System; Source: ; Event ID: 12, 13. Number of events: 9 | | | | | |
| Level | Date and Time | Source | Event ID | Task Category | |
| Information | 6/1/2014 2:26:34 AM | Kernel-General | 12 | None | |
| Information | 6/1/2014 2:02:05 AM | Kernel-General | 13 | None | |
| Information | 4/26/2014 12:16:34 AM | Hyper-V-Netvsc | 12 | (1003) | |
| Information | 4/26/2014 12:16:32 AM | Kernel-General | 12 | None | |
| Information | 4/26/2014 12:16:25 AM | Kernel-General | 13 | None | |
| Information | 4/26/2014 12:15:47 AM | Hyper-V-Netvsc | 12 | (1003) | |
| Information | 4/26/2014 12:15:05 AM | Kernel-General | 12 | None | |
| Information | 4/10/2014 6:54:17 PM | Kernel-General | 13 | None | |

Event 12, Kernel-General

General Details

The operating system started at system time 2014-06-01T02:26:34.485247200Z.

Use the viewer to determine which virtual machines are configured in a multi-instance configuration using the Azure portal, Azure PowerShell, or Azure CLI. For example, to determine which virtual machines are in a multi-instance configuration, you can browse the list of virtual machines with the Availability Set column added to the virtual machines browse dialog. In the following example, the Example-VM1 and Example-VM2 virtual machines are in a multi-instance configuration:



The screenshot shows the Azure portal's 'Virtual machines' blade. At the top, there are buttons for 'Add', 'Columns', and 'Refresh'. Below that, a search bar contains 'Example' and a link to 'All subscriptions'. The main area displays a table with three rows of virtual machine data:

| NAME | STATUS | LOCATION | AVAILABILITY SET |
|-------------|---------|----------|-------------------------|
| Example-VM1 | Running | West US | EXAMPLE-AVAILABILITYSET |
| Example-VM2 | Running | West US | EXAMPLE-AVAILABILITYSET |
| Example-VM3 | Running | West US | - |

Single-instance configuration updates

After the multi-instance configuration updates are complete, Azure will perform single-instance configuration updates. This update also causes a reboot to your virtual machines that are not running in availability sets.

Please note that even if you have only one instance running in an availability set, the Azure platform treats it as a multi-instance configuration update.

For virtual machines in a single-instance configuration, virtual machines are updated by shutting down the virtual machines, applying the update to the host machine, and restarting the virtual machines, approximately 15 minutes of downtime. These updates are run across all virtual machines in a region in a single maintenance window.

This planned maintenance event will impact the availability of your application for this type of virtual machine configuration. Azure offers a 1-week advanced notification for planned maintenance of virtual machines in the single-instance configuration.

Email notification

For single-instance and multi-instance virtual machine configurations only, Azure sends email communication in advance to alert you of the upcoming planned maintenance (1-week in advance). This email will be sent to the subscription administrator and co-administrator email accounts. Here is an example of this type of email:

Azure

Upcoming maintenance will affect single instance deployments of Microsoft Azure Virtual Machines.



As part of our ongoing commitment to performance, reliability, and security, we sometimes perform maintenance operations in our Azure regions and datacenters. These maintenance operations impact single instance deployments of Virtual Machines that are not using availability sets. To learn more, see the [availability sets](#) webpage.

We want to notify you of an upcoming maintenance operation. We are scheduling the update to occur during nonbusiness hours as much as possible in each region. It is also being scheduled to avoid affecting two regions in close proximity at the same time. Single instance virtual machine deployments that are not in [availability sets](#) will reboot once during this maintenance operation.

The following are the planned start times, provided in both Universal Time Coordinated (UTC) and United States Pacific Daylight Time (PDT). **We expect the update to finish within six to eight hours of the start time.**

| Region | PDT | UTC |
|----------------|---------------------------|---------------------------|
| Southeast Asia | 9:00
Friday, April 11 | 16:00
Friday, April 11 |
| North Europe | 11:00
Friday, April 11 | 18:00
Friday, April 11 |
| East US | 15:00
Friday, April 11 | 22:00
Friday, April 11 |
| Japan East | 14:00
Friday, April 11 | 21:00
Friday, April 11 |

Region pairs

When executing maintenance, Azure will only update the Virtual Machine instances in a single region of its 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. This will be scheduled at a separate time, enabling failover or load balancing between regions. However, other regions such as North Europe can be under maintenance at the same time as East US.

Please refer to the following table for information regarding current region pairs:

| REGION 1 | REGION 2 |
|------------------|------------------|
| North Central US | South Central US |

| REGION 1 | REGION 2 |
|---------------------|------------------|
| East US | West US |
| US East 2 | Central US |
| North Europe | West Europe |
| South East Asia | East Asia |
| East China | North China |
| Japan East | Japan West |
| Brazil South | South Central US |
| Australia Southeast | Australia East |
| India Central | India South |
| India West | India South |
| US Gov Iowa | US Gov Virginia |

How to Schedule Planned Maintenance on Azure VMs

1/17/2017 • 3 min to read • [Edit on GitHub](#)

IMPORTANT

Azure has two different deployment models for creating and working with resources: [Resource Manager and Classic](#). This article covers using the Classic deployment model. Microsoft recommends that most new deployments use the Resource Manager model. For information about planned maintenance in the Resource Manager model, see [here](#).

Multi and Single Instance VMs

For many customers running on Azure, it is critical that they are able to schedule when their VMs undergo planned maintenance, since this results in ~15 minutes of downtime. You can leverage availability sets to help control when provisioned VMs receive planned maintenance.

There are two possible configurations for VMs running on Azure. VMs are either configured as multi-instance or single-instance. If VMs are in an availability set, then they are configured as multi-instance. Note, even single VMs can be deployed in an availability set and they will be treated as multi-instance. If VMs are NOT in an availability set, then they are configured as single-instance. For details on availability sets, please see either [Manage the Availability of your Windows Virtual Machines](#) or [Manage the Availability of your Linux Virtual Machines](#).

Planned maintenance updates to single-instance and multi-instance VMs happen separately. By reconfiguring your VMs to be single-instance (if they are multi-instance) or to be multi-instance (if they are single-instance), you can control when their VMs receive the planned maintenance. Please see either [Planned maintenance for Azure Linux virtual machines](#) or [Planned maintenance for Azure Windows virtual machines](#) for details on planned maintenance for Azure VMs.

For Multi-instance Configuration

You can select the time planned maintenance impacts your VMs that are deployed in an Availability Set configuration by removing these VMs from availability sets.

1. An email will be sent to you 7 calendar days before the planned maintenance to your VMs in a Multi-instance configuration. The subscription IDs and names of the affected Multi-instance VMs will be included in the body of the email.
2. During those 7 days, you can choose the time your instances are updated by removing your multi-instance VMs in that region from their availability set. This change in configuration will result in a reboot, as the Virtual Machine is moving from one physical host, targeted for maintenance, to another physical host that isn't targeted for maintenance.
3. You can remove the VM from its availability set in the classic portal.
 - a. In the Classic portal, click on the VM and then select "configure."
 - b. Under "settings", you can see which Availability Set the VM is in.

settings

VIRTUAL MACHINE TIER

BASIC STANDARD

VIRTUAL MACHINE SIZE

D1 (1 core, 3.5 GB memory)

AVAILABILITY SET

AS1

Virtual machines in the selected availability set:

CATHERINEVM

- c. In the availability set dropdown menu, select "remove from availability set."

settings

VIRTUAL MACHINE TIER

BASIC STANDARD

VIRTUAL MACHINE SIZE

D1 (1 core, 3.5 GB memory)

AVAILABILITY SET

Remove from availability set

- d. At the bottom, select "save." Select "yes" to acknowledge that this action will restart the VM.
4. These VMs will be moved to Single-Instance hosts and will not be updated during the planned maintenance to Availability Set Configurations.
5. Once the update to Availability Set VMs is complete (according to schedule outlined in the original email), you should add the VMs back into their availability sets, and they will be re-configured as multi-instance VMs. Moving the VMs from Single-instance back to Multi-instance will result in a reboot. Typically, once all multi-instance updates are completed across the entire Azure environment, single-instance maintenance follows.

This can also be achieved using Azure PowerShell:

```
Get-AzureVM -ServiceName "<VmCloudServiceName>" -Name "<VmName>" | Remove-AzureAvailabilitySet | Update-AzureVM
```

For Single-instance Configuration

You can select the time planned maintenance impacts your VMs in a Single-instance configuration by adding these VMs into availability sets.

Step-by-step

1. An email will be sent to you 7 calendar days before the planned maintenance to VMs in a Single-instance configuration. The subscription IDs and names of the affected Single-Instance VMs will be included in the body of the email.
2. During those 7 days, you can choose the time your instance reboots by moving your Single-instance VMs into an availability set in that same region. This change in configuration will result in a reboot, as the Virtual Machine is moving from one physical host, targeted for maintenance, to another physical host that isn't targeted for maintenance.
3. Follow instructions here to add existing VMs into availability sets using the Classic Portal and Azure PowerShell (see Azure PowerShell sample in the note below).
4. Once these VMs are re-configured as Multi-instance, they will be excluded from the planned maintenance to Single-instance VMs.

5. Once the update to single-instance VMs is complete (according to schedule outlined in the original email), you can remove the VMs from their availability sets, and they will be re-configured as single-instance VMs.

This can also be achieved using Azure PowerShell:

```
Get-AzureVM -ServiceName "<VmCloudServiceName>" -Name "<VmName>" | Set-AzureAvailabilitySet -AvailabilitySetName "<AvSetName>" | Update-AzureVM
```

Options with HPC Pack to create and manage a Windows HPC cluster in Azure

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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 is Microsoft's free HPC solution built on Microsoft Azure and Windows Server technologies and supports a wide range of HPC workloads.

For more options to run large-scale parallel, batch, and HPC workloads 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](#).

Run an HPC Pack cluster in Azure VMs

Azure templates

- ([GitHub](#)) [HPC Pack 2016 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 on Windows Server 2012 R2](#)
- [HPC Pack compute node on Windows Server 2012 R2](#)
- [HPC Pack compute node with Excel on Windows Server 2012 R2](#)

PowerShell deployment script

- [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 cluster in Azure with Azure Active Directory](#)

Add worker role nodes to an HPC Pack cluster

- [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](#)

Integrate with Azure Batch

- [Burst to Azure Batch with HPC Pack](#)

Create RDMA clusters for MPI workloads

- [Set up a Windows RDMA cluster with HPC Pack to run MPI applications](#)

Create MATLAB Distributed Computing Server clusters on Azure VMs

1/17/2017 • 3 min to read • [Edit on GitHub](#)

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.
- **Cores quota** - You might need to increase the core 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

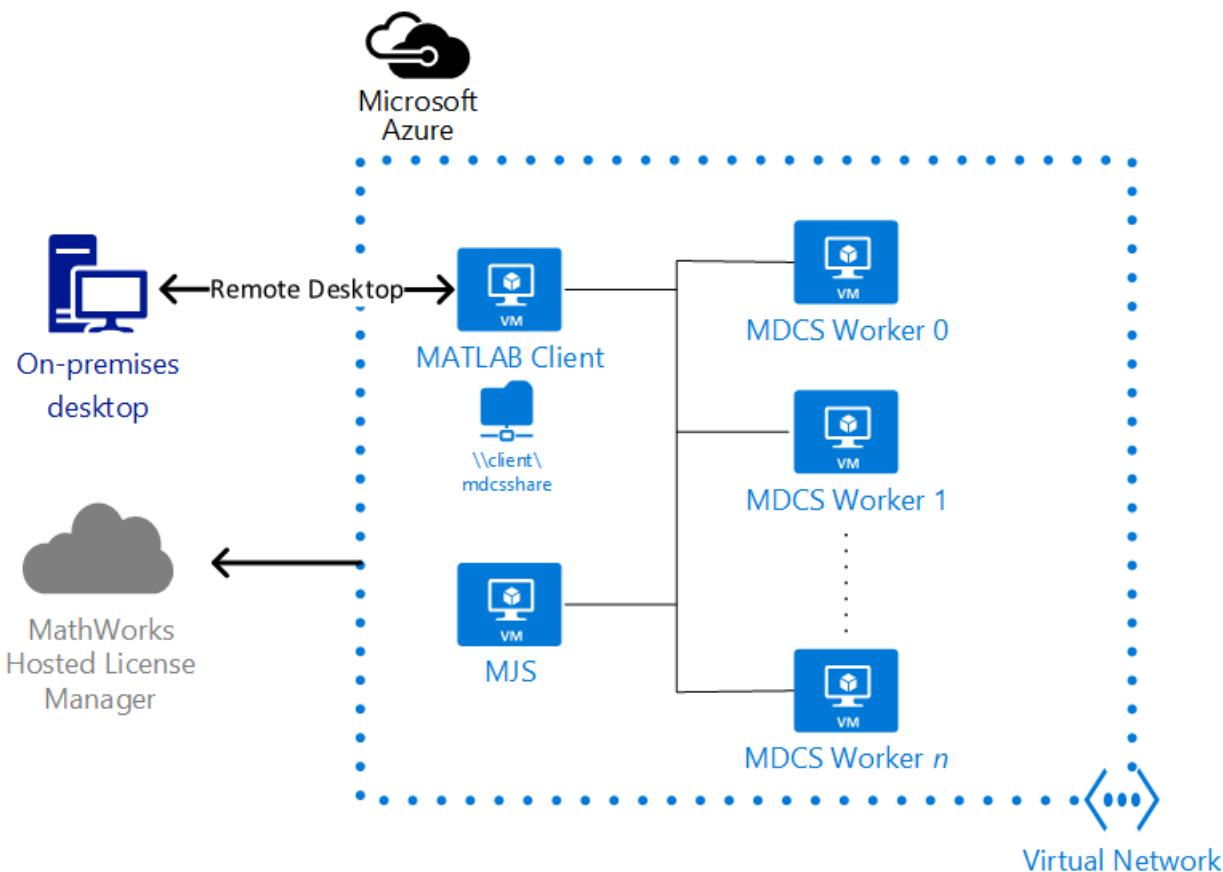
- Use the supplied PowerShell script or use the quickstart template to create a cluster from the base VM image.
- Manage the clusters using the supplied PowerShell script which allows you to list, pause, resume, and delete clusters.

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.

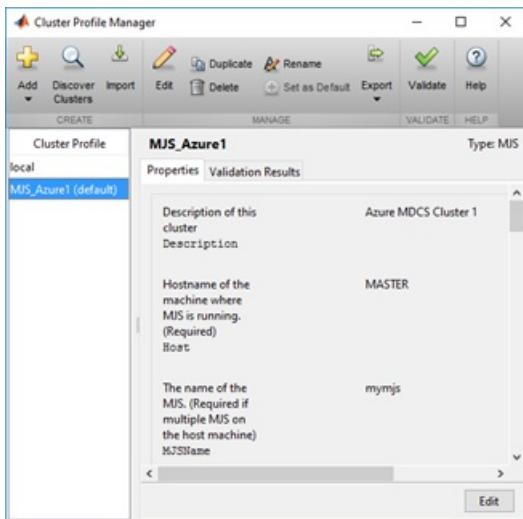


- To use the cluster, connect by Remote Desktop to the client node. The client node runs the MATLAB client.
- The client node has a file share that can be accessed by all workers.
- MathWorks Hosted License Manager is used for the license checks for all MATLAB software.
- By default, one MATLAB Distributed Computing Server worker per core 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.

Install and configure MongoDB on a Windows VM in Azure

1/17/2017 • 5 min to read • [Edit on GitHub](#)

MongoDB is a popular open-source, high-performance NoSQL database. This article guides you through installing and configuring MongoDB on a Windows Server 2012 R2 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 [Create a Windows Server VM using Azure PowerShell](#)
- [Attach a data disk to a Windows Server VM using the Azure portal](#) or [Attach a data disk to a Windows Server VM using Azure PowerShell](#)

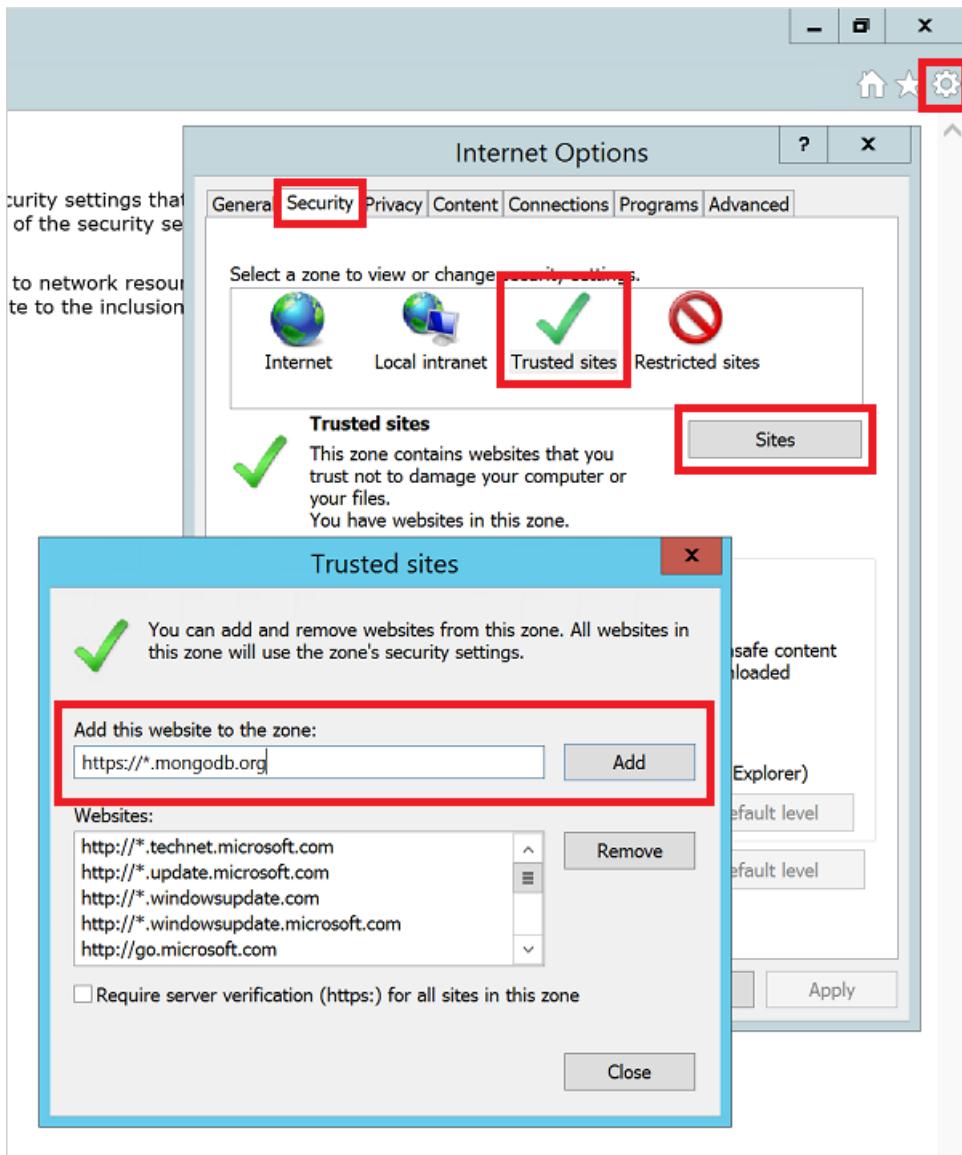
To begin installing and configuring MongoDB, [log on to your Windows Server VM](#) by using Remote Desktop.

Install MongoDB

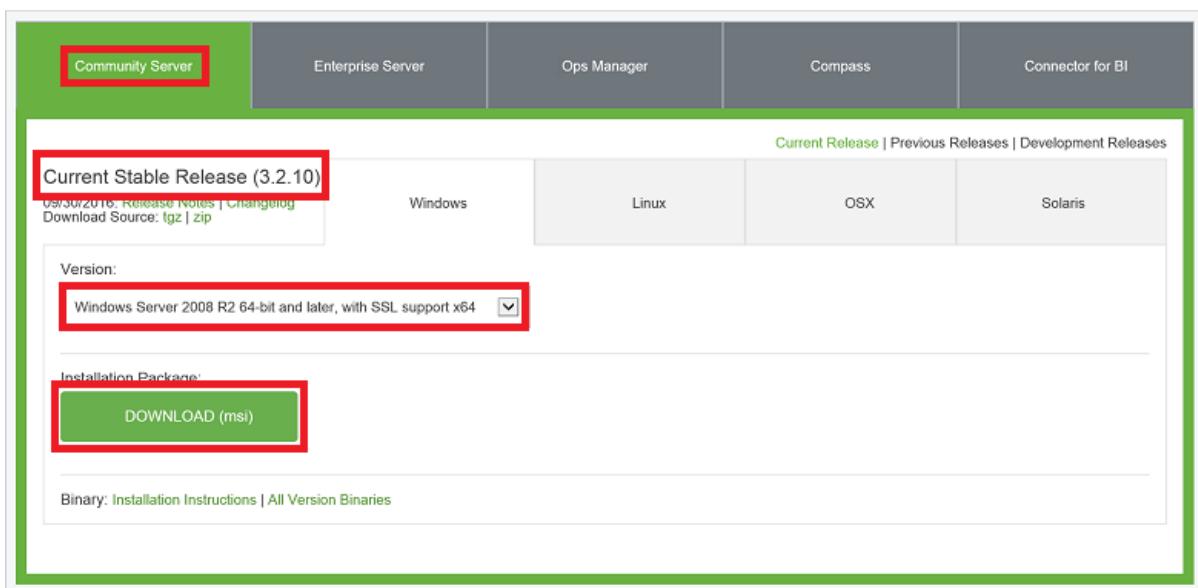
IMPORTANT

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 **Start** menu on the VM.
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.org** to the list of trusted sites, and then close the dialog box.



4. Browse to the [MongoDB - Downloads](http://www.mongodb.org/downloads) page (<http://www.mongodb.org/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.

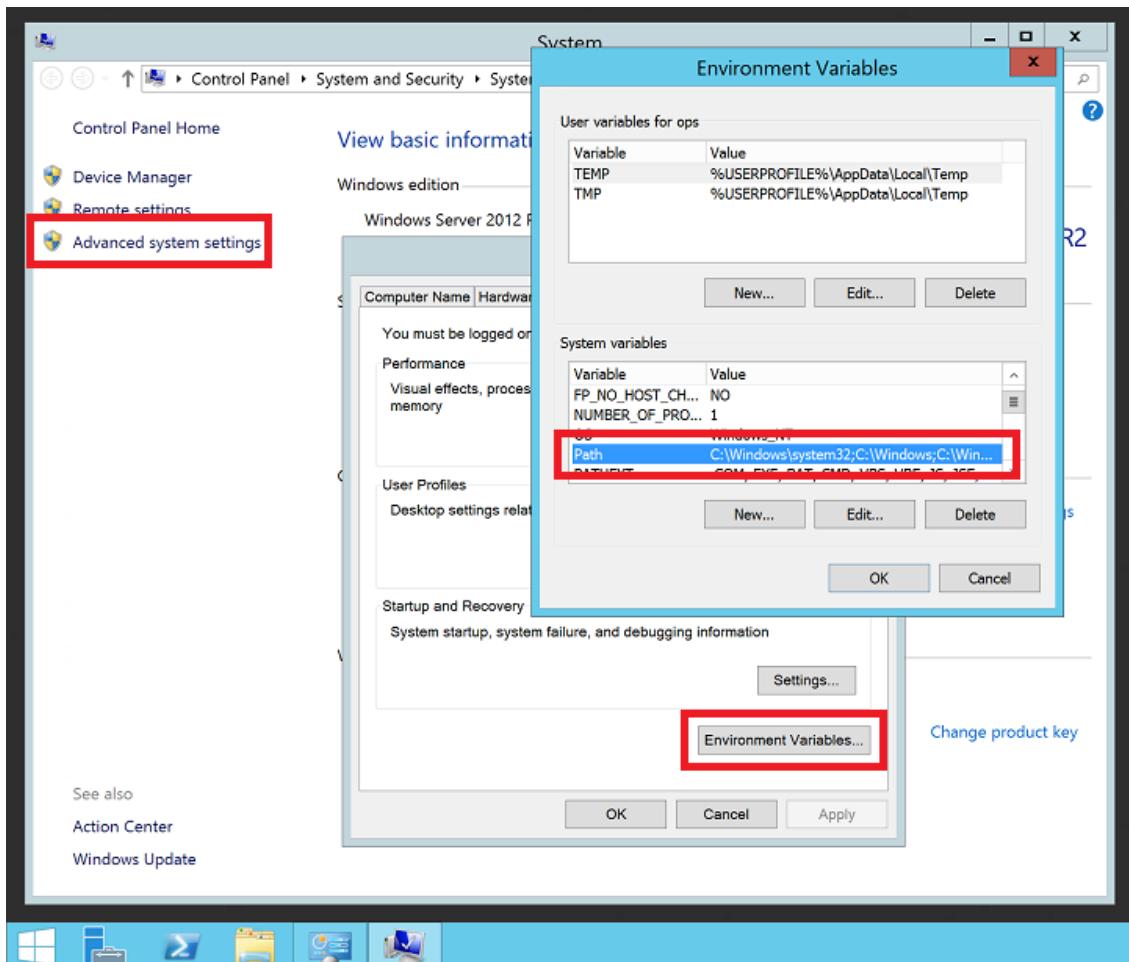
6. Read and accept the license agreement. When you're prompted, select **Complete** install.

7. 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\3.2\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
```

- 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.

- 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](#).

Overview of SQL Server on Azure Virtual Machines

1/17/2017 • 6 min to read • [Edit on GitHub](#)

This topic describes your options for running SQL Server on Azure virtual machines (VMs), along with [links to portal images](#) and an overview of [common tasks](#).

NOTE

If you're already familiar with SQL Server and just want to see how to deploy a SQL Server VM, see [Provision a SQL Server virtual machine in the Azure portal](#).

Overview

If you are a database administrator or a developer, Azure VMs provide a way to move your on-premises SQL Server workloads and applications to the Cloud. The following video provides a technical overview of SQL Server Azure VMs.



The video covers the following areas:

| TIME | AREA |
|-------|-------------------------------------|
| 00:21 | What are Azure VMs? |
| 01:45 | Security |
| 02:50 | Connectivity |
| 03:30 | Storage reliability and performance |
| 05:20 | VM sizes |
| 05:54 | High availability and SLA |
| 07:30 | Configuration support |
| 08:00 | Monitoring |
| 08:32 | Demo: Create a SQL Server 2016 VM |

NOTE

The video focuses on SQL Server 2016, but Azure provides VM images for many versions of SQL Server, including 2008, 2012, 2014, and 2016.

Scenarios

There are many reasons that you might choose to host your data in Azure. If your application is moving to Azure, it improves performance to also move the data. But there are other benefits. You automatically have access to multiple data centers for a global presence and disaster recovery. The data is also highly secured and durable.

SQL Server running on Azure VMs is one option for storing your relational data in Azure. It is good choice for several scenarios. For example, you might want to configure the Azure VM as similarly as possible to an on-premises SQL Server machine. Or you might want to run additional applications and services on the same database server. There are two main resources that can help you think through even more scenarios and considerations:

- [SQL Server on Azure virtual machines](#) provides an overview of the best scenarios for using SQL Server on Azure VMs.
- [Choose a cloud SQL Server option: Azure SQL \(PaaS\) Database or SQL Server on Azure VMs \(IaaS\)](#) provides a detailed comparison between SQL Database and SQL Server running on a VM.

Create a new SQL VM

The following sections provide direct links to the Azure portal for the SQL Server virtual machine gallery images. Depending on the image you select, you can either pay for SQL Server licensing costs on a per-minute basis, or you can bring your own license (BYOL).

Find step-by-step guidance for this process in the tutorial, [Provision a SQL Server virtual machine in the Azure portal](#). Also, review the [Performance best practices for SQL Server VMs](#), which explains how to select the appropriate machine size and other features available during provisioning.

Option 1: Create a SQL VM with per-minute licensing

The following table provides a matrix of the latest SQL Server images in the virtual machine gallery. Click on any link to begin creating a new SQL VM with your specified version, edition, and operating system.

| VERSION | OPERATING SYSTEM | EDITION |
|--|------------------------|---|
| 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 |

In addition to this list, other combinations of SQL Server versions and operating systems are available. Find other images through a marketplace search in the Azure portal.

Option 2: Create a SQL VM with an existing 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. To use your own license, use the matrix of SQL Server versions, editions, and operating systems below. In the portal, these image names are prefixed with **{BYOL}**.

| VERSION | OPERATING SYSTEM | EDITION |
|----------------------------|------------------------|--------------------------------|
| 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 addition to this list, other combinations of SQL Server versions and operating systems are available. Find other images through a marketplace search in the Azure portal (search for "{BYOL} SQL Server").

IMPORTANT

To use BYOL VM images, you must have an Enterprise Agreement with [License Mobility through Software Assurance on Azure](#). You also need a valid license for the version/edition of SQL Server you want to use. You must [provide the necessary BYOL information to Microsoft](#) within **10** days of provisioning your VM.

NOTE

It is not possible to change the licensing model of a pay-per-minute SQL Server VM to use your own license. In this case, you must create a new BYOL VM and migrate your databases to the new VM.

Manage your SQL VM

After provisioning your SQL Server VM, there are several optional management tasks. In many aspects, you configure and manage SQL Server exactly like you would manage an on-premises SQL Server instance. However, some tasks are specific to Azure. The following sections highlight some of these areas with links to more information.

Connect to the VM

One of the most basic management steps is to connect to your SQL Server VM through tools, such as SQL Server Management Studio (SSMS). For instructions on how to connect to your new SQL Server VM, 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](#).

Configure high availability

If you require high availability, consider configuring SQL Server Availability Groups. This involves multiple Azure VMs in a virtual network. The Azure portal has a template that sets up this configuration for you. For more information, see [Configure an AlwaysOn availability group in Azure Resource Manager virtual machines](#). If you want to manually configure your Availability Group and associated listener, see [Configure AlwaysOn Availability Groups in Azure VM](#).

For other high availability considerations, see [High Availability and Disaster Recovery for SQL Server in Azure Virtual Machines](#).

Back up your data

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](#). For an overview of all backup and restore options, see [Backup and Restore for SQL Server in Azure Virtual Machines](#).

Automate updates

Azure VMs can use [Automated Patching](#) to schedule a maintenance window for installing important windows and SQL Server updates automatically.

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, see the CEIP section of the [Accept License Terms](#) topic.

Next steps

[Explore the Learning Path](#) for SQL Server on Azure virtual machines.

For questions about pricing, see [Pricing](#). Select your target edition of SQL Server in the **OS/Software** list. Then view the prices for differently sized virtual machines.

More question? First, see the [SQL Server on Azure Virtual Machines FAQ](#). But also add your questions or comments to the bottom of any SQL VM topics to interact with Microsoft and the community.

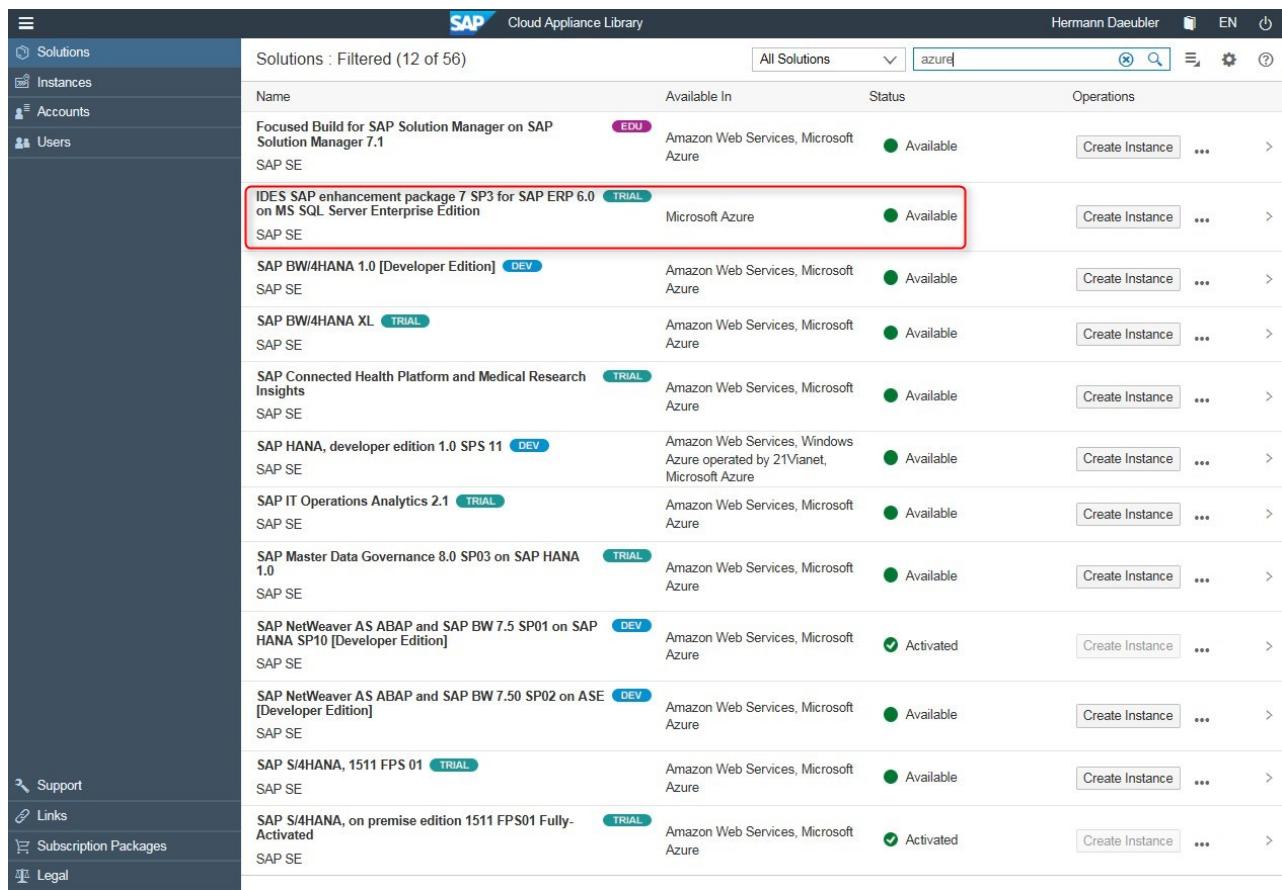
Deploying SAP IDES EHP7 SP3 for SAP ERP 6.0 on Microsoft Azure

1/17/2017 • 2 min to read • [Edit on GitHub](#)

This article describes how to deploy SAP IDES running with SQL Server and Windows OS on Microsoft Azure via SAP Cloud Appliance Library 3.0. The screenshots show the process step by step. Deploying other solutions in the list works the same way from a process perspective. One just has to select a different solution.

To start with SAP Cloud Appliance Library (SAP CAL) go [here](#). There is a blog from SAP about the new [SAP Cloud Appliance Library 3.0](#).

The following screenshots show step-by-step how to deploy SAP IDES on Microsoft Azure. The process works the same way for other solutions.



| Name | Available In | Status | Operations |
|--|--|-----------|---------------------------------------|
| Focused Build for SAP Solution Manager on SAP Solution Manager 7.1 SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| IDES SAP enhancement package 7 SP3 for SAP ERP 6.0 on MS SQL Server Enterprise Edition SAP SE | Microsoft Azure | Available | Create Instance ... > |
| SAP BW/4HANA 1.0 [Developer Edition] SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP BW/4HANA XL SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP Connected Health Platform and Medical Research Insights SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP HANA, developer edition 1.0 SPS 11 SAP SE | Amazon Web Services, Windows Azure operated by 21Vianet, Microsoft Azure | Available | Create Instance ... > |
| SAP IT Operations Analytics 2.1 SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP Master Data Governance 8.0 SP03 on SAP HANA 1.0 SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP NetWeaver AS ABAP and SAP BW 7.5 SP01 on SAP HANA SP10 [Developer Edition] SAP SE | Amazon Web Services, Microsoft Azure | Activated | Create Instance ... > |
| SAP NetWeaver AS ABAP and SAP BW 7.50 SP02 on ASE [Developer Edition] SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP S/4HANA, 1511 FPS 01 SAP SE | Amazon Web Services, Microsoft Azure | Available | Create Instance ... > |
| SAP S/4HANA, on premise edition 1511 FPS01 Fully Activated SAP SE | Amazon Web Services, Microsoft Azure | Activated | Create Instance ... > |

The first picture shows all solutions that are available on Microsoft Azure. The highlighted Windows-based SAP IDES solution that is only available on Azure was chosen to go through the process.

The screenshot shows the SAP Cloud Appliance Library interface. On the left, there's a sidebar with links for Solutions, Instances, Accounts (which is selected), and Users. The main area has a title 'Create Account' with a red box around it. Below it, 'Account Details' is selected. The '1. Account Details' section asks for general properties. The 'Name' field contains 'IDESAZCAL'. The 'Cloud Provider' dropdown is set to 'Microsoft Azure', with 'Amazon Web Services' as an alternative. The 'Subscription ID' dropdown shows 'Microsoft Azure' and 'Windows Azure operated by 21Vianet', with the latter being highlighted by a red box. A 'Download New Certificate' button is present. Below the form, steps 2 and 3 are listed: 'Upload the certificate to your subscription in Microsoft Azure within "1" hours.' and 'Test your connection.' A 'Test Connection' button is at the bottom. The bottom right of the main area has a 'Cancel' button.

First a new SAP CAL account has to be created. There are currently two choices for Azure - standard Azure and Azure on China mainland that is operated by partner 21Vianet.

This screenshot is similar to the previous one but includes a download dialog at the bottom. The dialog is yellow and asks, 'Do you want to open or save certificate.cer from calsap.com?'. It has 'Open', 'Save', and 'Cancel' buttons. The rest of the interface is identical to the first screenshot, showing the account creation form for Microsoft Azure.

Then one has to enter the Azure subscription ID that can be found on the Azure portal - also see further down how to get it. Afterwards an Azure management certificate needs to be downloaded.

The screenshot shows the Microsoft Azure portal interface. On the left sidebar, under the 'Subscriptions' section, the 'Subscriptions' item is highlighted with a red box. The main content area displays a table of active subscriptions:

| SUBSCRIPTION | SUBSCRIPTION ID | ROLE | SUBSCRIPTION STATUS |
|--------------|------------------------------------|----------------|---------------------|
| Azu ... | 195d8e-101c-0000-0000-000000000000 | Owner | Active |
| HAN ... | 21affa-101c-0000-0000-000000000000 | Owner | Active |
| SAP ... | 666a07-0000-0000-0000-000000000000 | Owner | Active |
| SAP ... | 666a07-0000-0000-0000-000000000000 | Owner | Active |
| SAP ... | 666a07-0000-0000-0000-000000000000 | Account editor | Active |

In the new Azure portal one finds the item "Subscriptions" on the left side. Click it to show all active subscriptions for your user.

The screenshot shows the Microsoft Azure portal interface. The 'SAP ...' subscription is selected. In the top navigation bar, the 'Settings' link is highlighted with a red box. The main content area displays the 'Management certificates' section, which includes a callout box explaining the use of service principals:

Client tools such as Visual Studio, Azure SDKs, and Windows PowerShell have traditionally used management certificates to authenticate access to your subscriptions. You can now use managed service principal accounts in Azure Active Directory, which can have finer-grained access for increased security to limit the risk of exposure. Management certificates are only applicable to the classic service management APIs, whereas service principals are required for the new Resource Manager APIs.

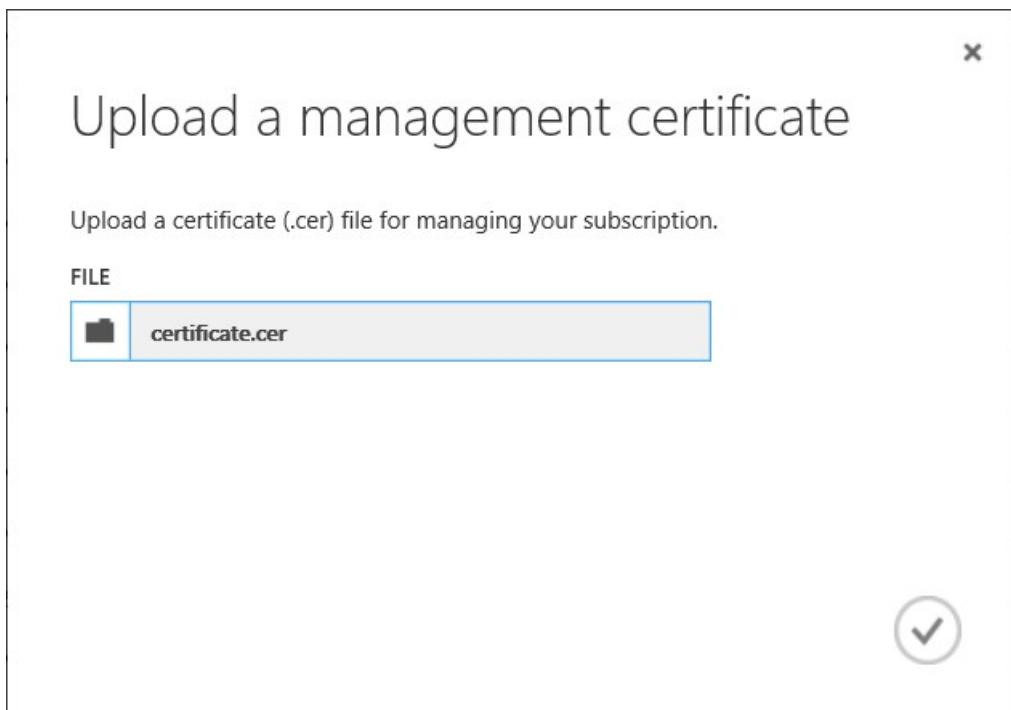
Learn more about using service principals with Resource Manager

Selecting one of the subscriptions and then choosing "Management Certificates" explains that there is a new concept using "service principals" for the new Azure Resource Manager model. SAP CAL isn't adapted yet for this new model and still requires the "classic" model and the former Azure portal to work with management certificates.

The screenshot shows the Microsoft Azure portal interface. On the left, a sidebar lists various services: SCHEDULER, API MANAGEMENT, MACHINE LEARNING, STREAM ANALYTICS, OPERATIONAL INSIGHTS, NETWORKS, TRAFFIC MANAGER, MANAGEMENT SERVICES, ACTIVE DIRECTORY, MARKETPLACE, and STORSIMPLE. At the bottom of the sidebar is a 'SETTINGS' icon, which is highlighted with a red box. The main content area has a title 'settings'. Below it is a navigation bar with tabs: SUBSCRIPTIONS (highlighted with a red box), MANAGEMENT CERTIFICATES, ADMINISTRATORS, AFFINITY GROUPS, and USAGE. Underneath is another navigation bar with tabs: SUBSCRIPTION, SUBSCRIPTION ID (highlighted with a red box), ACCOUNT ADMINISTRATOR, DIRECTORY, and a magnifying glass icon. The main table displays subscription details: SAP [...], e8b0c2ca-722b-4ba1-9e36-5aef7a60319 [highlighted with a red box], hermannnd@microsoft.com, and Microsoft (microsoft...).

Here one can see the former Azure portal. The upload of a management certificate gives SAP CAL the permissions to create virtual machines within a customer subscription. Under the "SUBSCRIPTIONS" tab one can find the subscription ID that has to be entered in the SAP CAL portal.

On the second tab, it's then possible to upload the management certificate that was downloaded before from SAP CAL.



A little dialog pops up to select the downloaded certificate file.

SAP Cloud Appliance Library

Hermann Daeubler EN ⌂

← Create Account ⓘ

1 Account Details

1. Account Details

Enter the general properties of the account:

*Name: IDESAZCAL

Description:

*Cloud Provider: Microsoft Azure

*Subscription ID: e[REDACTED]!9

To activate the account, complete the following steps:

1. Download and save the management certificate locally as an Internet security file (.CER file extension). [Download New Certificate](#)
2. Upload the certificate to your subscription in Microsoft Azure within "1" hours. [More Information](#)
3. Test your connection. [Test Connection](#)

The connection with your subscription in Microsoft Azure is valid.

Step 2

Cancel

Once the certificate was uploaded the connection between SAP CAL and the customer Azure subscription can be tested within SAP CAL. A little message should pop up which tells that the connection is valid.

SAP Cloud Appliance Library

Hermann Daeubler EN ⌂ ⓘ

Basic Mode: Create Instance

Account Details

Choose an existing account
 Create a new account

*Account: IDESAZCAL

Instance Details

*Name: IDESSQL (highlighted with red box)

*Region: West Europe

*Password: *****

*Retype Pa...: *****

The valid characters are: A-Z, a-z, 0-9, \$, #, _
The first character has to be one of the following: A-Z, a-z, \$, #
The first 3 characters cannot be one and the same
The password must contain at least 1 lowercase letter(s)
The password must contain at least 1 uppercase letter (s)
The password must contain at least 1 digit(s)
The password must be between 8 and 9 characters long

Solution
IDES SAP enhancement package 7 SP3 for SAP ERP 6.0 on MS SQL Server Enterprise Edition

Account
IDESAZCAL
Microsoft Azure

Cost Forecast
Disclaimer
USD 2.20 per hour when Active
USD 6.96 per month when Suspended

Note: Note that you will be charged by the cloud provider for this solution instance and its associated storage.

To reduce additional costs, your instance will be suspended after 8 hours.

Advanced Mode **Create** Cancel

After the setup of an account one has to select a solution that should be deployed and create an instance. With "basic" mode, it's really trivial. Enter an instance name, choose an Azure region and define the master password for the solution.

SAP Cloud Appliance Library

Hermann Daeubler EN ⌂ ⓘ

Instances : All (3)

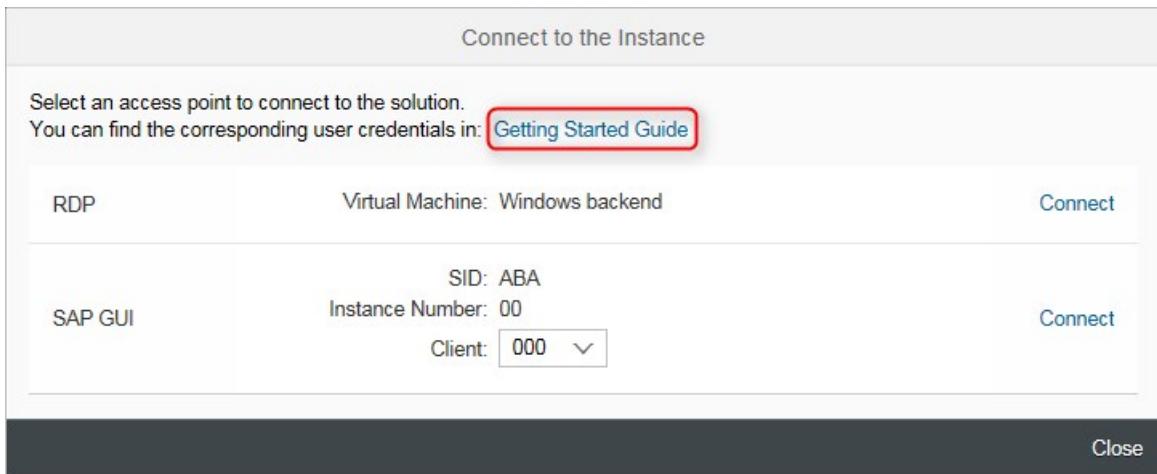
| Name | Owned By | Created On | Scheduled | Status | Operations |
|---------|-----------------------------|------------------------|-----------|---|------------|
| IDESSQL | Hermann Daeubler (C5042276) | Sep 12, 2016, 18:33:03 | No | Active | ... |
| IDEASQL | Hermann Daeubler (C5042276) | Sep 12, 2016, 18:33:03 | No | Not Active | ... |

After some time depending on the size and complexity of the solution (an estimation is given by SAP CAL) it's shown as "active" and ready for use. It is very simple.

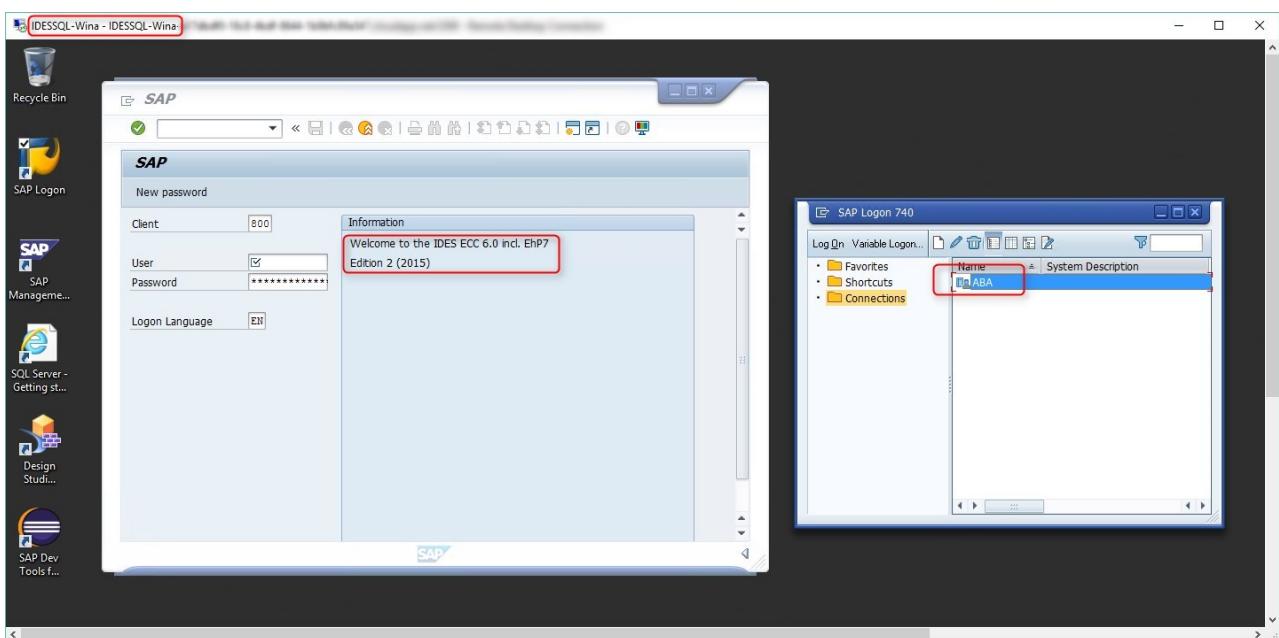
The screenshot shows the SAP Cloud Appliance Library interface. The left sidebar has navigation links: Solutions, Instances, Accounts, Users, Support, Links, Subscription Packages, and Legal. The main header is "SAP Cloud Appliance Library". The top right shows user info: Hermann Daeubler, EN, and a power button icon. The page title is "IDESSQL". The top navigation bar includes "Connect", "Suspend", "Edit", a help icon, and three dots. Below the title, tabs are "INFO", "SOLUTION INFO", "VIRTUAL MACHINES" (which is selected), and "SCHEDULE". A section titled "VIRTUAL MACHINES" contains a table with one row: "Windows backend" (Size: D12). Another section titled "Access Points" contains a table with two rows: "Windows backend" (Service: RDP, Port Range: 3389, IP Range: 0.0.0.0/0, Enabled: checked) and "Windows backend" (Service: SAP GUI, Port Range: 3200, IP Range: 0.0.0.0/0, Enabled: checked). Both access points are of type "Default".

Looking at some details of the solution one can see which kind of VMs were deployed. In this case there is one single Azure VM of size D12 that was created by SAP CAL.

On the Azure portal, the virtual machine can be found starting with the same instance name that was given in SAP CAL.



Now it's possible to connect to the solution via the connect button in the SAP CAL portal. The little dialog contains a link to a user guide that describes all the default credentials to work with the solution. [Here](#) is the link to the guide for the IDES solution.



Another option is to login to the Windows VM and start for example the pre-configured SAP GUI.

SAP NetWeaver on Azure Virtual Machines (VMs) – Planning and Implementation Guide

1/17/2017 • 115 min to read • [Edit on GitHub](#)

Microsoft Azure enables companies to acquire compute and storage resources in minimal time without lengthy procurement cycles. Azure Virtual Machines allow companies to deploy classical applications, like SAP NetWeaver based applications into Azure and extend their reliability and availability without having further resources available on-premises. Azure Virtual Machine Services also supports cross-premises connectivity, which enables companies to actively integrate Azure Virtual Machines into their on-premises domains, their Private Clouds and their SAP System Landscape. This white paper describes the fundamentals of Microsoft Azure Virtual Machine and provides a walk-through of planning and implementation considerations for SAP NetWeaver installations in Azure and as such should be the document to read before starting actual deployments of SAP NetWeaver on Azure. The paper complements the SAP Installation Documentation and SAP Notes which represent the primary resources for installations and deployments of SAP software on given platforms.

Summary

Cloud Computing is a widely used term which is gaining more and more importance within the IT industry, from small companies up to large and multinational corporations.

Microsoft Azure is the Cloud Services Platform from Microsoft which offers a wide spectrum of new possibilities. Now customers are able to rapidly provision and de-provision applications as a service in the cloud, so they are not limited to technical or budgeting restrictions. Instead of investing time and budget into hardware infrastructure, companies can focus on the application, business processes and its benefits for customers and users.

With Microsoft Azure Virtual Machine Services, Microsoft offers a comprehensive Infrastructure as a Service (IaaS) platform. SAP NetWeaver based applications are supported on Azure Virtual Machines (IaaS). This whitepaper will describe how to plan and implement SAP NetWeaver based applications within Microsoft Azure as the platform of choice.

The paper itself will focus on two main aspects:

- The first part will describe two supported deployment patterns for SAP NetWeaver based applications on Azure. It will also describe general handling of Azure with SAP deployments in mind.
- The second part will detail implementing the two different scenarios described in the first part.

For additional resources see chapter [Resources](#) in this document.

Definitions upfront

Throughout the document we will use the following terms:

- IaaS: Infrastructure as a Service.
- PaaS: Platform as a Service.
- SaaS: Software as a Service.
- ARM : Azure Resource Manager
- SAP Component: an individual SAP application such as ECC, BW, Solution Manager or EP. SAP components can be based on traditional ABAP or Java technologies or a non-NetWeaver based application such as Business Objects.
- SAP Environment: one or more SAP components logically grouped to perform a business function such as

Development, QAS, Training, DR or Production.

- SAP Landscape: This refers to the entire SAP assets in a customer's IT landscape. The SAP landscape includes all production and non-production environments.
- SAP System: The combination of DBMS layer and application layer of e.g. a SAP ERP development system, SAP BW test system, SAP CRM production system, etc.. In Azure deployments it is not supported to divide these two layers between on-premises and Azure. This means an SAP system is either deployed on-premises or it is deployed in Azure. However, you can deploy the different systems of an SAP landscape into either Azure or on-premises. For example, you could deploy the SAP CRM development and test systems in Azure but the SAP CRM production system on-premises.
- Cloud-Only deployment: A deployment where the Azure subscription is not connected via a site-to-site or ExpressRoute connection to the on-premises network infrastructure. In common Azure documentation these kinds of deployments are also described as 'Cloud-Only' deployments. Virtual Machines deployed with this method are accessed through the internet and a public IP address and/or a public DNS name assigned to the VMs in Azure. For Microsoft Windows the on-premises Active Directory (AD) and DNS is not extended to Azure in these types of deployments. Hence the VMs are not part of the on-premises Active Directory. Same is true for Linux implementations using e.g. OpenLDAP + Kerberos.

NOTE

Cloud-Only deployments in this document is defined as complete SAP landscapes are running exclusively in Azure without extension of Active Directory / OpenLDAP or name resolution from on-premises into public cloud. Cloud-Only configurations are not supported for production SAP systems or configurations where SAP STMS or other on-premises resources need to be used between SAP systems hosted on Azure and resources residing on-premises.

- Cross-Premises: Describes a scenario where VMs are deployed to an Azure subscription that has site-to-site, multi-site or ExpressRoute connectivity between the on-premises datacenter(s) and Azure. In common Azure documentation, these kinds of deployments are also described as Cross-Premises scenarios. The reason for the connection is to extend on-premises domains, on-premises Active Directory / OpenLDAP and on-premises DNS into Azure. The on-premises landscape is extended to the Azure assets of the subscription. Having this extension, the VMs can be part of the on-premises domain. Domain users of the on-premises domain can access the servers and can run services on those VMs (like DBMS services). Communication and name resolution between VMs deployed on-premises and Azure deployed VMs is possible. This is the scenario we expect most SAP assets to be deployed in. See [this article](#) and [this](#) for more information.

NOTE

Cross-Premises deployments of SAP systems where Azure Virtual Machines running SAP systems are members of an on-premises domain are supported for production SAP systems. Cross-Premises configurations are supported for deploying parts or complete SAP landscapes into Azure. Even running the complete SAP landscape in Azure requires having those VMs being part of on-premises domain and ADS / OpenLDAP. In former versions of the documentation we talked about Hybrid-IT scenarios, where the term 'Hybrid' is rooted in the fact that there is a cross-premises connectivity between on-premises and Azure. Plus, the fact that the VMs in Azure are part of the on-premises Active Directory / OpenLDAP.

Some Microsoft documentation describes Cross-Premises scenarios a bit differently, especially for DBMS HA configurations. In the case of the SAP related documents, the Cross-Premises scenario just boils down to having a site-to-site or private (ExpressRoute) connectivity and the fact that the SAP landscape is distributed between on-premises and Azure.

Resources

The following additional guides are available for the topic of SAP deployments on Azure:

- [SAP NetWeaver on Azure Virtual Machines \(VMs\) – Planning and Implementation Guide \(this document\)](#)

- SAP NetWeaver on Azure Virtual Machines (VMs) – Deployment Guide
- SAP NetWeaver on Azure Virtual Machines (VMs) – DBMS Deployment Guide
- SAP NetWeaver on Azure Virtual Machines (VMs) – High Availability Deployment Guide

IMPORTANT

Wherever possible a link to the referring SAP Installation Guide is used (Reference InstGuide-01, see <http://service.sap.com/instguides>). When it comes to the prerequisites and installation process, the SAP NetWeaver Installation Guides should always be read carefully, as this document only covers specific tasks for SAP NetWeaver systems installed in a Microsoft Azure Virtual Machine.

The following SAP Notes are related to the topic of SAP on Azure:

| NOTE NUMBER | TITLE |
|-------------------------|--|
| 1928533 | SAP Applications on Azure: Supported Products and Sizing |
| 2015553 | SAP on Microsoft Azure: Support Prerequisites |
| 1999351 | Troubleshooting Enhanced Azure Monitoring for SAP |
| 2178632 | Key Monitoring Metrics for SAP on Microsoft Azure |
| 1409604 | Virtualization on Windows: Enhanced Monitoring |
| 2191498 | SAP on Linux with Azure: Enhanced Monitoring |
| 2243692 | Linux on Microsoft Azure (IaaS) VM: SAP license issues |
| 1984787 | SUSE LINUX Enterprise Server 12: Installation notes |
| 2002167 | Red Hat Enterprise Linux 7.x: Installation and Upgrade |

Please also read the [SCN Wiki](#) that contains all SAP Notes for Linux.

General default limitations and maximum limitations of Azure subscriptions can be found in [this article](#)

Possible Scenarios

SAP is often seen as one of the most mission-critical applications within enterprises. The architecture and operations of these applications is mostly very complex and ensuring that you meet requirements on availability and performance is important.

Thus enterprises have to think carefully about which applications can be run in a Public Cloud environment, independent of the chosen Cloud provider.

Possible system types for deploying SAP NetWeaver based applications within public cloud environments are listed below:

1. Medium sized production systems
2. Development systems
3. Testing systems
4. Prototype systems
5. Learning / Demonstration systems

In order to successfully deploy SAP systems into either Azure IaaS or IaaS in general, it is important to understand the significant differences between the offerings of traditional outsourcers or hosters and IaaS offerings. Whereas the traditional hoster or outsourcer will adapt infrastructure (network, storage and server type) to the workload a customer wants to host, it is instead the customer's responsibility to choose the right workload for IaaS deployments.

As a first step, customers need to verify the following items:

- The SAP supported VM types of Azure
- The SAP supported products/releases on Azure
- The supported OS and DBMS releases for the specific SAP releases in Azure
- SAPS throughput provided by different Azure SKUs

The answers to these questions can be read in SAP Note [1928533](#).

As a second step, Azure resource and bandwidth limitations need to be compared to actual resource consumption of on-premises systems. Therefore, customers need to be familiar with the different capabilities of the Azure types supported with SAP in the area of:

- CPU and memory resources of different VM types and
- IOPS bandwidth of different VM types and
- Network capabilities of different VM types.

Most of that data can be found [here](#)

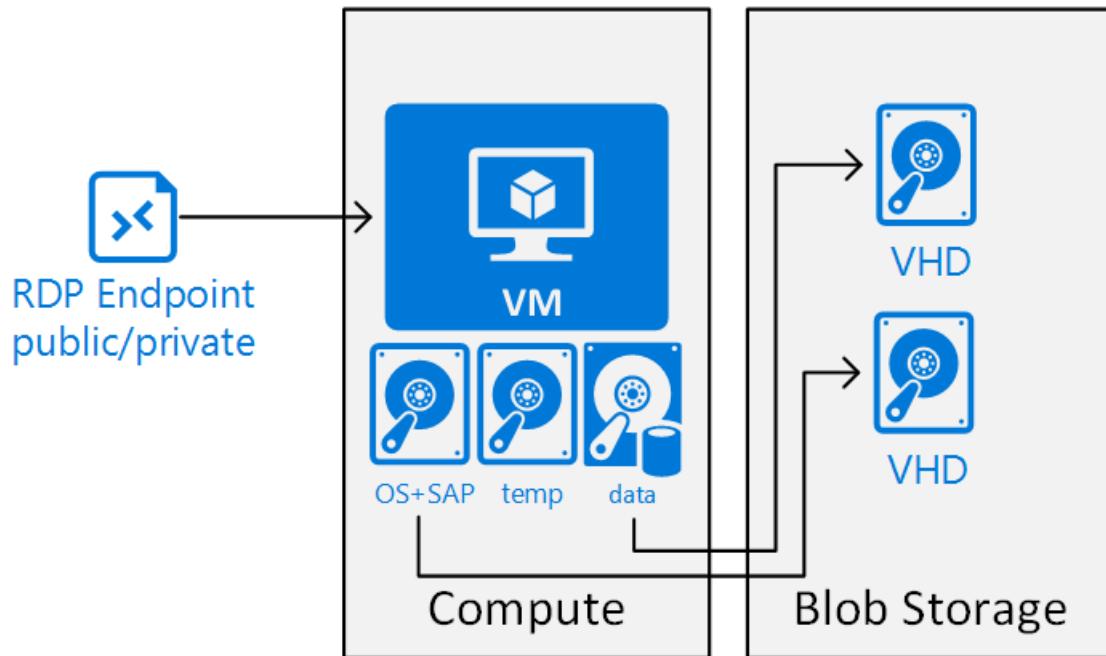
Keep in mind that the limits listed in the link above are upper limits. It does not mean that the limits for any of the resources, e.g. IOPS can be provided under all circumstances. The exceptions though are the CPU and memory resources of a chosen VM type. For the VM types supported by SAP, the CPU and memory resources are reserved and as such available at any point in time for consumption within the VM.

The Microsoft Azure platform like other IaaS platforms is a multi-tenant platform. This means that Storage, Network and other resources are shared between tenants. Intelligent throttling and quota logic is used to prevent one tenant from impacting the performance of another tenant (noisy neighbor) in a drastic way. Though logic in Azure tries to keep variances in bandwidth experienced small, highly shared platforms tend to introduce larger variances in resource/bandwidth availability than a lot of customers are used to in their on-premises deployments. As a result, you might experience different levels of bandwidth in regards to networking or storage I/O (the volume as well as latency) from minute to minute. The probability that a SAP system on Azure could experience larger variances than in an on-premises system needs to be taken into account.

A last step is to evaluate availability requirements. It can happen, that the underlying Azure infrastructure needs to get updated and requires the hosts running VMs to be rebooted. In these cases, VMs running on those hosts would be shut down and restarted as well. The timing of such maintenance is done during non-core business hours for a particular region but the potential window of a few hours during which a restart will occur is relatively wide. There are various technologies within the Azure platform that can be configured to mitigate some or all of the impact of such updates. Future enhancements of the Azure platform, DBMS and SAP application are designed to minimize the impact of such restarts.

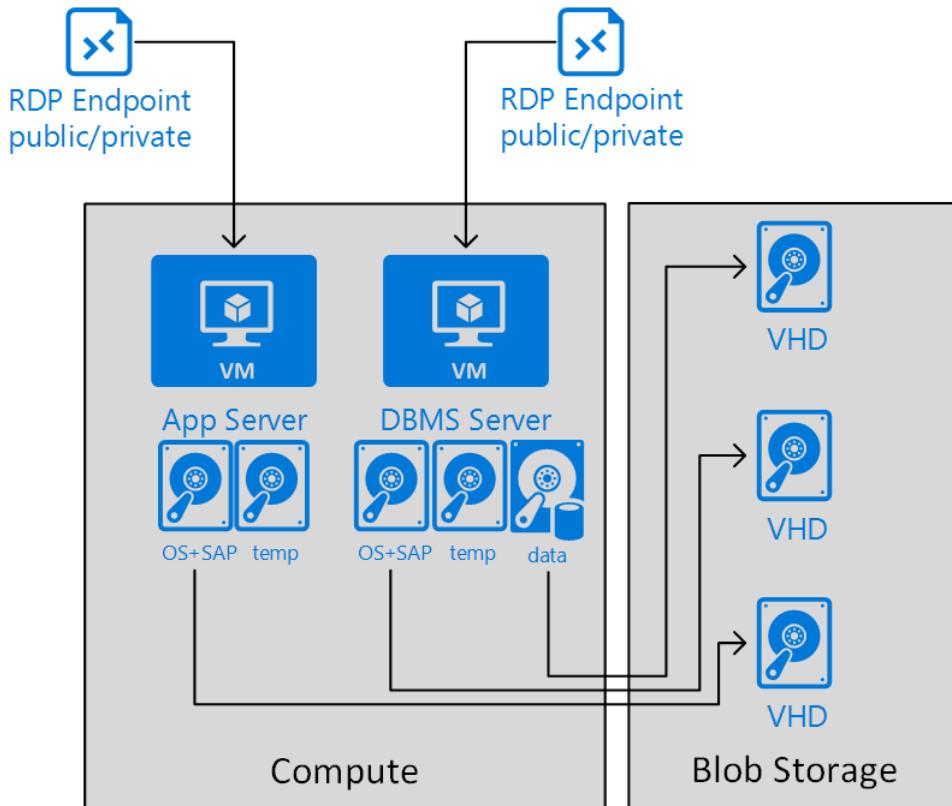
In order to successfully deploy a SAP system onto Azure, the on-premises SAP system(s) Operating System, Database and SAP applications must appear on the SAP Azure support matrix, fit within the resources the Azure infrastructure can provide and which can work with the Availability SLAs Microsoft Azure offers. As those systems are identified, you need to decide on one of the following two deployment scenarios.

Cloud-Only - Virtual Machine deployments into Azure without dependencies on the on-premises customer network



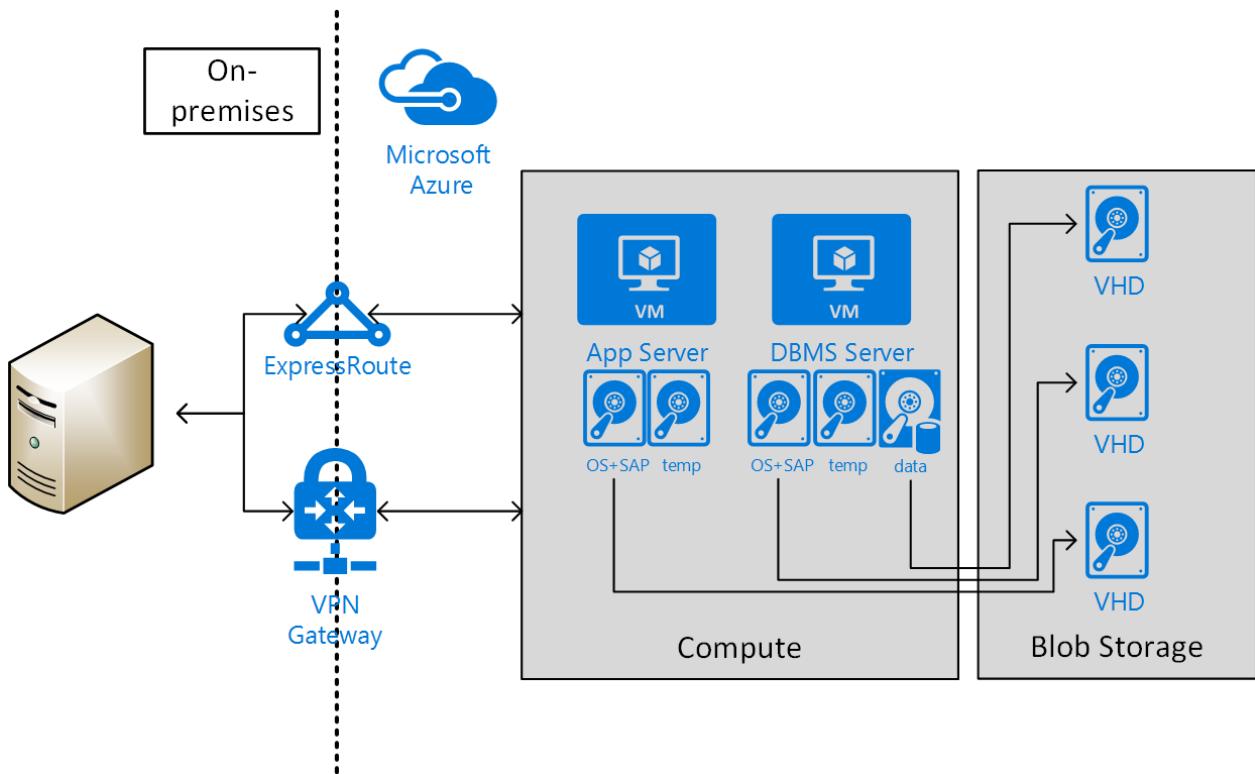
This scenario is typical for trainings or demo systems, where all the components of SAP and non-SAP software are installed within a single VM. Production SAP systems are not supported in this deployment scenario. In general, this scenario meets the following requirements:

- The VMs themselves are accessible over the public network. Direct network connectivity for the applications running within the VMs to the on-premises network of either the company owning the demos or trainings content or the customer is not necessary.
- In case of multiple VMs representing the trainings or demo scenario, network communications and name resolution needs to work between the VMs. But communications between the set of VMs need to be isolated so that several sets of VMs can be deployed side by side without interference.
- Internet connectivity is required for the end user to remote login into the VMs hosted in Azure. Depending on the guest OS, Terminal Services/RDS or VNC/ssh is used to access the VM to either fulfill the training tasks or perform the demos. If SAP ports such as 3200, 3300 & 3600 can also be exposed the SAP application instance can be accessed from any Internet connected desktop.
- The SAP system(s) (and VM(s)) represent a standalone scenario in Azure which only requires public internet connectivity for end user access and does not require a connection to other VMs in Azure.
- SAPGUI and a browser are installed and run directly on the VM.
- A fast reset of a VM to the original state and new deployment of that original state again is required.
- In the case of demo and training scenarios which are realized in multiple VMs, an Active Directory / OpenLDAP and/or DNS service is required for each set of VMs.



It is important to keep in mind that the VM(s) in each of the sets need to be deployed in parallel, where the VM names in each of the set are the same.

Cross-Premise - Deployment of single or multiple SAP VMs into Azure with the requirement of being fully integrated into the on-premises network



This scenario is a Cross-Premises scenario with many possible deployment patterns. It can be described as simply as running some parts of the SAP landscape on-premises and other parts of the SAP landscape on Azure. All aspects of the fact that part of the SAP components are running on Azure should be transparent for end users.

Hence the SAP Transport Correction System (STMS), RFC Communication, Printing, Security (like SSO), etc. will work seamlessly for the SAP systems running on Azure. But the Cross-Premises scenario also describes a scenario where the complete SAP landscape runs in Azure with the customer's domain and DNS extended into Azure.

NOTE

This is the deployment scenario which is supported for running productive SAP systems.

Read [this article](#) for more information on how to connect your on-premises network to Microsoft Azure

IMPORTANT

When we are talking about Cross-Premises scenarios between Azure and on-premises customer deployments, we are looking at the granularity of whole SAP systems. Scenarios which are *not supported* for Cross-Premises scenarios are:

- Running different layers of SAP applications in different deployment methods. E.g. running the DBMS layer on-premises, but the SAP application layer in VMs deployed as Azure VMs or vice versa.
- Some components of an SAP layer in Azure and some on-premises. E.g. splitting Instances of the SAP application layer between on-premises and Azure VMs.
- Distribution of VMs running SAP instances of one system over multiple Azure Regions is not supported.

The reason for these restrictions is the requirement for a very low latency high performance network within one SAP system, especially between the application instances and the DBMS layer of a SAP system.

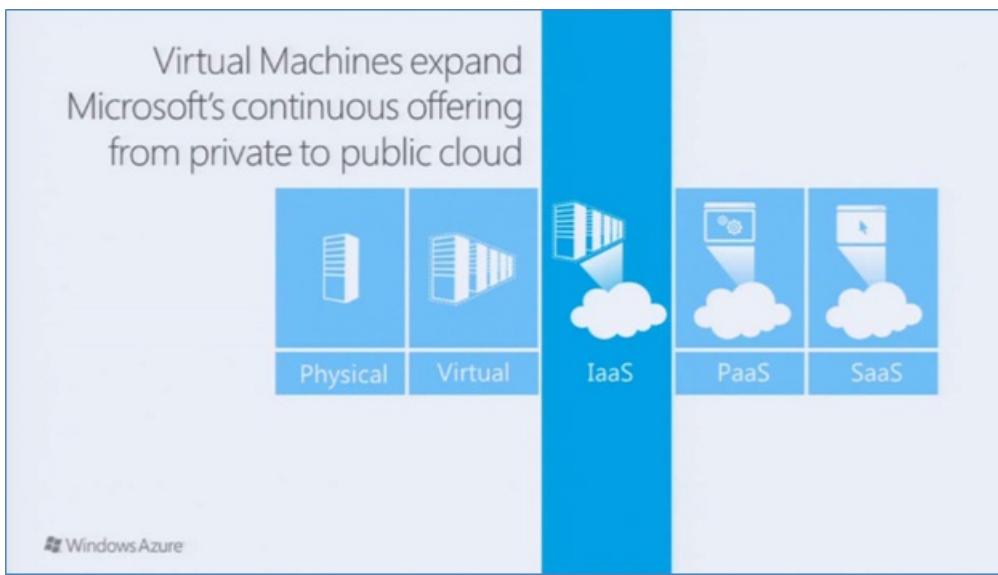
Supported OS and Database Releases

- Microsoft server software supported for Azure Virtual Machine Services is listed in this article: <http://support.microsoft.com/kb/2721672>.
- Supported operating system releases, database system releases supported on Azure Virtual Machine Services in conjunction with SAP software are documented in SAP Note [1928533](#).
- SAP applications and releases supported on Azure Virtual Machine Services are documented in SAP Note [1928533](#).
- Only 64Bit images are supported to run as Guest VMs in Azure for SAP scenarios. This also means that only 64-bit SAP applications and databases are supported.

Microsoft Azure Virtual Machine Services

The Microsoft Azure platform is an internet-scale cloud services platform hosted and operated in Microsoft data centers. The platform includes the Microsoft Azure Virtual Machine Services (Infrastructure as a Service, or IaaS) and a set of rich Platform as a Service (PaaS) capabilities.

The Azure platform reduces the need for up-front technology and infrastructure purchases. It simplifies maintaining and operating applications by providing on-demand compute and storage to host, scale and manage web application and connected applications. Infrastructure management is automated with a platform that is designed for high availability and dynamic scaling to match usage needs with the option of a pay-as-you-go pricing model.



With Azure Virtual Machine Services, Microsoft is enabling you to deploy custom server images to Azure as IaaS instances (see Figure 4). The Virtual Machines in Azure are based on Hyper-V virtual hard drives (VHD) and are able to run different operating systems as Guest OS.

From an operational perspective, the Azure Virtual Machine Service offers similar experiences as virtual machines deployed on premises. However, it has the significant advantage that you don't need to procure, administer and manage the infrastructure. Developers and Administrators have full control of the operating system image within these virtual machines. Administrators can logon remotely into those virtual machines to perform maintenance and troubleshooting tasks as well as software deployment tasks. In regard to deployment, the only restrictions are the sizes and capabilities of Azure VMs. These may not be as fine granular in configuration as this could be done on premises. There is a choice of VM types that represent a combination of:

- Number of vCPUs,
- Memory,
- Number of VHDs that can be attached,
- Network and Storage bandwidths.

The size and limitations of various different virtual machines sizes offered can be seen in a table in [this article](#)

As you will realize there are different families or series of virtual machines. As of Dec 2015, you can distinguish the following families of VMs:

- A0-A7 VM types: Not all of those are certified for SAP. First VM series that Azure IaaS got introduced with.
- A8-A11 VM types: High Performance computing instances. Running on different better performing compute hosts than other A-series VMs.
- D-Series VM types: Better performing than A0-A7. Not all of the VM types are certified with SAP.
- DS-Series VM types: use same hosts as D-series, but are able to connect to Azure Premium Storage (see chapter [Azure Premium Storage](#) of this document). Again not all VM types are certified with SAP.
- G-Series VM types: High memory VM types.
- GS-Series VM types : like G-Series but including the option to use Azure Premium Storage (see chapter [Azure Premium Storage](#) of this document). When using GS-Series VMs as database servers it's mandatory to use Premium Storage for DB data and transaction log files

You may find the same CPU and memory configurations in different VM series. Nevertheless, when you look up the throughput performance of these VMs out of the different series they might differ significantly. Despite having the same CPU and memory configuration. Reason is that the underlying host server hardware at the introduction of the different VM types had different throughput characteristics. Usually the difference shown in throughput performance also is reflected in the price of the different VMs.

Please note that not all different VM series might be offered in each one of the Azure Regions (for Azure Regions see next chapter). Also be aware that not all VMs or VM-Series are certified for SAP.

IMPORTANT

For the use of SAP NetWeaver based applications, only the subset of VM types and configurations listed in SAP Note [1928533](#) are supported.

Azure Regions

Microsoft allows to deploy Virtual Machines into so called 'Azure Regions'. An Azure Region may be one or multiple data centers that are located in close proximity. For most of the geopolitical regions in the world Microsoft has at least two Azure Regions. E.g. in Europe there is an Azure Region of 'North Europe' and one of 'West Europe'. Such two Azure Regions within a geopolitical region are separated by significant enough distance so that natural or technical disasters do not affect both Azure Regions in the same geopolitical region. Since Microsoft is steadily building out new Azure Regions in different geopolitical regions globally, the number of these regions is steadily growing and as of Dec 2015 reached the number of 20 Azure Regions with additional Regions announced already. You as a customer can deploy SAP systems into all these regions, including the two Azure Regions in China. For current up to date information about Azure regions see this website :

<https://azure.microsoft.com/regions/>

The Microsoft Azure Virtual Machine Concept

Microsoft Azure offers an Infrastructure as a Service (IaaS) solution to host Virtual Machines with similar functionalities as an on-premises virtualization solution. You are able to create Virtual Machines from within the Azure Portal, PowerShell or CLI, which also offer deployment and management capabilities.

Azure Resource Manager allows you to provision your applications using a declarative template. In a single template, you can deploy multiple services along with their dependencies. You use the same template to repeatedly deploy your application during every stage of the application life cycle.

More information about using ARM templates can be found here :

- [Deploy and manage virtual machines by using Azure Resource Manager templates and the Azure CLI](#)
- [Manage virtual machines using Azure Resource Manager and PowerShell](#)
- <https://azure.microsoft.com/documentation/templates/>

Another interesting feature is the ability to create images from Virtual Machines, which allows you to prepare certain repositories from which you are able to quickly deploy Virtual machine instances which meet your requirements.

More information about creating images from Virtual Machines can be found in [this article \(Windows\)](#) or in [this article \(Linux\)](#).

Fault Domains

Fault Domains represent a physical unit of failure, very closely related to the physical infrastructure contained in data centers, and while a physical blade or rack can be considered a Fault Domain, there is no direct one-to-one mapping between the two.

When you deploy multiple Virtual Machines as part of one SAP system in Microsoft Azure Virtual Machine Services, you can influence the Azure Fabric Controller to deploy your application into different Fault Domains, thereby meeting the requirements of the Microsoft Azure SLA. However, the distribution of Fault Domains over an Azure Scale Unit (collection of hundreds of Compute nodes or Storage nodes and networking) or the assignment of VMs to a specific Fault Domain is something over which you do not have direct control. In order to direct the Azure fabric controller to deploy a set of VMs over different Fault Domains, you need to assign an Azure Availability Set to the VMs at deployment time. For more information on Azure Availability Sets, see chapter [Azure Availability Sets](#) in this document.

Upgrade Domains

Upgrade Domains represent a logical unit that help to determine how a VM within an SAP system, that consists of SAP instances running in multiple VMs, will be updated. When an upgrade occurs, Microsoft Azure goes through the process of updating these Upgrade Domains one by one. By spreading VMs at deployment time over different Upgrade Domains you can protect your SAP system partly from potential downtime. In order to force Azure to deploy the VMs of an SAP system spread over different Upgrade Domains, you need to set a specific attribute at deployment time of each VM. Similar to Fault Domains, an Azure Scale Unit is divided into multiple Upgrade Domains. In order to direct the Azure fabric controller to deploy a set of VMs over different Upgrade Domains, you need to assign an Azure Availability Set to the VMs at deployment time. For more information on Azure Availability Sets, see chapter [Azure Availability Sets](#) below.

Azure Availability Sets

Azure Virtual Machines within one Azure Availability Set will be distributed by the Azure Fabric Controller over different Fault and Upgrade Domains. The purpose of the distribution over different Fault and Upgrade Domains is to prevent all VMs of an SAP system from being shut down in the case of infrastructure maintenance or a failure within one Fault Domain. By default, VMs are not part of an Availability Set. The participation of a VM in an Availability Set is defined at deployment time or later on by a reconfiguration and re-deployment of a VM.

To understand the concept of Azure Availability Sets and the way Availability Sets relate to Fault and Upgrade Domains, please read [this article](#)

To define availability sets for ARM via a json template see [the rest-api specs](#) and search for "availability".

Storage: Microsoft Azure Storage and Data Disks

Microsoft Azure Virtual Machines utilize different storage types. When implementing SAP on Azure Virtual Machine Services it is important to understand the differences between these two main types of storage:

- Non-Persistent, volatile storage.
- Persistent storage.

The non-persistent storage is directly attached to the running Virtual Machines and resides on the compute nodes themselves – the local instance storage (temporary storage). The size depends on the size of the Virtual Machine chosen when the deployment started. This storage type is volatile and therefore the disk is initialized when a Virtual Machine instance is restarted. Typically, the pagefile for the operating system is located on this temporary disk.



On Windows VMs the temp drive is mounted as drive D:\ in a deployed VM.



On Linux VMs it's mounted as /mnt/resource or /mnt. See more details here :

- [How to Attach a Data Disk to a Linux Virtual Machine](#)
- <http://blogs.msdn.com/b/mast/archive/2013/12/07/understanding-the-temporary-drive-on-windows-azure-virtual-machines.aspx>

The actual drive is volatile because it is getting stored on the host server itself. If the VM moved in a redeployment (e.g. due to maintenance on the host or shutdown and restart) the content of the drive is lost. Therefore, it is not an option to store any important data on this drive. The type of media used for this type of storage differs between different VM series with very different performance characteristics which as of June 2015 look like:

- A5-A7: Very limited performance. Not recommended for anything beyond page file

- A8-A11: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput.
- D-Series: Very good performance characteristics with some thousand IOPS and >1GB/sec throughput.
- DS-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput.
- G-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput.
- GS-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput.

Statements above are applying to the VM types that are certified with SAP. The VM-series with excellent IOPS and throughput qualify for leverage by some DBMS features. Please see the [DBMS Deployment Guide](#) for more details.

Microsoft Azure Storage provides persisted storage and the typical levels of protection and redundancy seen on SAN storage. Disks based on Azure Storage are virtual hard disk (VHDs) located in the Azure Storage Services. The local OS-Disk (Windows C:\, Linux / (/dev/sda1)) is stored on the Azure Storage, and additional Volumes/Disks mounted to the VM get stored there, too.

It is possible to upload an existing VHD from on-premises or create empty ones from within Azure and attach those to deployed VMs. Those VHDs are referenced as Azure Disks.

After creating or uploading a VHD into Azure Storage, it is possible to mount and attach those to an existing Virtual Machine and to copy existing (unmounted) VHD.

As those VHDs are persisted, data and changes within those are safe when rebooting and recreating a Virtual Machine instance. Even if an instance is deleted, these VHDs stay safe and can be redeployed or in case of non-OS disks can be mounted to other VMs.

Within the network of Azure Storage different redundancy levels can be configured:

- Minimum level that can be selected is 'local redundancy', which is equivalent to three-replica of the data within the same data center of an Azure Region (see chapter [Azure Regions](#)).
- Zone redundant storage which will spread the three images over different data centers within the same Azure Region.
- Default redundancy level is geographic redundancy which asynchronously replicates the content into another 3 images of the data into another Azure Region which is hosted in the same geopolitical region.

Also see the table on top of this article in regards to the different redundancy options:

<https://azure.microsoft.com/pricing/details/storage/>

More information in regards to Azure Storage can be found here:

- <https://azure.microsoft.com/documentation/services/storage/>
- <https://azure.microsoft.com/services/site-recovery>
- <https://msdn.microsoft.com/library/windowsazure/ee691964.aspx>
- <https://blogs.msdn.com/b/azuresecurity/archive/2015/11/17/azure-disk-encryption-for-linux-and-windows-virtual-machines-public-preview.aspx>

Azure Standard Storage

Azure Standard BLOB storage was the type of storage available when Azure IaaS was released. There were IOPS quotas enforced per single VHD. Latency experienced was not in the same class as SAN/NAS devices typically deployed for high-end SAP systems hosted on-premises. Nevertheless, the Azure Standard Storage proved sufficient for many hundreds SAP systems meanwhile deployed in Azure.

Azure Standard Storage is charged based on the actual data that is stored, the volume of storage transactions, outbound data transfers and redundancy option chosen. Many VHDs can be created at the maximum 1TB in size, but as long as those remain empty there is no charge. If you then fill one VHD with 100GB each, you will be charged for storing 100GB and not for the nominal size the VHD got created with.

Azure Premium Storage

In April 2015 Microsoft introduced Azure Premium Storage. Premium Storage got introduced with the goal to provide:

- Better I/O latency.
- Better throughput.
- Less variability in I/O latency.

For that purpose, a lot of changes were introduced of which the two most significant are:

- Usage of SSD disks in the Azure Storage nodes
- A new read cache that is backed by the local SSD of an Azure compute node

In opposite to Standard storage where capabilities did not change dependent on the size of the disk (or VHD), Premium Storage currently has 3 different disk categories which are shown at the end of this article before the FAQ section : <https://azure.microsoft.com/pricing/details/storage/>

You see that IOPS/VHD and disk throughput/VHD are dependent on the size category of the disks

Cost basis in the case of Premium Storage is not the actual data volume stored in such VHDs, but the size category of such a VHD, independent of the amount of the data that is stored within the VHD.

You also can create VHDs on Premium Storage that are not directly mapping into the size categories shown. This may be the case, especially when copying VHDs from Standard Storage into Premium Storage. In such cases a mapping to the next largest Premium Storage disk option is performed.

Please be aware that only certain VM series can benefit from the Azure Premium Storage. As of Dec 2015, these are the DS- and GS-series. The DS-series is basically the same as D-series with the exception that DS-series has the ability to mount Premium Storage based VMs additionally to VHDs that are hosted on Azure Standard Storage. Same thing is valid for G-series compared to GS-series.

If you are checking out the part of the DS-series VMs in [this article](#) you also will realize that there are data volume limitations to Premium Storage VHDs on the granularity of the VM level. Different DS-series or GS-series VMs also have different limitations in regards to the number of VHDs that can be mounted. These limits are documented in the article mentioned above as well. But in essence it means that if you e.g. mount 32 x P30 disks/VHDs to a single DS14 VM you can NOT get 32 x the maximum throughput of a P30 disk. Instead the maximum throughput on VM level as documented in the article will limit data throughput.

More information on Premium Storage can be found here: <http://azure.microsoft.com/blog/2015/04/16/azure-premium-storage-now-generally-available-2>

Azure Storage Accounts

When deploying services or VMs in Azure, deployment of VHDs and VM Images must be organized in units called Azure Storage Accounts. When planning an Azure deployment, you need to carefully consider the restrictions of Azure. On the one side, there is a limited number of Storage Accounts per Azure subscription. Although each Azure Storage Account can hold a large number of VHD files, there is a fixed limit on the total IOPS per Storage Account. When deploying hundreds of SAP VMs with DBMS systems creating significant IO calls, it is recommended to distribute high IOPS DBMS VMs between multiple Azure Storage Accounts. Care must be taken not to exceed the current limit of Azure Storage Accounts per subscription. Because storage is a vital part of the database deployment for an SAP system, this concept is discussed in more detail in the already referenced [DBMS Deployment Guide](#).

More information about Azure Storage Accounts can be found in [this article](#). Reading this article, you will realize that there are differences in the limitations between Azure Standard Storage Accounts and Premium Storage Accounts. Major differences are the volume of data that can be stored within such a Storage Account. In Standard Storage the volume is a magnitude larger than with Premium Storage. On the other side the Standard Storage Account is severely limited in IOPS (see column 'Total Request Rate'), whereas the Azure Premium Storage Account has no such limitation. We will discuss details and results of these differences when discussing the

deployments of SAP systems, especially the DBMS servers.

Within a Storage Account, you have the possibility to create different containers for the purpose of organizing and categorizing different VHDs. These containers are usually used to e.g. separate VHDs of different VMs. There are no performance implications in using just one container or multiple containers underneath a single Azure Storage Account.

Within Azure a VHD name follows the following naming connection that needs to provide a unique name for the VHD within Azure:

```
http(s)://<storage account name>.blob.core.windows.net/<container name>/<vhd name>
```

As mentioned the string above needs to uniquely identify the VHD that is stored on Azure Storage.

Microsoft Azure Networking

Microsoft Azure will provide a network infrastructure which allows the mapping of all scenarios which we want to realize with SAP software. The capabilities are:

- Access from the outside, directly to the VMs via Windows Terminal Services or ssh/VNC
- Access to services and specific ports used by applications within the VMs
- Internal Communication and Name Resolution between a group of VMs deployed as Azure VMs
- Cross-Premises Connectivity between a customer's on-premises network and the Azure network
- Cross Azure Region or data center connectivity between Azure sites

More information can be found here: <https://azure.microsoft.com/documentation/services/virtual-network/>

There are a lot of different possibilities to configure name and IP resolution in Azure. In this document, Cloud-Only scenarios rely on the default of using Azure DNS (in contrast to defining an own DNS service). There is also a new Azure DNS service which can be used instead of setting up your own DNS server. More information can be found in [this article](#) and on [this page](#).

For Cross-Premises scenarios we are relying on the fact that the on-premises AD/OpenLDAP/DNS has been extended via VPN or private connection to Azure. For certain scenarios as documented here, it might be necessary to have an AD/OpenLDAP replica installed in Azure.

Because networking and name resolution is a vital part of the database deployment for an SAP system, this concept is discussed in more detail in the [DBMS Deployment Guide](#).

Azure Virtual Networks

By building up an Azure Virtual Network you can define the address range of the private IP addresses allocated by Azure DHCP functionality. In Cross-Premises scenarios, the IP address range defined will still be allocated using DHCP by Azure. However, Domain Name resolution will be done on-premises (assuming that the VMs are a part of an on-premises domain) and hence can resolve addresses beyond different Azure Cloud Services.

Every Virtual Machine in Azure needs to be connected to a Virtual Network.

More details can be found in [this article](#) and on [this page](#).

NOTE

By default, once a VM is deployed you cannot change the Virtual Network configuration. The TCP/IP settings must be left to the Azure DHCP server. Default behavior is Dynamic IP assignment.

The MAC address of the virtual network card may change e.g. after re-size and the Windows or Linux guest OS will pick up the new network card and will automatically use DHCP to assign the IP and DNS addresses in this case.

Static IP Assignment

It is possible to assign fixed or reserved IP addresses to VMs within an Azure Virtual Network. Running the VMs in an Azure Virtual Network opens a great possibility to leverage this functionality if needed or required for some scenarios. The IP assignment remains valid throughout the existence of the VM, independent of whether the VM is running or shutdown. As a result, you need to take the overall number of VMs (running and stopped VMs) into account when defining the range of IP addresses for the Virtual Network. The IP address remains assigned either until the VM and its Network Interface is deleted or until the IP address gets de-assigned again. Please see detailed information in [this article](#).

Multiple NICs per VM

You can define multiple virtual network interface cards (vNIC) for an Azure Virtual Machine. With the ability to have multiple vNICs you can start to set up network traffic separation where e.g. client traffic is routed through one vNIC and backend traffic is routed through a second vNIC. Dependent on the type of VM there are different limitations in regards to the number of vNICs. Exact details, functionality and restrictions can be found in these articles:

- [Create a VM with multiple NICs](#)
- [Deploy multi NIC VMs using a template](#)
- [Deploy multi NIC VMs using PowerShell](#)
- [Deploy multi NIC VMs using the Azure CLI](#)

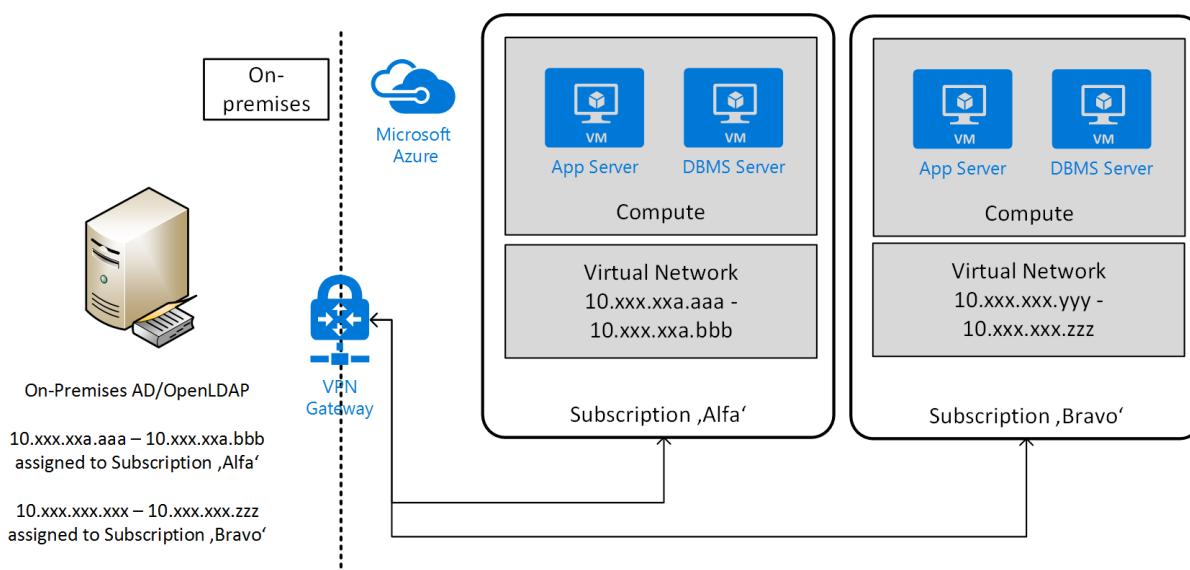
Site-to-Site Connectivity

Cross-Premises is Azure VMs and On-Premises linked with a transparent and permanent VPN connection. It is expected to become the most common SAP deployment pattern in Azure. The assumption is that operational procedures and processes with SAP instances in Azure should work transparently. This means you should be able to print out of these systems as well as use the SAP Transport Management System (TMS) to transport changes from a development system in Azure to a test system which is deployed on-premises. More documentation around site-to-site can be found in [this article](#)

VPN Tunnel Device

In order to create a site-to-site connection (on-premises data center to Azure data center), you will need to either obtain and configure a VPN device, or use Routing and Remote Access Service (RRAS) which was introduced as a software component with Windows Server 2012.

- [Create a virtual network with a site-to-site VPN connection using PowerShell](#)
- [About VPN devices for Site-to-Site VPN Gateway connections](#)
- [VPN Gateway FAQ](#)



The Figure above shows two Azure subscriptions have IP address subranges reserved for usage in Virtual Networks in Azure. The connectivity from the on-premises network to Azure is established via VPN.

Point-to-Site VPN

Point-to-site VPN requires every client machine to connect with its own VPN into Azure. For the SAP scenarios we are looking at, point-to-site connectivity is not practical. Therefore, no further references will be given to point-to-site VPN connectivity.

Multi-Site VPN

Azure also nowadays offers the possibility to create Multi-Site VPN connectivity for one Azure subscription. Previously a single subscription was limited to one site-to-site VPN connection. This limitation went away with Multi-Site VPN connections for a single subscription. This makes it possible to leverage more than one Azure Region for a specific subscription through Cross-Premises configurations.

For more documentation please see [this article](#)

VNet to VNet Connection

Using Multi-Site VPN, you need to configure a separate Azure Virtual Network in each of the regions. However very often you have the requirement that the software components in the different regions should communicate with each other. Ideally this communication should not be routed from one Azure Region to on-premises and from there to the other Azure Region. To shortcut, Azure offers the possibility to configure a connection from one Azure Virtual Network in one region to another Azure Virtual Network hosted in another region. This functionality is called VNet-to-VNet connection. More details on this functionality can be found here: <https://azure.microsoft.com/documentation/articles/vpn-gateway-vnet-vnet-rm-ps/>.

Private Connection to Azure – ExpressRoute

Microsoft Azure ExpressRoute allows the creation of private connections between Azure data centers and either the customer's on-premises infrastructure or in a co-location environment. ExpressRoute is offered by various MPLS (packet switched) VPN providers or other Network Service Providers. ExpressRoute connections do not go over the public Internet. ExpressRoute connections offer higher security, more reliability through multiple parallel circuits, faster speeds and lower latencies than typical connections over the Internet.

Find more details on Azure ExpressRoute and offerings here:

- <https://azure.microsoft.com/documentation/services/expressroute/>
- <https://azure.microsoft.com/pricing/details/expressroute/>
- <https://azure.microsoft.com/documentation/articles/expressroute-faqs/>

Express Route enables multiple Azure subscriptions through one ExpressRoute circuit as documented here

- <https://azure.microsoft.com/documentation/articles/expressroute-howto-linkvnet-arm/>
- <https://azure.microsoft.com/documentation/articles/expressroute-howto-circuit-arm/>

Forced tunneling in case of Cross-Premise

For VMs joining on-premises domains through site-to-site, point-to-site or ExpressRoute, you need to make sure that the Internet proxy settings are getting deployed for all the users in those VMs as well. By default, software running in those VMs or users using a browser to access the internet would not go through the company proxy, but would connect straight through Azure to the internet. But even the proxy setting is not a 100% solution to direct the traffic through the company proxy since it is responsibility of software and services to check for the proxy. If software running in the VM is not doing that or an administrator manipulates the settings, traffic to the Internet can be detoured again directly through Azure to the Internet.

In order to avoid this, you can configure Forced Tunneling with site-to-site connectivity between on-premises and Azure. The detailed description of the Forced Tunneling feature is published here

<https://azure.microsoft.com/documentation/articles/vpn-gateway-forced-tunneling-rm/>

Forced Tunneling with ExpressRoute is enabled by customers advertising a default route via the ExpressRoute

BGP peering sessions.

Summary of Azure Networking

This chapter contained many important points about Azure Networking. Here is a summary of the main points:

- Azure Virtual Networks allows to set up the network according to your own needs
- Azure Virtual Networks can be leveraged to assign IP address ranges to VMs or assign fixed IP addresses to VMs
- To set up a Site-To-Site or Point-To-Site connection you need to create an Azure Virtual Network first
- Once a virtual machine has been deployed it is no longer possible to change the Virtual Network assigned to the VM

Quotas in Azure Virtual Machine Services

We need to be clear about the fact that the storage and network infrastructure is shared between VMs running a variety of services in the Azure infrastructure. And just as in the customer's own data centers, over-provisioning of some of the infrastructure resources does take place to a degree. The Microsoft Azure Platform uses disk, CPU, network and other quotas to limit the resource consumption and to preserve consistent and deterministic performance. The different VM types (A5, A6, etc) have different quotas for the number of disks, CPU, RAM and Network.

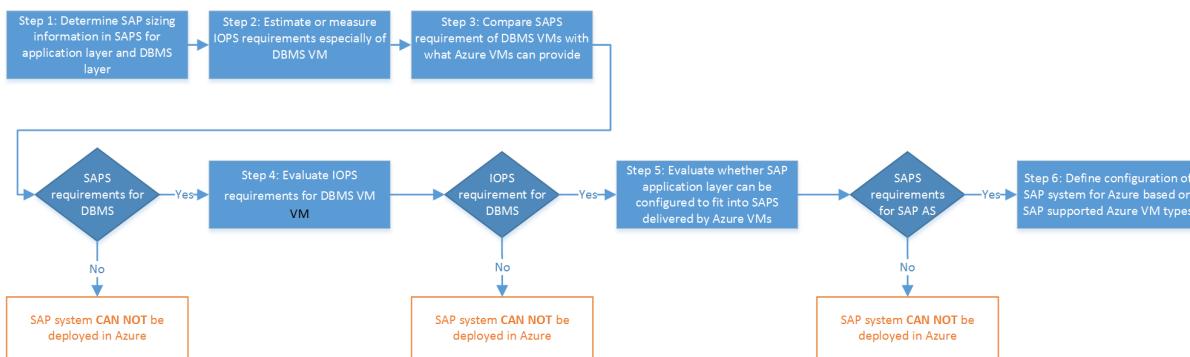
NOTE

CPU and memory resources of the VM types supported by SAP are pre-allocated on the host nodes. This means that once the VM is deployed, the resources on the host will be available as defined by the VM type.

When planning and sizing SAP on Azure solutions the quotas for each virtual machine size must be considered. The VM quotas are described [here](#).

The quotas described represent the theoretical maximum values. The limit of IOPS per VHD may be achieved with small IOs (8kb) but possibly may not be achieved with large IOs (1Mb). The IOPS limit is enforced on the granularity of single VHDs.

As a rough decision tree to decide whether an SAP system fits into Azure Virtual Machine Services and its capabilities or whether an existing system needs to be configured differently in order to deploy the system on Azure, the decision tree below can be used:



Step 1: The most important information to start with is the SAPS requirement for a given SAP system. The SAPS requirements need to be separated out into the DBMS part and the SAP application part, even if the SAP system is already deployed on-premises in a 2-tier configuration. For existing systems, the SAPS related to the hardware in use often can be determined or estimated based on existing SAP benchmarks. The results can be found here: <http://global.sap.com/campaigns/benchmark/index.epx>. For newly deployed SAP systems, you should have gone through a sizing exercise which should determine the SAPS requirements of the system. See also this blog and attached document for SAP sizing on Azure : <http://blogs.msdn.com/b/saponsqlserver/archive/2015/12/01/new->

Step 2: For existing systems, the I/O volume and I/O operations per second on the DBMS server should be measured. For newly planned systems, the sizing exercise for the new system also should give rough ideas of the I/O requirements on the DBMS side. If unsure, you eventually need to conduct a Proof of Concept.

Step 3: Compare the SAPS requirement for the DBMS server with the SAPS the different VM types of Azure can provide. The information on SAPS of the different Azure VM types is documented in SAP Note [1928533](#). The focus should be on the DBMS VM first since the database layer is the layer in a SAP NetWeaver system that does not scale out in the majority of deployments. In contrast, the SAP application layer can be scaled out. If none of the SAP supported Azure VM types can deliver the required SAPS, the workload of the planned SAP system can't be run on Azure. You either need to deploy the system on-premises or you need to change the workload volume for the system.

Step 4: As documented [here](#), Azure enforces an IOPS quota per VHD independent whether you use Standard Storage or Premium Storage. Dependent on the VM type, the number of VHDs which can be mounted varies. As a result, you can calculate a maximum IOPS number that can be achieved with each of the different VM types. Dependent on the database file layout, you can stripe VHDs to become one volume in the guest OS. However, if the current IOPS volume of a deployed SAP system exceeds the calculated limits of the largest VM type of Azure and if there is no chance to compensate with more memory, the workload of the SAP system can be impacted severely. In such cases, you can hit a point where you should not deploy the system on Azure.

Step 5: Especially in SAP systems which are deployed on-premises in 2-Tier configurations, the chances are that the system might need to be configured on Azure in a 3-Tier configuration. In this step, you need to check whether there is a component in the SAP application layer which can't be scaled out and which would not fit into the CPU and memory resources the different Azure VM types offer. If there indeed is such a component, the SAP system and its workload can't be deployed into Azure. But if you can scale-out the SAP application components into multiple Azure VMs, the system can be deployed into Azure.

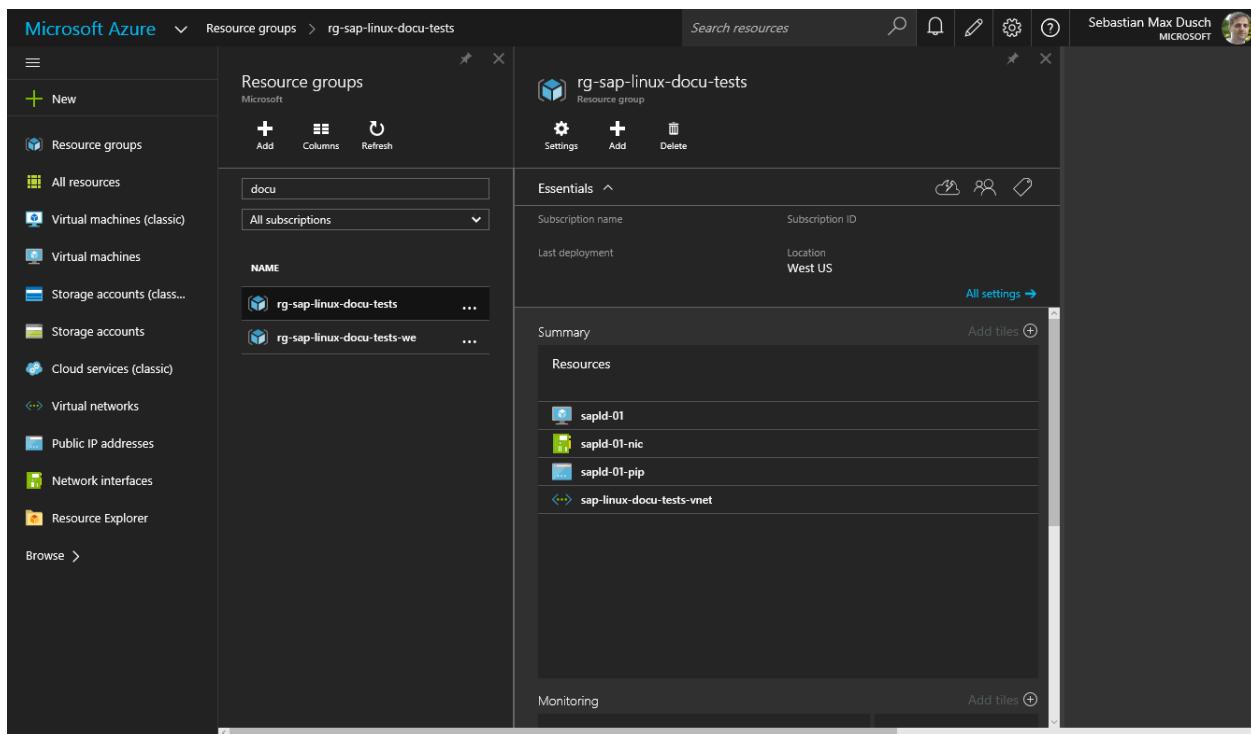
Step 6: If the DBMS and SAP application layer components can be run in Azure VMs, the configuration needs to be defined with regard to:

- Number of Azure VMs
- VM types for the individual components
- Number of VHDs in DBMS VM to provide enough IOPS

Managing Azure Assets

Azure Portal

The Azure Portal is one of three interfaces to manage Azure VM deployments. The basic management tasks, like deploying VMs from images, can be done through the Azure Portal. In addition, the creation of Storage Accounts, Virtual Networks and other Azure components are also tasks the Azure Portal can handle very well. However, functionality like uploading VHDs from on-premises to Azure or copying a VHD within Azure are tasks which require either third party tools or administration through PowerShell or CLI.



Administration and configuration tasks for the Virtual Machine instance are possible from within the Azure Portal.

Besides restarting and shutting down a Virtual Machine you can also attach, detach and create data disks for the Virtual Machine instance, to capture the instance for image preparation and configure the size of the Virtual Machine instance.

The Azure Portal provides basic functionality to deploy and configure VMs and many other Azure services. However not all available functionality is covered by the Azure Portal. In the Azure Portal, it's not possible to perform tasks like:

- Uploading VHDs to Azure
- Copying VMs

Management via Microsoft Azure PowerShell cmdlets

Windows PowerShell is a powerful and extensible framework that has been widely adopted by customers deploying larger numbers of systems in Azure. After the installation of PowerShell cmdlets on a desktop, laptop or dedicated management station, the PowerShell cmdlets can be run remotely.

The process to enable a local desktop/laptop for the usage of Azure PowerShell cmdlets and how to configure those for the usage with the Azure subscription(s) is described in [this article](#).

More detailed steps on how to install, update and configure the Azure PowerShell cmdlets can also be found in [this chapter of the Deployment Guide](#).

Customer experience so far has been that PowerShell (PS) is certainly the more powerful tool to deploy VMs and to create custom steps in the deployment of VMs. All of the customers running SAP instances in Azure are using PS cmdlets to supplement management tasks they do in the Azure Portal or are even using PS cmdlets exclusively to manage their deployments in Azure. Since the Azure specific cmdlets share the same naming convention as the more than 2000 Windows related cmdlets, it is an easy task for Windows administrators to leverage those cmdlets.

See example here : <http://blogs.technet.com/b/keithmayer/archive/2015/07/07/18-steps-for-end-to-end-iaas-provisioning-in-the-cloud-with-azure-resource-manager-arm-powershell-and-desired-state-configuration-dsc.aspx>

Deployment of the Azure Monitoring Extension for SAP (see chapter [Azure Monitoring Solution for SAP](#) in this

document) is only possible via PowerShell or CLI. Therefore it is mandatory to setup and configure PowerShell or CLI when deploying or administering an SAP NetWeaver system in Azure.

As Azure provides more functionality, new PS cmdlets are going to be added that requires an update of the cmdlets. Therefore it makes sense to check the Azure Download site at least once the month

<https://azure.microsoft.com/downloads/> for a new version of the cmdlets. The new version will just be installed on top of the older version.

For a general list of Azure related PowerShell commands check here:

<https://msdn.microsoft.com/library/azure/dn708514.aspx>.

Management via Microsoft Azure CLI commands

For customers who use Linux and want to manage Azure resources Powershell might not be an option. Microsoft offers Azure CLI as an alternative. The Azure CLI provides a set of open source, cross-platform commands for working with the Azure Platform. The Azure CLI provides much of the same functionality found in the Azure portal.

For information about installation, configuration and how to use CLI commands to accomplish Azure tasks see

- [Install the Azure CLI](#)
- [Deploy and manage virtual machines by using Azure Resource Manager templates and the Azure CLI](#)
- [Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager](#)

Please also read chapter [Azure CLI for Linux VMs](#) in the [Deployment Guide](#) on how to use Azure CLI to deploy the Azure Monitoring Extension for SAP.

Different ways to deploy VMs for SAP in Azure

In this chapter you will learn the different ways to deploy a VM in Azure. Additional preparation procedures, as well as handling of VHDs and VMs in Azure are covered in this chapter.

Deployment of VMs for SAP

Microsoft Azure offers multiple ways to deploy VMs and associated disks. Thus it is very important to understand the differences since preparations of the VMs might differ depending on the method of deployment. In general, we will take a look at the following scenarios:

Moving a VM from on-premises to Azure with a non-generalized disk

You plan to move a specific SAP system from on-premises to Azure. This can be done by uploading the VHD which contains the OS, the SAP Binaries and DBMS binaries plus the VHDs with the data and log files of the DBMS to Azure. In contrast to [scenario #2 below](#), you keep the hostname, SAP SID and SAP user accounts in the Azure VM as they were configured in the on-premises environment. Therefore, generalizing the image is not necessary. Please see chapters [Preparation for moving a VM from on-premises to Azure with a non-generalized disk](#) of this document for on-premises preparation steps and upload of non-generalized VMs or VHDs to Azure. Please read chapter [Scenario 3: Moving a VM from on-premises using a non-generalized Azure VHD with SAP](#) in the [Deployment Guide](#) for detailed steps of deploying such an image in Azure.

Deploying a VM with a customer specific image

Due to specific patch requirements of your OS or DBMS version, the provided images in the Azure Marketplace might not fit your needs. Therefore, you might need to create a VM using your own 'private' OS/DBMS VM image which can be deployed several times afterwards. To prepare such a 'private' image for duplication, the following items have to be considered :



The Windows settings (like Windows SID and hostname) must be abstracted/generalized on the on-premises VM via the sysprep command.



Please follow the steps described in these articles for [SUSE](#) or [Red Hat](#) to prepare a VHD to be uploaded to Azure.

If you have already installed SAP content in your on-premises VM (especially for 2-Tier systems), you can adapt the SAP system settings after the deployment of the Azure VM through the instance rename procedure supported by the SAP Software Provisioning Manager (SAP Note [1619720](#)). See chapters [Preparation for deploying a VM with a customer specific image for SAP](#) and [Uploading a VHD from on-premises to Azure](#) of this document for on-premises preparation steps and upload of a generalized VM to Azure. Please read chapter [Scenario 2: Deploying a VM with a custom image for SAP](#) in the [Deployment Guide](#) for detailed steps of deploying such an image in Azure.

Deploying a VM out of the Azure Marketplace

You would like to use a Microsoft or 3rd party provided VM image from the Azure Marketplace to deploy your VM. After you deployed your VM in Azure, you follow the same guidelines and tools to install the SAP software and/or DBMS inside your VM as you would do in an on-premises environment. For more detailed deployment description, please see chapter [Scenario 1: Deploying a VM out of the Azure Marketplace for SAP](#) in the [Deployment Guide](#).

Preparing VMs with SAP for Azure

Before uploading VMs into Azure you need to make sure the VMs and VHDs fulfill certain requirements. There are small differences depending on the deployment method that is used.

Preparation for moving a VM from on-premises to Azure with a non-generalized disk

A common deployment method is to move an existing VM which runs an SAP system from on-premises to Azure. That VM and the SAP system in the VM just should run in Azure using the same hostname and very likely the same SAP SID. In this case the guest OS of VM should not be generalized for multiple deployments. If the on-premises network got extended into Azure (see chapter [Cross-Premise - Deployment of single or multiple SAP VMs into Azure with the requirement of being fully integrated into the on-premises network](#) in this document), then even the same domain accounts can be used within the VM as those were used before on-premises.

Requirements when preparing your own Azure VM Disk are:

- Originally the VHD containing the operating system could have a maximum size of 127GB only. This limitation got eliminated at the end of March 2015. Now the VHD containing the operating system can be up to 1TB in size as any other Azure Storage hosted VHD as well.
- It needs to be in the fixed VHD format. Dynamic VHDs or VHDs in VHDx format are not yet supported on Azure. Dynamic VHDs will be converted to static VHDs when you upload the VHD with PowerShell commandlets or CLI
- VHDs which are mounted to the VM and should be mounted again in Azure to the VM need to be in a fixed VHD format as well. The same size limit of the OS disk applies to data disks as well. VHDs can have a maximum size of 1TB. Dynamic VHDs will be converted to static VHDs when you upload the VHD with PowerShell commandlets or CLI
- Add another local account with administrator privileges which can be used by Microsoft support or which can be assigned as context for services and applications to run in until the VM is deployed and more appropriate users can be used.
- For the case of using a Cloud-Only deployment scenario (see chapter [Cloud-Only - Virtual Machine deployments into Azure without dependencies on the on-premises customer network](#) of this document) in combination with this deployment method, domain accounts might not work once the Azure Disk is deployed in Azure. This is especially true for accounts which are used to run services like the DBMS or SAP applications. Therefore you need to replace such domain accounts with VM local accounts and delete the on-premises domain accounts in the VM. Keeping on-premises domain users in the VM image is not an issue when the VM

is deployed in the Cross-Premises scenario as described in chapter [Cross-Premise - Deployment of single or multiple SAP VMs into Azure with the requirement of being fully integrated into the on-premises network](#) in this document.

- If domain accounts were used as DBMS logins or users when running the system on-premises and those VMs are supposed to be deployed in Cloud-Only scenarios, the domain users need to be deleted. You need to make sure that the local administrator plus another VM local user is added as a login/user into the DBMS as administrators.
- Add other local accounts as those might be needed for the specific deployment scenario.

Windows

In this scenario no generalization (sysprep) of the VM is required to upload and deploy the VM on Azure. Make sure that drive D:\ is not used Set disk automount for attached disks as described in chapter [Setting automount for attached disks](#) in this document.

Linux

In this scenario no generalization (waagent -deprovision) of the VM is required to upload and deploy the VM on Azure. Make sure that /mnt/resource is not used and that ALL disks are mounted via uuid. For the OS disk make sure that the bootloader entry also reflects the uuid-based mount.

Preparation for deploying a VM with a customer specific image for SAP

VHD files that contain a generalized OS are also stored in containers on Azure Storage Accounts. You can deploy a new VM from such an image VHD by referencing the VHD as a source VHD in your deployment template files as described in chapter [Scenario 2: Deploying a VM with a custom image for SAP](#) of the [Deployment Guide](#).

Requirements when preparing your own Azure VM Image are:

- Originally the VHD containing the operating system could have a maximum size of 127GB only. This limitation got eliminated at the end of March 2015. Now the VHD containing the operating system can be up to 1TB in size as any other Azure Storage hosted VHD as well.
- It needs to be in the fixed VHD format. Dynamic VHDs or VHDs in VHDx format are not yet supported on Azure. Dynamic VHDs will be converted to static VHDs when you upload the VHD with PowerShell commandlets or CLI
- VHDs which are mounted to the VM and should be mounted again in Azure to the VM need to be in a fixed VHD format as well. The same size limit of the OS disk applies to data disks as well. VHDs can have a maximum size of 1TB. Dynamic VHDs will be converted to static VHDs when you upload the VHD with PowerShell commandlets or CLI
- Since all the Domain users registered as users in the VM will not exist in a Cloud-Only scenario (see chapter [Cloud-Only - Virtual Machine deployments into Azure without dependencies on the on-premises customer network](#) of this document), services using such domain accounts might not work once the Image is deployed in Azure. This is especially true for accounts which are used to run services like DBMS or SAP applications. Therefore you need to replace such domain accounts with VM local accounts and delete the on-premises domain accounts in the VM. Keeping on-premises domain users in the VM image might not be an issue when the VM is deployed in the Cross-Premise scenario as described in chapter [Cross-Premise - Deployment of single or multiple SAP VMs into Azure with the requirement of being fully integrated into the on-premises network](#) in this document.
- Add another local account with administrator privileges which can be used by Microsoft support in problem investigations or which can be assigned as context for services and applications to run in until the VM is deployed and more appropriate users can be used.
- In Cloud-Only deployments and where domain accounts were used as DBMS logins or users when running the system on-premises, the domain users should be deleted. You need to make sure that the local

administrator plus another VM local user is added as a login/user of the DBMS as administrators.

- Add other local accounts as those might be needed for the specific deployment scenario.
- If the image contains an installation of SAP NetWeaver and renaming of the host name from the original name at the point of the Azure deployment is likely, it is recommended to copy the latest versions of the SAP Software Provisioning Manager DVD into the template. This will enable you to easily use the SAP provided rename functionality to adapt the changed hostname and/or change the SID of the SAP system within the deployed VM image as soon as a new copy is started.

Windows

Make sure that drive D:\ is not used Set disk automount for attached disks as described in chapter [Setting automount for attached disks](#) in this document.

Linux

Make sure that /mnt/resource is not used and that ALL disks are mounted via uid. For the OS disk make sure the bootloader entry also reflects the uid-based mount.

- SAP GUI (for administrative and setup purposes) can be pre-installed in such a template.
- Other software necessary to run the VMs successfully in Cross-Premises scenarios can be installed as long as this software can work with the rename of the VM.

If the VM is prepared sufficiently to be generic and eventually independent of accounts/users not available in the targeted Azure deployment scenario, the last preparation step of generalizing such an image is conducted.

Generalizing a VM

Windows

The last step is to log in to a VM with an Administrator account. Open a Windows command window as 'administrator'. Go to ...\\windows\\system32\\sysprep and execute sysprep.exe. A small window will appear. It is important to check the 'Generalize' option (the default is un-checked) and change the Shutdown Option from its default of 'Reboot' to 'Shutdown'. This procedure assumes that the sysprep process is executed on-premises in the Guest OS of a VM. If you want to perform the procedure with a VM already running in Azure, follow the steps described in [this article](#).

Linux

[How to capture a Linux virtual machine to use as a Resource Manager template](#)

Transferring VMs and VHDs between on-premises to Azure

Since uploading VM images and disks to Azure is not possible via the Azure Portal, you need to use Azure PowerShell cmdlets or CLI. Another possibility is the use of the tool 'AzCopy'. The tool can copy VHDs between on-premises and Azure (in both directions). It also can copy VHDs between Azure Regions. Please consult [this documentation](#) for download and usage of AzCopy.

A third alternative would be to use various third party GUI oriented tools. However, please make sure that these tools are supporting Azure Page Blobs. For our purposes we need to use Azure Page Blob store (the differences are described here: <https://msdn.microsoft.com/library/windowsazure/ee691964.aspx>). Also the tools provided by Azure are very efficient in compressing the VMs and VHDs which need to be uploaded. This is important because this efficiency in compression reduces the upload time (which varies anyway depending on the upload link to the internet from the on-premises facility and the Azure deployment region targeted). It is a fair assumption that uploading a VM or VHD from European location to the U.S. based Azure data centers will take

longer than uploading the same VMs/VHDs to the European Azure data centers.

Uploading a VHD from on-premises to Azure

To upload an existing VM or VHD from the on-premises network such a VM or VHD needs to meet the requirements as listed in chapter [Preparation for moving a VM from on-premises to Azure with a non-generalized disk](#) of this document.

Such a VM does NOT need to be generalized and can be uploaded in the state and shape it has after shutdown on the on-premises side. The same is true for additional VHDs which don't contain any operating system.

Uploading a VHD and making it an Azure Disk

In this case we want to upload a VHD, either with or without an OS in it, and mount it to a VM as a data disk or use it as OS disk. This is a multi-step process

Powershell

- Login to your subscription with *Login-AzureRmAccount*
- Set the subscription of your context with *Set-AzureRmContext* and parameter SubscriptionId or SubscriptionName - see <https://msdn.microsoft.com/library/mt619263.aspx>
- Upload the VHD with *Add-AzureRmVhd* to an Azure Storage Account - see <https://msdn.microsoft.com/library/mt603554.aspx>
- Set the OS disk of a new VM config to the VHD with *Set-AzureRmVMOSDisk* - see <https://msdn.microsoft.com/library/mt603746.aspx>
- Create a new VM from the VM config with *New-AzureRmVM* - see <https://msdn.microsoft.com/library/mt603754.aspx>
- Add a data disk to a new VM with *Add-AzureRmVMDataDisk* - see <https://msdn.microsoft.com/library/mt603673.aspx>

Azure CLI

- Switch to Azure Resource Manager mode with *azure config mode arm*
- Login to your subscription with *azure login*
- Select your subscription with *azure account set <subscription name or id>*
- Upload the VHD with *azure storage blob upload* - see [Using the Azure CLI with Azure Storage](#)
- Create a new VM specifying the uploaded VHD as OS disk with *azure vm create* and parameter -d
- Add a data disk to a new VM with *vm disk attach-new*

Template

- Upload the VHD with Powershell or Azure CLI
- Deploy the VM with a JSON template referencing the VHD as shown in [this example JSON template](#).

Deployment of a VM Image

To upload an existing VM or VHD from the on-premises network in order to use it as an Azure VM image such a VM or VHD need to meet the requirements listed in chapter [Preparation for deploying a VM with a customer specific image for SAP](#) of this document.

- Use *sysprep* on Windows or *waagent -deprovision* on Linux to generalize your VM - see [How to capture a Windows virtual machine in the Resource Manager deployment model](#) or [How to capture a Linux virtual machine to use as a Resource Manager template](#)
- Login to your subscription with *Login-AzureRmAccount*
- Set the subscription of your context with *Set-AzureRmContext* and parameter SubscriptionId or SubscriptionName - see <https://msdn.microsoft.com/library/mt619263.aspx>
- Upload the VHD with *Add-AzureRmVhd* to an Azure Storage Account - see <https://msdn.microsoft.com/library/mt603554.aspx>

- Set the OS disk of a new VM config to the VHD with `Set-AzureRmVMOSDisk -SourceImageUri -CreateOption fromImage` - see <https://msdn.microsoft.com/library/mt603746.aspx>
- Create a new VM from the VM config with `New-AzureRmVM` - see <https://msdn.microsoft.com/library/mt603754.aspx>

Azure CLI

- Use `sysprep` on Windows or `waagent -deprovision` on Linux to generalize your VM - see [How to capture a Windows virtual machine in the Resource Manager deployment model](#) or [How to capture a Linux virtual machine to use as a Resource Manager template](#) for Linux
- Switch to Azure Resource Manager mode with `azure config mode arm`
- Login to your subscription with `azure login`
- Select your subscription with `azure account set <subscription name or id>`
- Upload the VHD with `azure storage blob upload` - see [Using the Azure CLI with Azure Storage](#)
- Create a new VM specifying the uploaded VHD as OS disk with `azure vm create` and parameter -Q

Template

- Use `sysprep` on Windows or `waagent -deprovision` on Linux to generalize your VM - see [How to capture a Windows virtual machine in the Resource Manager deployment model](#) or [How to capture a Linux virtual machine to use as a Resource Manager template](#) for Linux
- Upload the VHD with Powershell or Azure CLI
- Deploy the VM with a JSON template referencing the image VHD as shown in [this example JSON template](#).

Downloading VHDS to on-premises

Azure Infrastructure as a Service is not a one-way street of only being able to upload VHDS and SAP systems. You can move SAP systems from Azure back into the on-premises world as well.

During the time of the download the VHDS can't be active. Even when downloading VHDS which are mounted to VMs, the VM needs to be shutdown. If you only want to download the database content which then should be used to set up a new system on-premises and if it is acceptable that during the time of the download and the setup of the new system that the system in Azure can still be operational, you could avoid a long downtime by performing a compressed database backup into a VHD and just download that VHD instead of also downloading the OS base VM.

Powershell

Once the SAP system is stopped and the VM is shutdown, you can use the PowerShell cmdlet `Save-AzureRmVhd` on the on-premises target to download the VHD disks back to the on-premises world. In order to do that, you need the URL of the VHD which you can find in the 'Storage Section' of the Azure Portal (need to navigate to the Storage Account and the storage container where the VHD was created) and you need to know where the VHD should be copied to.

Then you can leverage the command by simply defining the parameter `SourceUri` as the URL of the VHD to download and the `LocalFilePath` as the physical location of the VHD (including its name). The command could look like:

```
Save-AzureRmVhd -ResourceGroupName <resource group name of storage account> -SourceUri http://<storage account name>.blob.core.windows.net/<container name>/sapidesdata.vhd -LocalFilePath E:\Azure_downloads\sapidesdata.vhd
```

For more details of the `Save-AzureRmVhd` cmdlet, please check here

<https://msdn.microsoft.com/library/mt622705.aspx>.

CLI

Once the SAP system is stopped and the VM is shutdown, you can use the Azure CLI command `azure storage blob download` on the on-premises target to download the VHD disks back to the on-premises world. In order to

do that, you need the name and the container of the VHD which you can find in the 'Storage Section' of the Azure Portal (need to navigate to the Storage Account and the storage container where the VHD was created) and you need to know where the VHD should be copied to.

Then you can leverage the command by simply defining the parameters blob and container of the VHD to download and the destination as the physical target location of the VHD (including its name). The command could look like:

```
azure storage blob download --blob <name of the VHD to download> --container <container of the VHD to download> --account-name <storage account name of the VHD to download> --account-key <storage account key> --destination <destination of the VHD to download>
```

Transferring VMs and VHDs within Azure

Copying SAP systems within Azure

A SAP system or even a dedicated DBMS server supporting a SAP application layer will likely consist of several VHDs which contain either the OS with the binaries or the data and log file(s) of the SAP database. Neither the Azure functionality of copying VHDs nor the Azure functionality of saving VHDs to disk has a synchronization mechanism which would snapshot multiple VHDs synchronously. Therefore, the state of the copied or saved VHDs even if those are mounted against the same VM would be different. This means that in the concrete case of having different data and logfile(s) contained in the different VHDs, the database in the end would be inconsistent.

Conclusion: In order to copy or save VHDs which are part of an SAP system configuration you need to stop the SAP system and also need to shut down the deployed VM. Only then can you copy or download the set of VHDs to either create a copy of the SAP system in Azure or on-premises.

Data disks are stored as VHD files in an Azure Storage Account and can be directly attach to a virtual machine or be used as an image. In this case, the VHD is copied to another location before being attached to the virtual machine. The full name of the VHD file in Azure must be unique within Azure. As mentioned earlier already, the name is kind of a three-part name that looks like:

```
http(s)://<storage account name>.blob.core.windows.net/<container name>/<vhd name>
```

Powershell

You can use Azure PowerShell cmdlets to copy a VHD as shown in [this article](#).

CLI

You can use Azure CLI to copy a VHD as shown in [this article](#)

Azure Storage tools

- <http://azurerestorageexplorer.codeplex.com/releases/view/125870>

There also are professional editions of Azure Storage Explorers which can be found here:

- <http://www.cerebrata.com/>
- <http://clumsyleaf.com/products/cloudxplorer>

The copy of a VHD itself within a storage account is a process which takes only a few seconds (similar to SAN hardware creating snapshots with lazy copy and copy on write). After you have a copy of the VHD file you can attach it to a virtual machine or use it as an image to attach copies of the VHD to virtual machines.

Powershell

```

# attach a vhd to a vm
$vm = Get-AzureRmVM -ResourceGroupName <resource group name> -Name <vm name>
$vm = Add-AzureRmVMDataDisk -VM $vm -Name newdatadisk -VhdUri <path to vhd> -Caching <caching option> -
DiskSizeInGB $null -Lun <lun e.g. 0> -CreateOption attach
$vm | Update-AzureRmVM

# attach a copy of the vhd to a vm
$vm = Get-AzureRmVM -ResourceGroupName <resource group name> -Name <vm name>
$vm = Add-AzureRmVMDataDisk -VM $vm -Name newdatadisk -VhdUri <new path of vhd> -SourceImageUri <path to image
vhd> -Caching <caching option> -DiskSizeInGB $null -Lun <lun e.g. 0> -CreateOption fromImage
$vm | Update-AzureRmVM

```

CLI

```

azure config mode arm

# attach a vhd to a vm
azure vm disk attach <resource group name> <vm name> <path to vhd>

# attach a copy of the vhd to a vm
# this scenario is currently not possible with Azure CLI. A workaround is to manually copy the vhd to the
destination.

```

Copying disks between Azure Storage Accounts

This task cannot be performed on the Azure Portal. You can use Azure PowerShell cmdlets, Azure CLI or a third party storage browser. The PowerShell cmdlets or CLI commands can create and manage blobs, which include the ability to asynchronously copy blobs across Storage Accounts and across regions within the Azure subscription.

Powershell

Copying VHDS between subscriptions is also possible. For more information read [this article](#).

The basic flow of the PS cmdlet logic looks like this:

- Create a storage account context for the source storage account with *New-AzureStorageContext* - see <https://msdn.microsoft.com/library/dn806380.aspx>
- Create a storage account context for the target storage account with *New-AzureStorageContext* - see <https://msdn.microsoft.com/library/dn806380.aspx>
- Start the copy with

```

Start-AzureStorageBlobCopy -SrcBlob <source blob name> -SrcContainer <source container name> -SrcContext
<variable containing context of source storage account> -DestBlob <target blob name> -DestContainer <target
container name> -DestContext <variable containing context of target storage account>

```

- Check the status of the copy in a loop with

```

Get-AzureStorageBlobCopyState -Blob <target blob name> -Container <target container name> -Context <variable
containing context of target storage account>

```

- Attach the new VHD to a virtual machine as described above.

For examples see [this article](#)

CLI

- Start the copy with

```
azure storage blob copy start --source-blob <source blob name> --source-container <source container name> --account-name <source storage account name> --account-key <source storage account key> --dest-container <target container name> --dest-blob <target blob name> --dest-account-name <target storage account name> --dest-account-key <target storage account name>
```

- Check the status if the copy in a loop with

```
azure storage blob copy show --blob <target blob name> --container <target container name> --account-name <target storage account name> --account-key <target storage account name>
```

- Attach the new VHD to a virtual machine as described above.

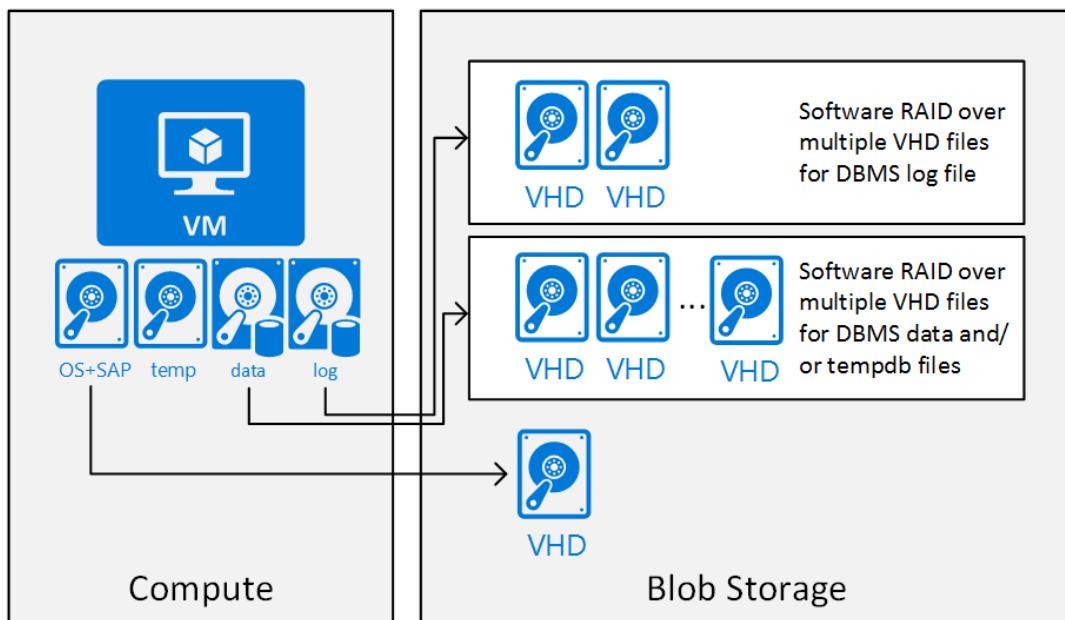
For examples see [this article](#)

Disk Handling

VM/VHD structure for SAP deployments

Ideally the handling of the structure of a VM and the associated VHDs should be very simple. In on-premises installations, customers developed many ways of structuring a server installation.

- One base VHD which contains the OS and all the binaries of the DBMS and/or SAP. Since March 2015, this VHD can be up to 1TB in size instead of earlier restrictions that limited it to 127GB.
- One or multiple VHDs which contains the DBMS log file of the SAP database and the log file of the DBMS temp storage area (if the DBMS supports this). If the database log IOPS requirements are high, you need to stripe multiple VHDs in order to reach the IOPS volume required.
- A number of VHDs containing one or two database files of the SAP database and the DBMS temp data files as well (if the DBMS supports this).



With many customers we saw configurations where, for example, SAP and DBMS binaries were not installed on the c:\ drive where the OS was installed. There were various reasons for this, but when we went back to the root, it usually was that the drives were small and OS upgrades needed additional space 10-15 years ago. Both conditions do not apply these days too often anymore. Today the c:\ drive can be mapped on large

volume disks or VMs. In order to keep deployments simple in their structure, it is recommended to follow the following deployment pattern for SAP NetWeaver systems in Azure

The Windows operating system pagefile should be on the D: drive (non-persistent disk)



Place the Linux swapfile under /mnt/resource on Linux as described in [this article](#). The swap file can be configured in the configuration file of the Linux Agent /etc/waagent.conf. Add or change the following settings:

```
ResourceDisk.EnableSwap=y  
ResourceDisk.SwapSizeMB=30720
```

To activate the changes, you need to restart the Linux Agent with

```
sudo service waagent restart
```

Please read SAP Note [1597355](#) for more details on the recommended swap file size

The number of VHDs used for the DBMS data files and the type of Azure Storage these VHDs are hosted on should be determined by the IOPS requirements and the latency required. Exact quotas are described in [this article](#)

Experience of SAP deployments over the last 2 years taught us some lessons which can be summarized as:

- IOPS traffic to different data files is not always the same since existing customer systems might have differently sized data files representing their SAP database(s). As a result it turned out to be better using a RAID configuration over multiple VHDs to place the data files LUNs carved out of those. There were situations, especially with Azure Standard Storage where an IOPS rate hit the quota of a single VHD against the DBMS transaction log. In such scenarios the use of Premium Storage is recommended or alternatively aggregating multiple Standard Storage VHDs with a software RAID.



- [Performance best practices for SQL Server in Azure Virtual Machines](#)



- [Configure Software RAID on Linux](#)
- [Configure LVM on a Linux VM in Azure](#)
- [Azure Storage secrets and Linux I/O optimizations](#)

- Premium Storage is showing significant better performance, especially for critical transaction log writes. For SAP scenarios that are expected to deliver production like performance, it is highly recommended to use VM-Series that can leverage Azure Premium Storage.

Keep in mind that the VHD which contains the OS, and as we recommend, the binaries of SAP and the database (base VM) as well, is not anymore limited to 127GB. It now can have up to 1TB in size. This should be enough space to keep all the necessary file including e.g. SAP batch job logs.

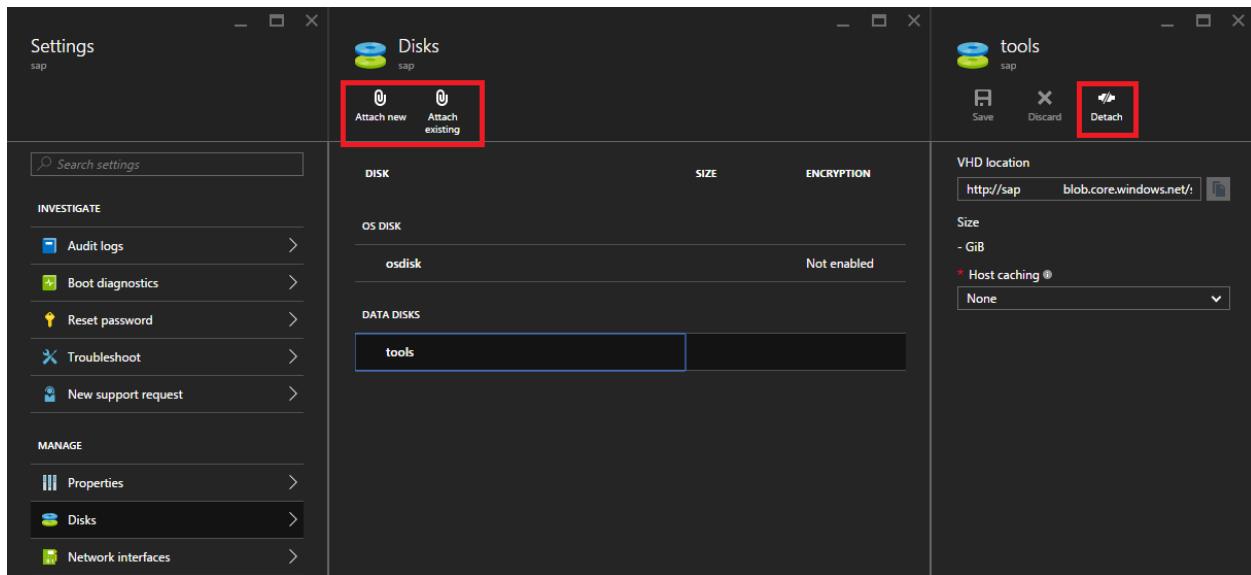
For more suggestions and more details, specifically for DBMS VMs, please consult the [DBMS Deployment Guide](#)

Disk Handling

In most scenarios you need to create additional disks in order to deploy the SAP database into the VM. We talked

about the considerations on number of VHDs in chapter [VM/VHD structure for SAP deployments](#) of this document. The Azure Portal allows to attach and detach disks once a base VM is deployed. The disks can be attached/detached when the VM is up and running as well as when it is stopped. When attaching a disk, the Azure Portal offers to attach an empty disk or an existing disk which at this point in time is not attached to another VM.

Note: VHDs can only be attached to one VM at any given time.



You need to decide whether you want to create a new and empty VHD (which would be created in the same Storage Account as the base VM is in) or whether you want to select an existing VHD that was uploaded earlier and should be attached to the VM now.

IMPORTANT: You **DO NOT** want to use Host Caching with Azure Standard Storage. You should leave the Host Cache preference at the default of NONE. With Azure Premium Storage you should enable Read Caching if the I/O characteristic is mostly read like typical I/O traffic against database data files. In case of database transaction log file no caching is recommended.

Windows

[How to attach a data disk in the Azure portal](#)

If disks are attached, you need to log in into the VM to open the Windows Disk Manager. If automount is not enabled as recommended in chapter [Setting automount for attached disks](#), the newly attached volume needs to be taken online and initialized.

Linux

If disks are attached, you need to log in into the VM and initialize the disks as described in [this article](#)

If the new disk is an empty disk, you need to format the disk as well. For formatting, especially for DBMS data and log files the same recommendations as for bare-metal deployments of the DBMS apply.

As already mentioned in chapter [The Microsoft Azure Virtual Machine Concept](#), an Azure Storage account does not provide infinite resources in terms of I/O volume, IOPS and data volume. Usually DBMS VMs are most affected by this. It might be best to use a separate Storage Account for each VM if you have few high I/O volume VMs to deploy in order to stay within the limit of the Azure Storage Account volume. Otherwise, you need to see how you can balance these VMs between different Storage accounts without hitting the limit of each single Storage Account. More details are discussed in the [DBMS Deployment Guide](#). You should also keep these limitations in mind for pure SAP application server VMs or other VMs which eventually might require additional

VHDs.

Another topic which is relevant for Storage Accounts is whether the VHDs in a Storage Account are getting Geo-replicated. Geo-replication is enabled or disabled on the Storage Account level and not on the VM level. If geo-replication is enabled, the VHDs within the Storage Account would be replicated into another Azure data center within the same region. Before deciding on this, you should think about the following restriction:

Azure Geo-replication works locally on each VHD in a VM and does not replicate the IOs in chronological order across multiple VHDs in a VM. Therefore, the VHD that represents the base VM as well as any additional VHDs attached to the VM are replicated independent of each other. This means there is no synchronization between the changes in the different VHDs. The fact that the IOs are replicated independently of the order in which they are written means that geo-replication is not of value for database servers that have their databases distributed over multiple VHDs. In addition to the DBMS, there also might be other applications where processes write or manipulate data in different VHDs and where it is important to keep the order of changes. If that is a requirement, geo-replication in Azure should not be enabled. Dependent on whether you need or want geo-replication for a set of VMs, but not for another set, you can already categorize VMs and their related VHDs into different Storage Accounts that have geo-replication enabled or disabled.

Setting automount for attached disks



Windows

For VMs which are created from own Images or Disks, it is necessary to check and possibly set the automount parameter. Setting this parameter will allow the VM after a restart or redeployment in Azure to mount the attached/mounted drives again automatically. The parameter is set for the images provided by Microsoft in the Azure Marketplace.

In order to set the automount, please check the documentation of the command line executable diskpart.exe here:

- [DiskPart Command-Line Options](#)
- [Automount](#)

The Windows command line window should be opened as administrator.

If disks are attached, you need to log in into the VM to open the Windows Disk Manager. If automount is not enabled as recommended in chapter [Setting automount for attached disks](#), the newly attached volume > needs to be taken online and initialized.



You need to initialize an newly attached empty disk as described in [this article](#). You also need to add new disks to the /etc/fstab.

Final Deployment

For the final Deployment and exact steps, especially with regards to the deployment of SAP Extended Monitoring, please refer to the [Deployment Guide](#).

Accessing SAP systems running within Azure VMs

For Cloud-Only scenarios, you might want to connect to those SAP systems across the public internet using SAP GUI. For these cases, the following procedures need to be applied.

Later in the document we will discuss the other major scenario, connecting to SAP systems in Cross-Premises deployments which have a site-to-site connection (VPN tunnel) or Azure ExpressRoute connection between the on-premises systems and Azure systems.

Remote Access to SAP systems

With Azure Resource Manager there are no default endpoints anymore like in the former classic model. All ports of an Azure ARM VM are open as long as:

1. No Network Security Group is defined for the subnet or the network interface. Network traffic to Azure VMs can be secured via so-called "Network Security Groups". For more information see [What is a Network Security Group \(NSG\)?](#)
2. No Azure Load Balancer is defined for the network interface

See the architecture difference between classic model and ARM as described in [this article](#).

Configuration of the SAP System and SAP GUI connectivity for Cloud-Only scenario

Please see this article which describes details to this topic:

<http://blogs.msdn.com/b/saponsqlserver/archive/2014/06/24/sap-gui-connection-closed-when-connecting-to-sap-system-in-azure.aspx>

Changing Firewall Settings within VM

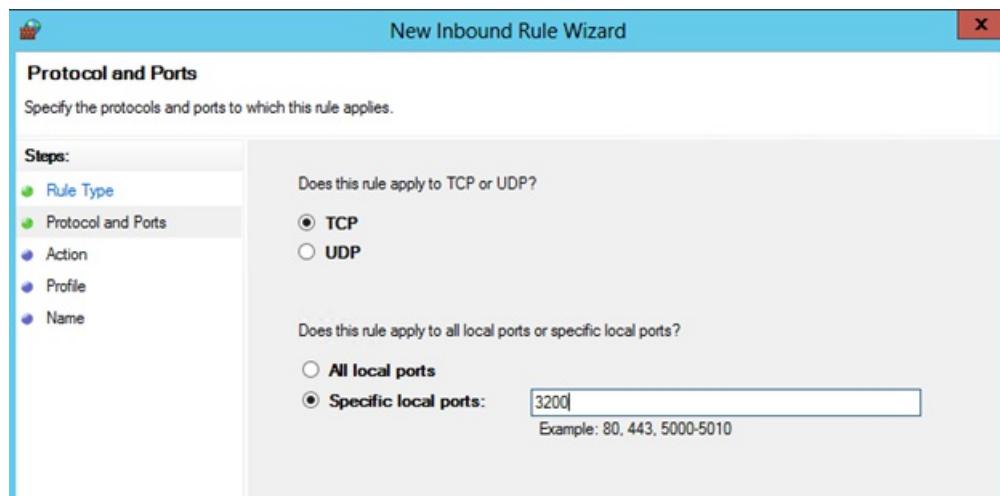
It might be necessary to configure the firewall on your virtual machines to allow inbound traffic to your SAP system.



By default, the Windows Firewall within an Azure deployed VM is turned on. You now need to allow the SAP Port to be opened, otherwise the SAP GUI will not be able to connect. To do this:

- Open Control Panel\System and Security\Windows Firewall to 'Advanced Settings'.
- Now right-click on Inbound Rules and chose 'New Rule'.
- In the following Wizard chose to create a new 'Port' rule.
- In the next step of the wizard, leave the setting at TCP and type in the port number you want to open. Since our SAP instance ID is 00, we took 3200. If your instance has a different instance number, the port you defined earlier based on the instance number should be opened.
- In the next part of the wizard, you need to leave the item 'Allow Connection' checked.
- In the next step of the wizard you need to define whether the rule applies for Domain, Private and Public network. Please adjust it if necessary to your needs. However, connecting with SAP GUI from the outside through the public network, you need to have the rule applied to the public network.
- In the last step of the wizard, you need to give the rule a name and then save the rule by pressing 'Finish'

The rule becomes effective immediately.



The Linux images in the Azure Marketplace do not enable the iptables firewall by default and the connection to your SAP system should work. If you enabled iptables or another firewall, please refer to the documentation of iptables or the used firewall to allow inbound tcp traffic to port 32xx (where xx is the system number of your SAP system).

Security recommendations

The SAP GUI does not connect immediately to any of the SAP instances (port 32xx) which are running, but first connects via the port opened to the SAP message server process (port 36xx). In the past the very same port was used by the message server for the internal communication to the application instances. To prevent on-premises application servers from inadvertently communicating with a message server in Azure the internal communication ports can be changed. It is highly recommended to change the internal communication between the SAP message server and its application instances to a different port number on systems that have been cloned from on-premises systems, such as a clone of development for project testing etc. This can be done with the default profile parameter:

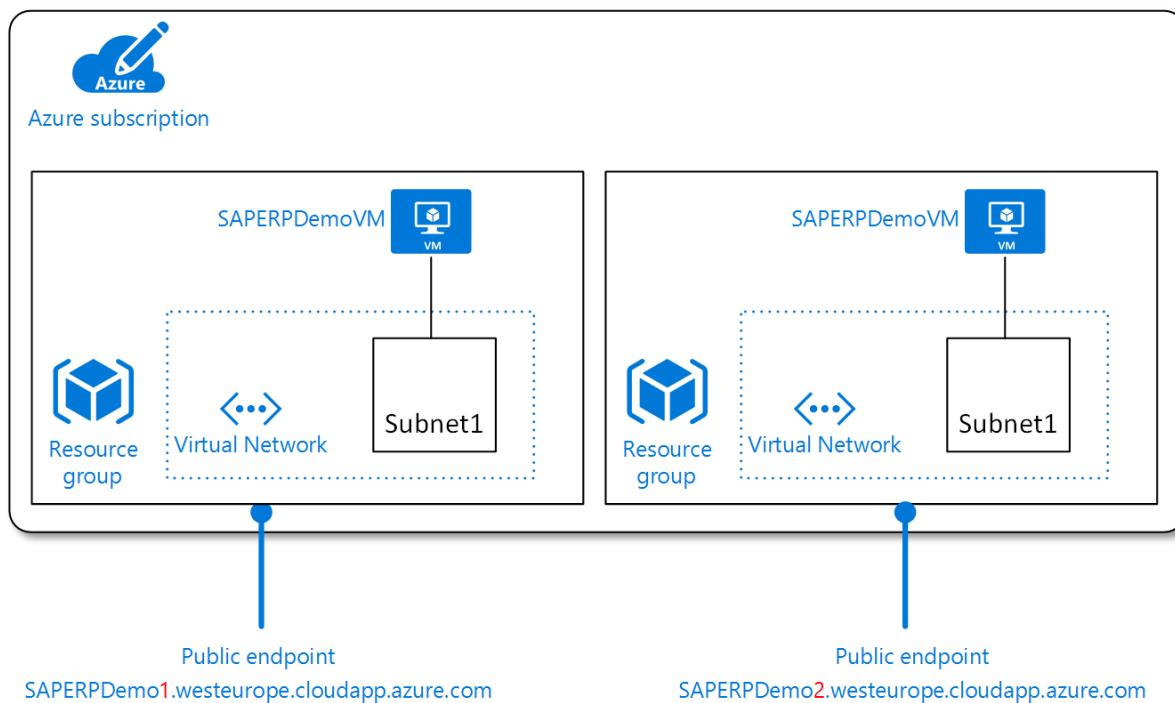
```
rdisp/msserv_internal
```

as documented in:

https://help.sap.com/saphelp_nwpi71/helpdata/en/47/c56a6938fb2d65e10000000a42189c/content.htm

Concepts of Cloud-Only deployment of SAP instances

Single VM with SAP NetWeaver demo/training scenario



In this scenario (see chapter [Cloud-Only](#) of this document) we are implementing a typical training/demo system scenario where the complete training/demo scenario is contained within a single VM. We assume that the deployment is done through VM image templates. We also assume that multiple of these demo/trainings VMs need to be deployed with the VMs having the same name.

The assumption is that you created a VM Image as described in some sections of chapter [Preparing VMs with SAP for Azure](#) in this document.

The sequence of events to implement the scenario looks like this:

Powershell

- Create a new resource group for every training/demo landscape

```
$rgName = "SAPERPDemo1"  
New-AzureRmResourceGroup -Name $rgName -Location "North Europe"
```

- Create a new storage account

```
$suffix = Get-Random -Minimum 100000 -Maximum 999999  
$account = New-AzureRmStorageAccount -ResourceGroupName $rgName -Name "saperpdemo$suffix" -SkuName  
Standard_LRS -Kind "Storage" -Location "North Europe"
```

- Create a new virtual network for every training/demo landscape to enable the usage of the same hostname and IP addresses. The virtual network is protected by a Network Security Group that only allows traffic to port 3389 to enable Remote Desktop access and port 22 for SSH.

```
# Create a new Virtual Network  
$rdpRule = New-AzureRmNetworkSecurityRuleConfig -Name SAPERPDemoNSGRDP -Protocol * -SourcePortRange * -  
DestinationPortRange 3389 -Access Allow -Direction Inbound -SourceAddressPrefix * -DestinationAddressPrefix * -  
Priority 100  
$sshRule = New-AzureRmNetworkSecurityRuleConfig -Name SAPERPDemoNSGSSH -Protocol * -SourcePortRange * -  
DestinationPortRange 22 -Access Allow -Direction Inbound -SourceAddressPrefix * -DestinationAddressPrefix * -  
Priority 101  
$nsg = New-AzureRmNetworkSecurityGroup -Name SAPERPDemoNSG -ResourceGroupName $rgName -Location "North  
Europe" -SecurityRules $rdpRule,$sshRule  
  
$subnetConfig = New-AzureRmVirtualNetworkSubnetConfig -Name Subnet1 -AddressPrefix 10.0.1.0/24 -  
NetworkSecurityGroup $nsg  
$vnet = New-AzureRmVirtualNetwork -Name SAPERPDemoVNet -ResourceGroupName $rgName -Location "North Europe" -  
AddressPrefix 10.0.1.0/24 -Subnet $subnetConfig
```

- Create a new public IP address that can be used to access the virtual machine from the internet

```
# Create a public IP address with a DNS name  
$pip = New-AzureRmPublicIpAddress -Name SAPERPDemoPIP -ResourceGroupName $rgName -Location "North Europe" -  
DomainNameLabel $rgName.ToLower() -AllocationMethod Dynamic
```

- Create a new network interface for the virtual machine

```
# Create a new Network Interface  
$nic = New-AzureRmNetworkInterface -Name SAPERPDemoNIC -ResourceGroupName $rgName -Location "North Europe" -  
Subnet $vnet.Subnets[0] -PublicIpAddress $pip
```

- Create a virtual machine. For the Cloud-Only scenario every VM will have the same name. The SAP SID of the SAP NetWeaver instances in those VMs will be the same as well. Within the Azure Resource Group, the name of the VM needs to be unique, but in different Azure Resource Groups you can run VMs with the same name. The default 'Administrator' account of Windows or 'root' for Linux are not valid. Therefore, a new administrator user name needs to be defined together with a password. The size of the VM also needs to be defined.

```

#####
# Create a new virtual machine with an official image from the Azure Marketplace
#####
$cred=Get-Credential -Message "Type the name and password of the local administrator account."
$vmconfig = New-AzureRmVMConfig -VMName SAPERPDemo -VMSize Standard_D11

# select image
$vmconfig = Set-AzureRmVMSourceImage -VM $vmconfig -PublisherName "MicrosoftWindowsServer" -Offer
"WindowsServer" -Skus "2012-R2-Datacenter" -Version "latest"
$vmconfig = Set-AzureRmVMOperatingSystem -VM $vmconfig -Windows -ComputerName "SAPERPDemo" -Credential $cred -
ProvisionVMAgent -EnableAutoUpdate
# $vmconfig = Set-AzureRmVMSourceImage -VM $vmconfig -PublisherName "SUSE" -Offer "SLES" -Skus "12" -Version
"latest"
# $vmconfig = Set-AzureRmVMSourceImage -VM $vmconfig -PublisherName "RedHat" -Offer "RHEL" -Skus "7.2" -
Version "latest"
# $vmconfig = Set-AzureRmVMOperatingSystem -VM $vmconfig -Linux -ComputerName "SAPERPDemo" -Credential $cred

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

$diskName="os"
$osDiskUri=$account.PrimaryEndpoints.Blob.ToString() + "vhds/" + $diskName + ".vhd"
$vmconfig = Set-AzureRmVMOSDisk -VM $vmconfig -Name $diskName -VhdUri $osDiskUri -CreateOption fromImage
$vm = New-AzureRmVM -ResourceGroupName $rgName -Location "North Europe" -VM $vmconfig

```

```

#####
# Create a new virtual machine with a VHD that contains the private image that you want to use
#####
$cred=Get-Credential -Message "Type the name and password of the local administrator account."
$vmconfig = New-AzureRmVMConfig -VMName SAPERPDemo -VMSize Standard_D11

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

$diskName="osfromimage"
$osDiskUri=$account.PrimaryEndpoints.Blob.ToString() + "vhds/" + $diskName + ".vhd"

$vmconfig = Set-AzureRmVMOSDisk -VM $vmconfig -Name $diskName -VhdUri $osDiskUri -CreateOption fromImage -
SourceImageUri <path to VHD that contains the OS image> -Windows
$vmconfig = Set-AzureRmVMOperatingSystem -VM $vmconfig -Windows -ComputerName "SAPERPDemo" -Credential $cred
#$vmconfig = Set-AzureRmVMOSDisk -VM $vmconfig -Name $diskName -VhdUri $osDiskUri -CreateOption fromImage -
SourceImageUri <path to VHD that contains the OS image> -Linux
#$vmconfig = Set-AzureRmVMOperatingSystem -VM $vmconfig -Linux -ComputerName "SAPERPDemo" -Credential $cred

$vm = New-AzureRmVM -ResourceGroupName $rgName -Location "North Europe" -VM $vmconfig

```

- Optionally add additional disks and restore necessary content. Be aware that all blob names (URLs to the blobs) must be unique within Azure.

```

# Optional: Attach additional data disks
$vm = Get-AzureRmVM -ResourceGroupName $rgName -Name SAPERPDemo
$dataDiskUri = $account.PrimaryEndpoints.Blob.ToString() + "vhds/dataldisk.vhd"
Add-AzureRmVMDataDisk -VM $vm -Name datadisk -VhdUri $dataDiskUri -DiskSizeInGB 1023 -CreateOption empty |
Update-AzureRmVM

```

CLI

The following example code can be used on Linux. For Windows, please either use PowerShell as described above or adapt the example to use %rgName% instead of \$rgName and set the environment variable using the Windows command `set`.

- Create a new resource group for every training/demo landscape

```
rgName=SAPERPDemo1
rgNameLower=saperpdemo1
azure group create $rgName "North Europe"
```

- Create a new storage account

```
azure storage account create --resource-group $rgName --location "North Europe" --kind Storage --sku-name LRS
$rgNameLower
```

- Create a new virtual network for every training/demo landscape to enable the usage of the same hostname and IP addresses. The virtual network is protected by a Network Security Group that only allows traffic to port 3389 to enable Remote Desktop access and port 22 for SSH.

```
azure network nsg create --resource-group $rgName --location "North Europe" --name SAPERPDemoNSG
azure network nsg rule create --resource-group $rgName --nsg-name SAPERPDemoNSG --name SAPERPDemoNSGRDP --
protocol \* --source-address-prefix \* --source-port-range \* --destination-address-prefix \* --destination-
port-range 3389 --access Allow --priority 100 --direction Inbound
azure network nsg rule create --resource-group $rgName --nsg-name SAPERPDemoNSG --name SAPERPDemoNSGSSH --
protocol \* --source-address-prefix \* --source-port-range \* --destination-address-prefix \* --destination-
port-range 22 --access Allow --priority 101 --direction Inbound

azure network vnet create --resource-group $rgName --name SAPERPDemoVNet --location "North Europe" --address-
prefixes 10.0.1.0/24
azure network vnet subnet create --resource-group $rgName --vnet-name SAPERPDemoVNet --name Subnet1 --address-
prefix 10.0.1.0/24 --network-security-group-name SAPERPDemoNSG
```

- Create a new public IP address that can be used to access the virtual machine from the internet

```
azure network public-ip create --resource-group $rgName --name SAPERPDemoPIP --location "North Europe" --
domain-name-label $rgNameLower --allocation-method Dynamic
```

- Create a new network interface for the virtual machine

```
azure network nic create --resource-group $rgName --location "North Europe" --name SAPERPDemoNIC --public-ip-
name SAPERPDemoPIP --subnet-name Subnet1 --subnet-vnet-name SAPERPDemoVNet
```

- Create a virtual machine. For the Cloud-Only scenario every VM will have the same name. The SAP SID of the SAP NetWeaver instances in those VMs will be the same as well. Within the Azure Resource Group, the name of the VM needs to be unique, but in different Azure Resource Groups you can run VMs with the same name. The default 'Administrator' account of Windows or 'root' for Linux are not valid. Therefore, a new administrator user name needs to be defined together with a password. The size of the VM also needs to be defined.

```
azure vm create --resource-group $rgName --location "North Europe" --name SAPERPDemo --nic-name SAPERPDemoNIC
--image-urn MicrosoftWindowsServer:WindowsServer:2012-R2-Datacenter:latest --os-type Windows --admin-username
<username> --admin-password <password> --vm-size Standard_D11 --os-disk-vhd
https://$rgNameLower.blob.core.windows.net/vhds/os.vhd --disable-boot-diagnostics
# azure vm create --resource-group $rgName --location "North Europe" --name SAPERPDemo --nic-name
SAPERPDemoNIC --image-urn SUSE:SLES:12:latest --os-type Linux --admin-username <username> --admin-password
<password> --vm-size Standard_D11 --os-disk-vhd https://$rgNameLower.blob.core.windows.net/vhds/os.vhd --
disable-boot-diagnostics
# azure vm create --resource-group $rgName --location "North Europe" --name SAPERPDemo --nic-name
SAPERPDemoNIC --image-urn RedHat:RHEL:7.2:latest --os-type Linux --admin-username <username> --admin-password
<password> --vm-size Standard_D11 --os-disk-vhd https://$rgNameLower.blob.core.windows.net/vhds/os.vhd --
disable-boot-diagnostics
```

```
#####
# Create a new virtual machine with a VHD that contains the private image that you want to use
#####
azure vm create --resource-group $rgName --location "North Europe" --name SAPERPDemo --nic-name SAPERPDemoNIC
--os-type Windows --admin-username <username> --admin-password <password> --vm-size Standard_D11 --os-disk-vhd
https://$rgNameLower.blob.core.windows.net/vhds/os.vhd -Q <path to image vhd> --disable-boot-diagnostics
#azure vm create --resource-group $rgName --location "North Europe" --name SAPERPDemo --nic-name SAPERPDemoNIC
--os-type Linux --admin-username <username> --admin-password <password> --vm-size Standard_D11 --os-disk-vhd
https://$rgNameLower.blob.core.windows.net/vhds/os.vhd -Q <path to image vhd> --disable-boot-diagnostics
```

- Optionally add additional disks and restore necessary content. Be aware that all blob names (URLs to the blobs) must be unique within Azure.

```
# Optional: Attach additional data disks
azure vm disk attach-new --resource-group $rgName --vm-name SAPERPDemo --size-in-gb 1023 --vhd-name datadisk
```

Template

You can use the sample templates on the [azure-quickstart-templates](#) repository on github.

- [Simple Linux VM](#)
- [Simple Windows VM](#)
- [VM from image](#)

Implement a set of VMs which need to communicate within Azure

This Cloud-Only scenario is a typical scenario for training and demo purposes where the software representing the demo/training scenario is spread over multiple VMs. The different components installed in the different VMs need to communicate with each other. Again, in this scenario no on-premises network communication or Cross-Premises scenario is needed.

This scenario is an extension of the installation described in chapter [Single VM with SAP NetWeaver demo/training scenario](#) of this document. In this case more virtual machines will be added to an existing resource group. In the following example the training landscape consists of an SAP ASCS/SCS VM, a VM running a DBMS and an SAP Application Server instance VM.

Before you build this scenario you need to think about basic settings as already exercised in the scenario before.

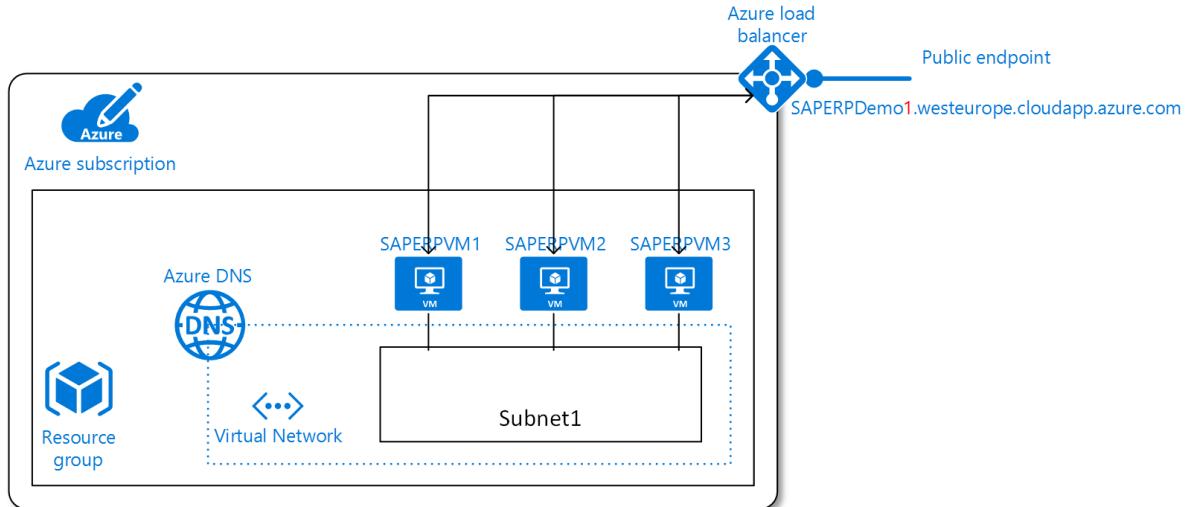
Resource Group and Virtual Machine naming

All resource group names must be unique. Develop your own naming scheme of your resources, such as

`<rg-name>-suffix.`

The virtual machine name has to be unique within the resource group.

Setup Network for communication between the different VMs



To prevent naming collisions with clones of the same training/demo landscapes, you need to create an Azure Virtual Network for every landscape. DNS name resolution will be provided by Azure or you can configure your own DNS server outside Azure (not to be further discussed here). In this scenario we do not configure our own DNS. For all virtual machines inside one Azure Virtual Network communication via hostnames will be enabled.

The reasons to separate training or demo landscapes by virtual networks and not only resource groups could be:

- The SAP landscape as set up needs its own AD/OpenLDAP and a Domain Server needs to be part of each of the landscapes.
- The SAP landscape as set up has components that need to work with fixed IP addresses.

More details about Azure Virtual Networks and how to define them can be found in [this article](#).

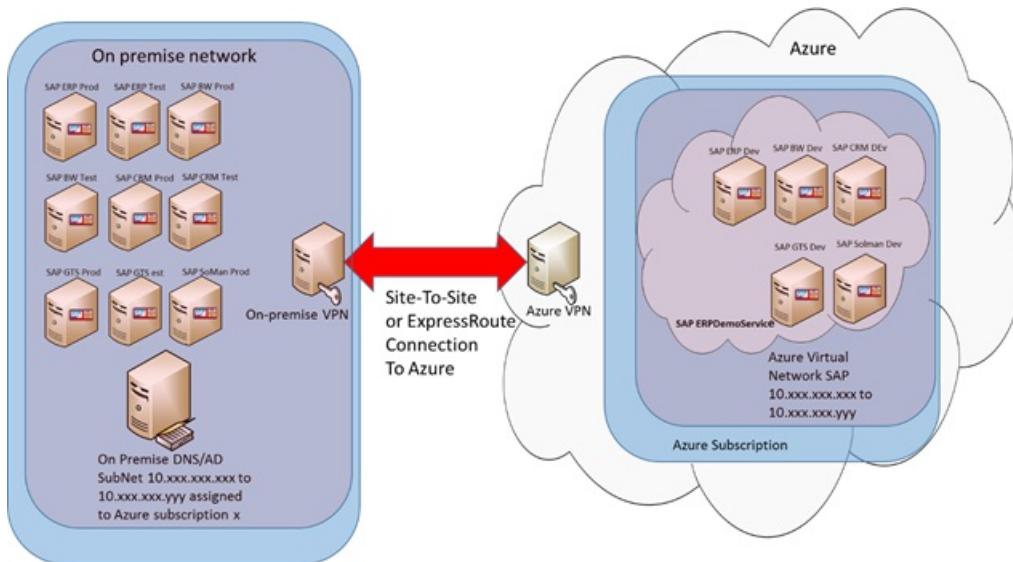
Deploying SAP VMs with Corporate Network Connectivity (Cross-Premises)

You run a SAP landscape and want to divide the deployment between bare-metal for high-end DBMS servers, on-premises virtualized environments for application layers and smaller 2-Tier configured SAP systems and Azure IaaS. The base assumption is that SAP systems within one SAP landscape need to communicate with each other and with many other software components deployed in the company, independent of their deployment form. There also should be no differences introduced by the deployment form for the end user connecting with SAP GUI or other interfaces. These conditions can only be met when we have the on-premises Active Directory/OpenLDAP and DNS services extended to the Azure systems through site-to-site/multi-site connectivity or private connections like Azure ExpressRoute.

In order to get more background on the implementation details of SAP on Azure, we encourage you to read chapter [Concepts of Cloud-Only deployment of SAP instances](#) of this document which explains some of the basics constructs of Azure and how these should be used with SAP applications in Azure.

Scenario of a SAP landscape

The Cross-Premises scenario can be roughly described like in the graphics below:



The scenario shown above describes a scenario where the on-premises AD/OpenLDAP and DNS is extended to Azure. On the on-premises side, a certain IP address range is reserved per Azure subscription. The IP address range will be assigned to an Azure Virtual Network on the Azure side.

Security considerations

The minimum requirement is the use of secure communication protocols such as SSL/TLS for browser access or VPN-based connections for system access to the Azure services. The assumption is that companies handle the VPN connection between their corporate network and Azure very differently. Some companies might blankly open all the ports. Some other companies might want to be very precise in which ports they need to open, etc.

In the table below typical SAP communication ports are listed. Basically it is sufficient to open the SAP gateway port.

| SERVICE | PORT NAME | EXAMPLE <code><nn></code> = 01 | DEFAULT RANGE (MIN-MAX) | COMMENT |
|----------------|--------------------------------------|--------------------------------------|-------------------------|---|
| Dispatcher | <code>sapdp<nn></code> see * | 3201 | 3200 – 3299 | SAP Dispatcher, used by SAP GUI for Windows and Java |
| Message server | <code>sapms<sid></code> see ** | 3600 | free sapms<anySID> | sid = SAP-System-ID |
| Gateway | <code>sapgw<nn></code> see * | 3301 | free | SAP gateway, used for CPIC and RFC communication |
| SAP router | <code>sapdp99</code> | 3299 | free | Only CI (central instance) Service names can be reassigned in /etc/services to an arbitrary value after installation. |

*) nn = SAP Instance Number

**) sid = SAP-System-ID

More detailed information on ports required for different SAP products or services by SAP products can be found here <http://scn.sap.com/docs/DOC-17124>. With this document you should be able to open dedicated ports in the VPN device necessary for specific SAP products and scenarios.

Other security measures when deploying VMs in such a scenario could be to create a [Network Security Group](#) to define access rules.

Dealing with different Virtual Machine Series

In the course of last 12 months Microsoft added many more VM types that differ either in number of vCPUs, memory or more important on hardware it is running on. Not all those VMs are supported with SAP (see supported VM types in SAP Note [1928533](#)). Some of those VMs run on different host hardware generations. These host hardware generations are getting deployed in the granularity of an Azure Scale-Unit. Means cases may arise where the different VM sizes you chose can't be run on the same Scale-Unit. An Availability Set is limited in the ability to span Scale-Units based of different hardware. E.g. if you want to run the DBMS on A5-A11 VMs and the SAP application layer on G-Series VMs, you would be forced to deploy a single SAP system or different SAP systems within different Availability Sets.

Printing on a local network printer from SAP instance in Azure

Printing over TCP/IP in Cross-Premises scenario

Setting up your on-premises TCP/IP based network printers in an Azure VM is overall the same as in your corporate network, assuming you do have a VPN Site-To-Site tunnel or ExpressRoute connection established.

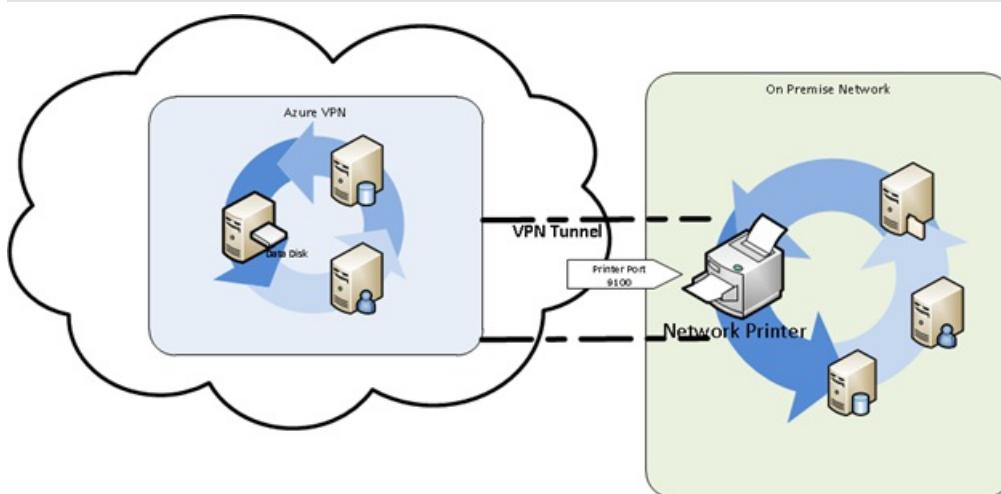


To do this:

- Some network printers come with a configuration wizard which makes it easy to set up your printer in an Azure VM. If no wizard software has been distributed with the printer the "manual" way to set up the printer is to create a new TCP/IP printer port.
- Open Control Panel -> Devices and Printers -> Add a printer
- Choose Add a printer using a TCP/IP address or hostname
- Type in the IP address of the printer
- Printer Port standard 9100
- If necessary install the appropriate printer driver manually.



- like for Windows just follow the standard procedure to install a network printer
- just follow the public Linux guides for [SUSE](#) or [Red Hat](#) on how to add a printer.



Host-based printer over SMB (shared printer) in Cross-Premises scenario

Host-based printers are not network-compatible by design. But a host-based printer can be shared among computers on a network as long as the printer is connected to a powered-on computer. Connect your corporate network either Site-To-Site or ExpressRoute and share your local printer. The SMB protocol uses NetBIOS instead

of DNS as name service. The NetBIOS host name can be different from the DNS host name. The standard case is that the NetBIOS host name and the DNS host name are identical. The DNS domain does not make sense in the NetBIOS name space. Accordingly, the fully qualified DNS host name consisting of the DNS host name and DNS domain must not be used in the NetBIOS name space.

The printer share is identified by a unique name in the network:

- Host name of the SMB host (always needed).
- Name of the share (always needed).
- Name of the domain if printer share is not in the same domain as SAP system.
- Additionally, a user name and a password may be required to access the printer share.

How to:



Share your local printer. In the Azure VM open the Windows Explorer and type in the share name of the printer. A printer installation wizard will guide you through the installation process.



Here are some examples of documentation about configuring network printers in Linux or including a chapter regarding printing in Linux. It will work the same way in an Azure Linux VM as long as the VM is part of a VPN :

- SLES [https://en.opensuse.org/SDB:Printing_via_SMB_\(Samba\)_Share_or_Windows_Share](https://en.opensuse.org/SDB:Printing_via_SMB_(Samba)_Share_or_Windows_Share)
- RHEL https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/s1-printing-smb-printer.html

USB Printer (printer forwarding)

In Azure the ability of the Remote Desktop Services to provide users the access to their local printer devices in a remote session is not available.



More details on printing with Windows can be found here:

<http://technet.microsoft.com/library/jj590748.aspx>.

Integration of SAP Azure Systems into Correction and Transport System (TMS) in Cross-Premises

The SAP Change and Transport System (TMS) needs to be configured to export and import transport request across systems in the landscape. We assume that the development instances of an SAP system (DEV) are located in Azure whereas the quality assurance (QA) and productive systems (PRD) are on-premises. Furthermore, we assume that there is a central transport directory.

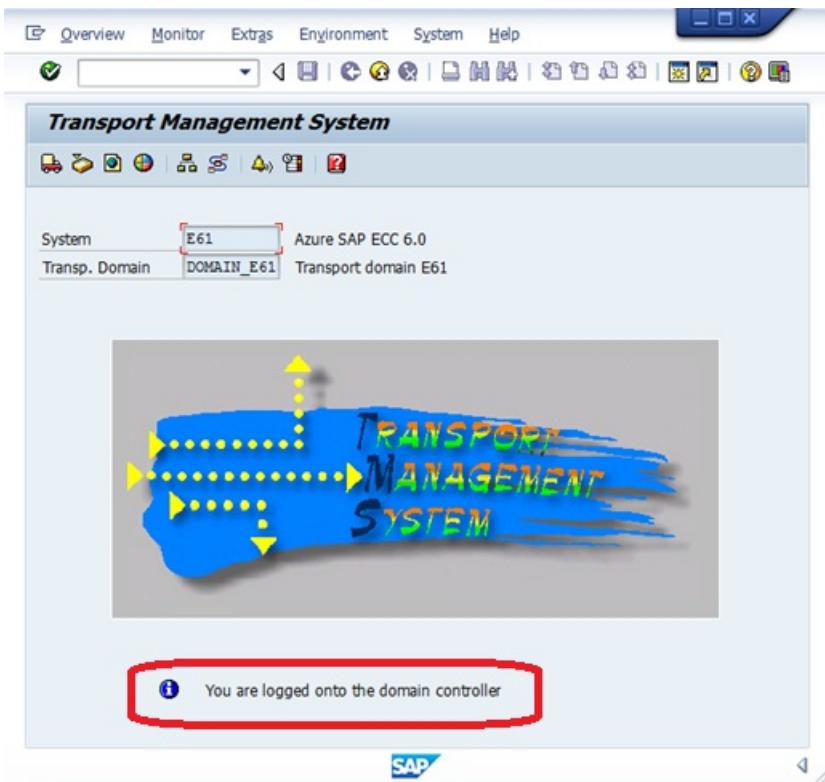
Configuring the Transport Domain

Configure your Transport Domain on the system you designated as the Transport Domain Controller as described in [Configuring the Transport Domain Controller](#). A system user TMSADM will be created and the required RFC destination will be generated. You may check these RFC connections using the transaction SM59. Hostname resolution must be enabled across your transport domain.

How to:

- In our scenario we decided the on-premises QAS system will be the CTS domain controller. Call transaction STMS. The TMS dialog box appears. A Configure Transport Domain dialog box is displayed. (This dialog box only appears if you have not yet configured a transport domain.)

- Make sure that the automatically created user TMSADM is authorized (SM59 -> ABAP Connection -> TMSADM@E61.DOMAIN_E61 -> Details -> Utilities(M) -> Authorization Test). The initial screen of transaction STMS should show that this SAP System is now functioning as the controller of the transport domain as shown here:



Including SAP Systems in the Transport Domain

The sequence of including an SAP system in a transport domain looks as follows:

- On the DEV system in Azure go to the transport system (Client 000) and call transaction STMS. Choose Other Configuration from the dialog box and continue with Include System in Domain. Specify the Domain Controller as target host ([Including SAP Systems in the Transport Domain](#)). The system is now waiting to be included in the transport domain.
- For security reasons, you then have to go back to the domain controller to confirm your request. Choose System Overview and Approve of the waiting system. Then confirm the prompt and the configuration will be distributed.

This SAP system now contains the necessary information about all the other SAP systems in the transport domain. At the same time, the address data of the new SAP system is sent to all the other SAP systems, and the SAP system is entered in the transport profile of the transport control program. Check whether RFCs and access to the transport directory of the domain work.

Continue with the configuration of your transport system as usual as described in the documentation [Change and Transport System](#).

How to:

- Make sure your STMS on premises is configured correctly.
- Make sure the hostname of the Transport Domain Controller can be resolved by your virtual machine on Azure and vice versa.
- Call transaction STMS -> Other Configuration -> Include System in Domain.
- Confirm the connection in the on premises TMS system.
- Configure transport routes, groups and layers as usual.

In site-to-site connected Cross-Premises scenarios, the latency between on-premises and Azure still can be

substantial. If we follow the sequence of transporting objects through development and test systems to production or think about applying transports or support packages to the different systems, you realize that, dependent on the location of the central transport directory, some of the systems will encounter high latency reading or writing data in the central transport directory. The situation is similar to SAP landscape configurations where the different systems are spread through different data centers with substantial distance between the data centers.

In order to work around such latency and have the systems work fast in reading or writing to or from the transport directory, you can setup two STMS transport domains (one for on-premises and one with the systems in Azure and link the transport domains. Please check this documentation which explains the principles behind this concept in the SAP TMS:

http://help.sap.com/saphelp_me60/helpdata/en/c4/6045377b52253de10000009b38f889/content.htm?frameset=/en/57/38dd924eb711d182bf0000e829fbfe/frameset.htm.

How to:

- Set up a transport domain on each location (on-premises and Azure) using transaction STMS
http://help.sap.com/saphelp_nw70ehp3/helpdata/en/44/b4a0b47acc11d1899e0000e829fbdb/content.htm
- Link the domains with a domain link and confirm the link between the two domains.
http://help.sap.com/saphelp_nw73ehp1/helpdata/en/a3/139838280c4f18e10000009b38f8cf/content.htm
- Distribute the configuration to the linked system.

RFC traffic between SAP instances located in Azure and on-premises (Cross-Premises)

RFC traffic between systems which are on-premises and in Azure needs to work. To setup a connection call transaction SM59 in a source system where you need to define an RFC connection towards the target system. The configuration is similar to the standard setup of an RFC Connection.

We assume that in the Cross-Premises scenario, the VMs which run SAP systems that need to communicate with each other are in the same domain. Therefore the setup of an RFC connection between SAP systems does not differ from the setup steps and inputs in on-premises scenarios.

Accessing 'local' fileshares from SAP instances located in Azure or vice versa

SAP instances located in Azure need to access file shares which are within the corporate premises. In addition, on-premises SAP instances need to access file shares which are located in Azure. To enable the file shares you must configure the permissions and sharing options on the local system. Make sure to open the ports on the VPN or ExpressRoute connection between Azure and your datacenter.

Supportability

Azure Monitoring Solution for SAP

In order to enable the monitoring of mission critical SAP systems on Azure the SAP monitoring tools SAPOS COL or SAP Host Agent get data off the Azure Virtual Machine Service host via an Azure Monitoring Extension for SAP. Since the demands by SAP were very specific to SAP applications, Microsoft decided not to generically implement the required functionality into Azure, but leave it for customers to deploy the necessary monitoring components and configurations to their Virtual Machines running in Azure. However, deployment and lifecycle management of the monitoring components will be mostly automated by Azure.

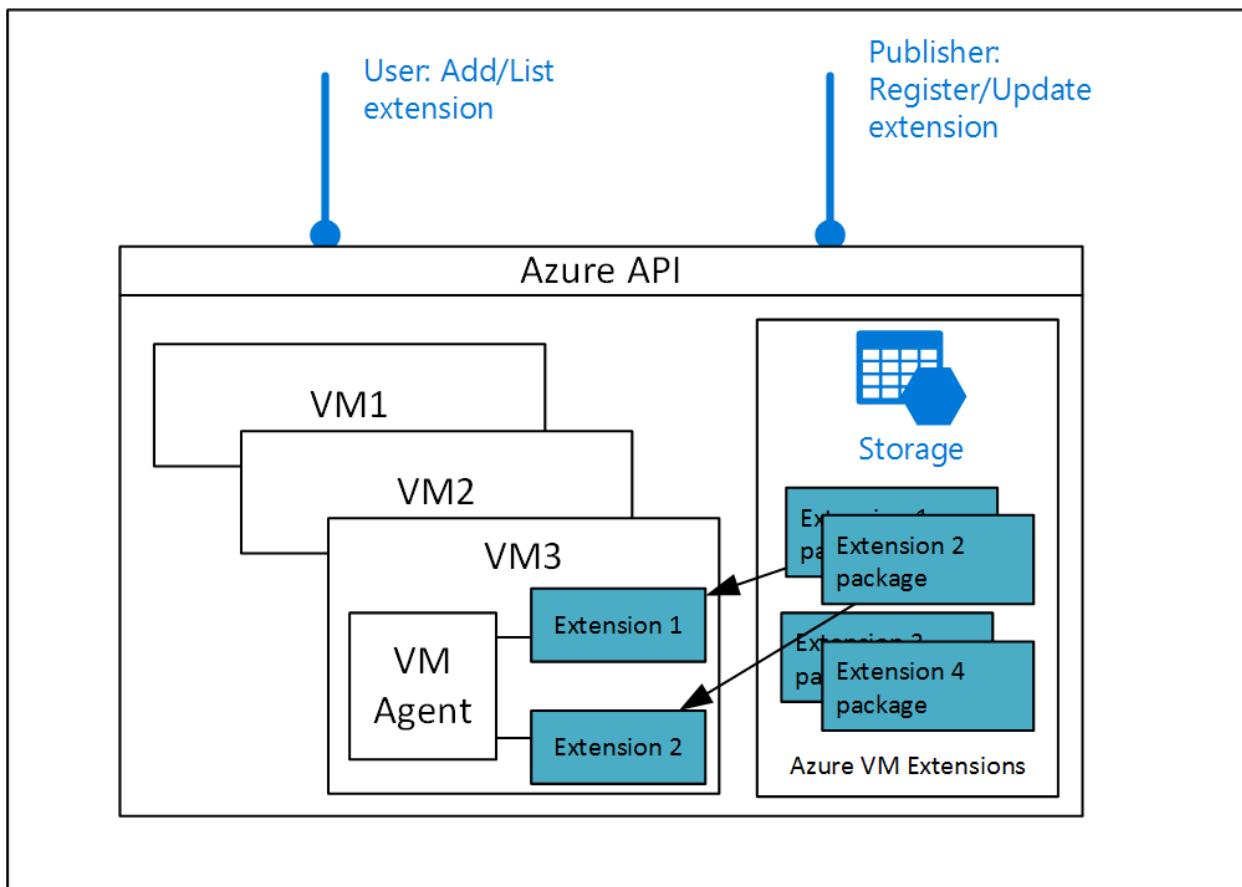
Solution design

The solution developed to enable SAP Monitoring is based on the architecture of Azure VM Agent and Extension framework. The idea of the Azure VM Agent and Extension framework is to allow installation of software application(s) available in the Azure VM Extension gallery within a VM. The principle idea behind this concept is to allow (in cases like the Azure Monitoring Extension for SAP), the deployment of special functionality into a VM and the configuration of such software at deployment time.

Since February 2014, the 'Azure VM Agent' that enables handling of specific Azure VM Extensions within the VM

is injected into Windows VMs by default on VM creation in the Azure Portal. In case of SUSE or Red Hat Linux the VM agent is already part of the Azure Marketplace image. In case one would upload a Linux VM from on-premises to Azure the VM agent has to be installed manually.

The basic building blocks of the Monitoring solution in Azure for SAP looks like this:

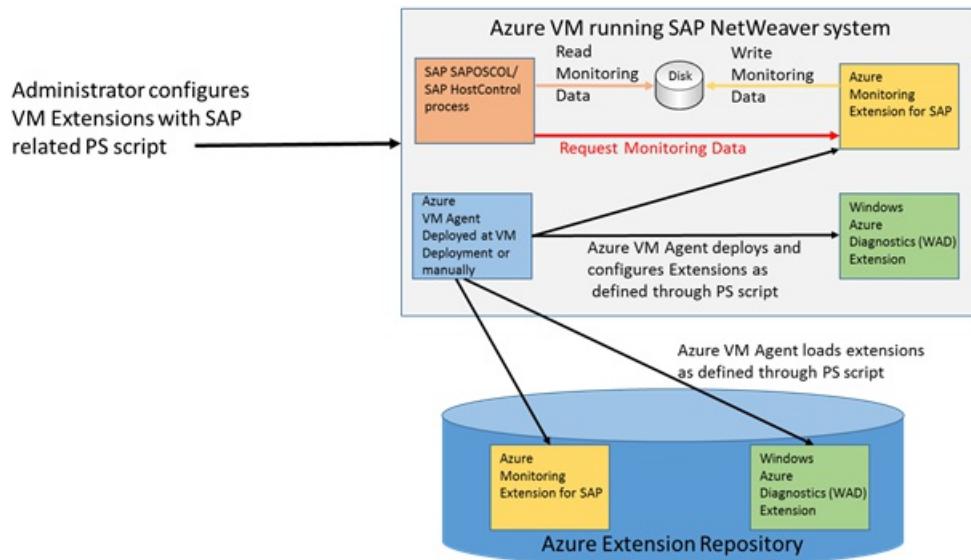


As shown in the block diagram above, one part of the monitoring solution for SAP is hosted in the Azure VM Image and Azure Extension Gallery which is a globally replicated repository that is managed by Azure Operations. It is the responsibility of the joint SAP/MS team working on the Azure implementation of SAP to work with Azure Operations to publish new versions of the Azure Monitoring Extension for SAP. This Azure Monitoring Extension for SAP will use the Microsoft Azure Diagnostics (WAD) Extension or Linux Azure Diagnostics (LAD) to get the necessary information.

When you deploy a new Windows VM, the 'Azure VM Agent' is automatically added into the VM. The function of this agent is to coordinate the loading and configuration of the Azure Extensions for monitoring of SAP NetWeaver Systems. For Linux VMs the Azure VM Agent is already part of the Azure Marketplace OS image.

However, there is a step that still needs to be executed by the customer. This is the enablement and configuration of the performance collection. The process related to the 'configuration' is automated by a PowerShell script or CLI command. The PowerShell script can be downloaded in the Microsoft Azure Script Center as described in the [Deployment Guide](#).

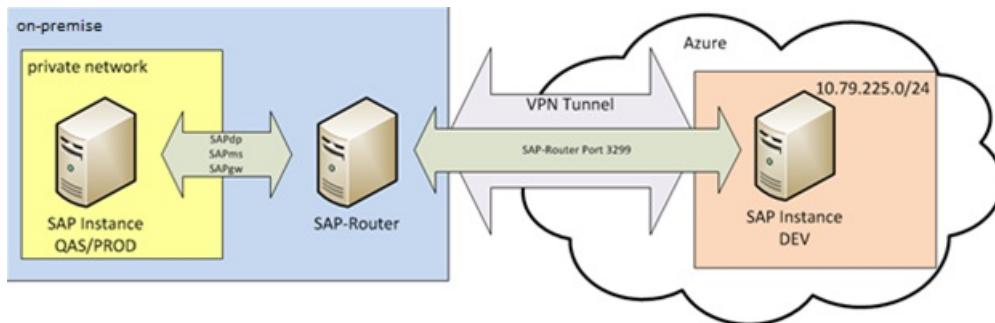
The overall Architecture of the Azure monitoring solution for SAP looks like:



For the exact how-to and for detailed steps of using these PowerShell cmdlets or CLI command during deployments, follow the instructions given in the [Deployment Guide](#).

Integration of Azure located SAP instance into SAProuter

SAP instances running in Azure need to be accessible from SAProuter as well.



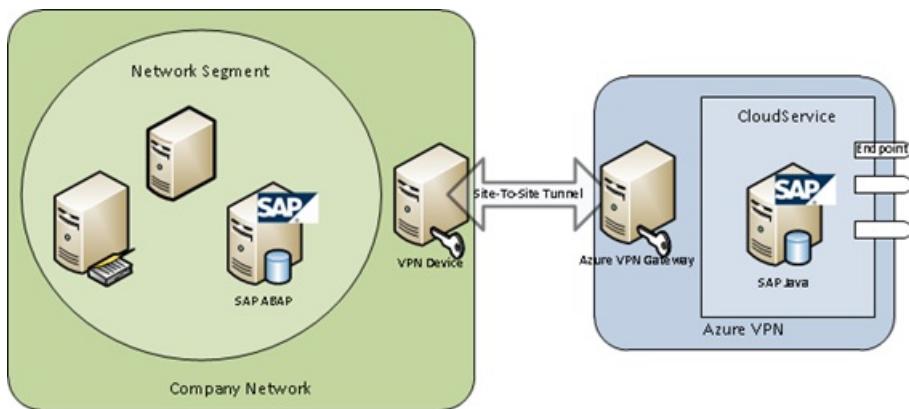
A SAProuter enables the TCP/IP communication between participating systems if there is no direct IP connection. This provides the advantage that no end-to-end connection between the communication partners is necessary on network level. The SAProuter is listening on port 3299 by default. To connect SAP instances through a SAProuter you need to give the SAProuter string and host name with any attempt to connect.

SAP NetWeaver AS Java

So far the focus of the document has been SAP NetWeaver in general or the SAP NetWeaver ABAP stack. In this small section, specific considerations for the SAP Java stack are listed. One of the most important SAP NetWeaver Java exclusively based applications is the SAP Enterprise Portal. Other SAP NetWeaver based applications like SAP PI and SAP Solution Manager use both the SAP NetWeaver ABAP and Java stacks. Therefore, there certainly is a need to consider specific aspects related to the SAP NetWeaver Java stack as well.

SAP Enterprise Portal

The setup of an SAP Portal in an Azure Virtual Machine does not differ from an on premises installation if you are deploying in Cross-Premises scenarios. Since the DNS is done by on-premises, the port settings of the individual instances can be done as configured on-premises. The recommendations and restrictions described in this document so far apply for an application like SAP Enterprise Portal or the SAP NetWeaver Java stack in general.



A special deployment scenario by some customers is the direct exposure of the SAP Enterprise Portal to the Internet while the virtual machine host is connected to the company network via site-to-site VPN tunnel or ExpressRoute. For such a scenario, you have to make sure that specific ports are open and not blocked by firewall or network security group. The same mechanics would need to be applied when you want to connect to an SAP Java instance from on-premises in a Cloud-Only scenario.

The initial portal URI is `http(s):<Portalserver>:5XX00/irj` where the port is formed by `50000 plus (Systemnumber × 100)`. The default portal URI of SAP system 00 is `<dns name>.<azure region>.Cloudapp.azure.com:PublicPort/irj`. For more details, have a look at http://help.sap.com/saphelp_nw70ehp1/helpdata/de/a2/f9d7fed2adc340ab462ae159d19509/frameset.htm.

| PRIORITY | NAME | SOURCE | DESTINATION | SERVICE | ACTION |
|----------|-------------------|--------|-------------|-----------|--------|
| 100 | SAPPortal | Any | Any | Any/50000 | Allow |
| 1000 | default-allow-ssh | Any | Any | TCP/22 | Allow |

If you want to customize the URL and/or ports of your SAP Enterprise Portal, please check this documentation:

- [Change Portal URL](#)
- [Change Default port numbers, Portal port numbers](#)

High Availability (HA) and Disaster Recovery (DR) for SAP NetWeaver running on Azure Virtual Machines

Definition of terminologies

The Term **high availability (HA)** is generally related to a set of technologies that minimizes IT disruptions by providing business continuity of IT services through redundant, fault-tolerant or failover protected components inside the **same** data center. In our case, within one Azure Region.

Disaster recovery (DR) is also targeting minimizing IT services disruption, and their recovery but across **different** data centers, that are usually located hundreds of kilometers away. In our case usually between different Azure Regions within the same geopolitical region or as established by you as a customer.

Overview of High Availability

We can separate the discussion about SAP high availability in Azure into two parts:

- **Azure infrastructure high availability**, e.g. HA of compute (VMs), network, storage etc. and its benefits for

increasing SAP application availability.

- **SAP application high availability**, e.g. HA of SAP software components:

- SAP application servers
- SAP ASCS/SCS instance
- DB server

and how it can be combined with Azure infrastructure HA.

SAP High Availability in Azure has some differences compared to SAP High Availability in an on-premises physical or virtual environment. The following paper from SAP describes standard SAP High Availability configurations in virtualized environments on Windows: <http://scn.sap.com/docs/DOC-44415>. There is no sapinst-integrated SAP-HA configuration for Linux like it exists for Windows. Regarding SAP HA on-premises for Linux find more information here : <http://scn.sap.com/docs/DOC-8541>.

Azure Infrastructure High Availability

There is no single-VM SLA available on Azure Virtual Machines right now. To get an idea how the availability of a single VM might look like you can simply build the product of the different available Azure SLAs: <https://azure.microsoft.com/support/legal/sla/>.

The basis for the calculation is 30 days per month, or 43200 minutes. Therefore, 0.05% downtime corresponds to 21.6 minutes. As usual, the availability of the different services will multiply in the following way:

(Availability Service #1/100) * (Availability Service #2/100) * (Availability Service #3/100) *...

Like:

$(99.95/100) * (99.9/100) * (99.9/100) = 0.9975$ or an overall availability of 99.75%.

Virtual Machine (VM) High Availability

There are two types of Azure platform events that can affect the availability of your virtual machines: planned maintenance and unplanned maintenance.

- 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.
- Unplanned maintenance events occur when the hardware or 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 will automatically migrate your virtual machine from the unhealthy physical server hosting your virtual machine to a healthy physical server. Such events are rare, but may also cause your virtual machine to reboot.

More details can be found in this documentation: <http://azure.microsoft.com/documentation/articles/virtual-machines-manage-availability>

Azure Storage Redundancy

The data in your Microsoft Azure Storage Account is always replicated to ensure durability and high availability, meeting the Azure Storage SLA even in the face of transient hardware failures

Since Azure Storage is keeping 3 images of the data by default, RAID5 or RAID1 across multiple Azure disks are not necessary.

More details can be found in this article: <http://azure.microsoft.com/documentation/articles/storage-redundancy/>

Utilizing Azure Infrastructure VM Restart to Achieve "Higher Availability" of SAP Applications

If you decide not to use functionalities like Windows Server Failover Clustering (WSFC) or a Linux equivalent (the latter one is not supported yet on Azure in combination with SAP software), Azure VM Restart is utilized to protect a SAP System against planned and unplanned downtime of the Azure physical server infrastructure and overall underlying Azure platform.

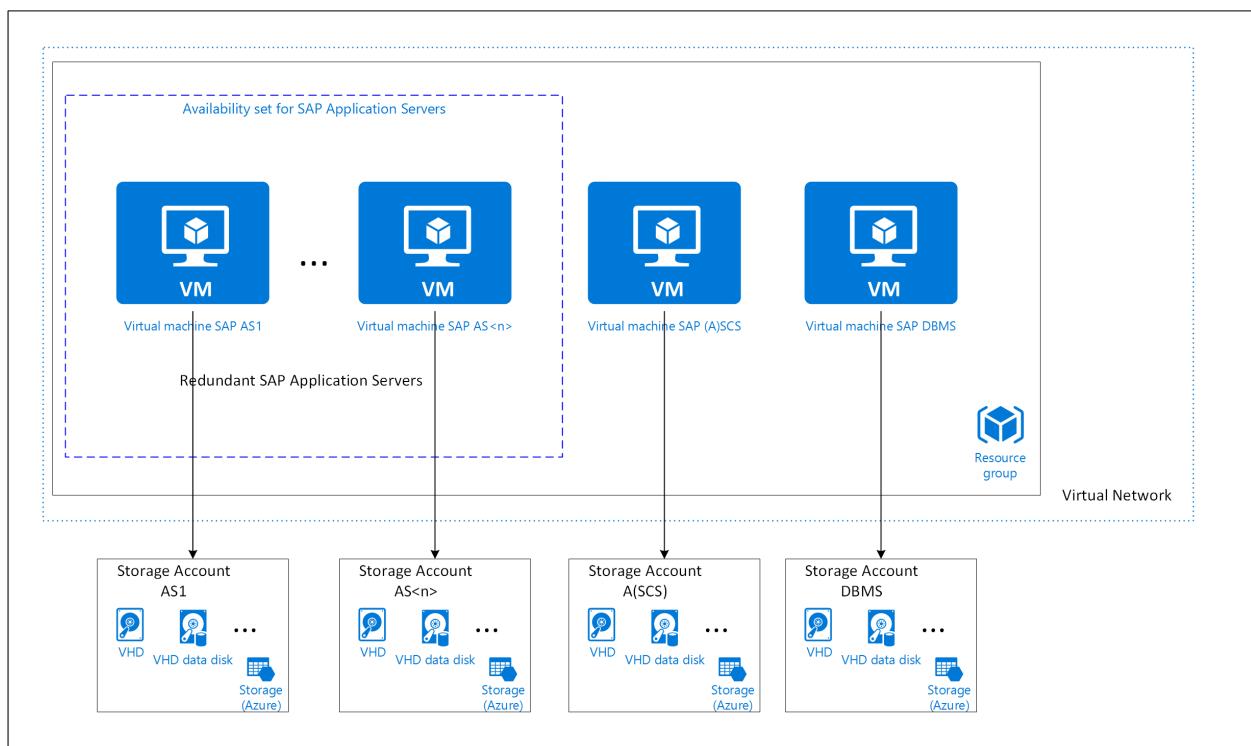
NOTE

It is important to mention that Azure VM Restart primarily protects VMs and NOT applications. VM Restart does not offer high availability for SAP applications, but it does offer a certain level of infrastructure availability and therefore indirectly "higher availability" of SAP systems. There is also no SLA for the time it will take to restart a VM after a planned or unplanned host outage. Therefore, this method of 'high availability' is not suitable for critical components of a SAP system like (A)SCS or DBMS.

Another important infrastructure element for high availability is storage. E.g. Azure Storage SLA is 99,9 % availability. If one deploys all VMs with its disks into a single Azure Storage Account, potential Azure Storage unavailability will cause unavailability of all VMs that are placed in that Azure Storage Account, and also all SAP components running inside of those VMs.

Instead of putting all VMs into one single Azure Storage Account, you can also use dedicated storage accounts for each VM, and in this way increase overall VM and SAP application availability by using multiple independent Azure Storage Accounts.

A sample architecture of a SAP NetWeaver system that uses Azure infrastructure HA could look like this:



For critical SAP components we achieved the following so far :

- High Availability of SAP Application Servers (AS)

SAP application server instances are redundant components. Each SAP AS instance is deployed on its own VM, that is running in a different Azure Fault and Upgrade Domain (see chapters [Fault Domains](#) and [Upgrade Domains](#)). This is ensured by using Azure Availability Sets (see chapter [Azure Availability Sets](#)). Potential planned or unplanned unavailability of an Azure Fault or Upgrade Domain will cause unavailability of a restricted number of VMs with their SAP AS instances. Each SAP AS instance is placed in its own Azure Storage account – potential unavailability of one Azure Storage Account will cause unavailability of only one VM with its SAP AS instance. However, be aware that there is a limit of Azure Storage Accounts within one Azure subscription. To ensure automatic start of (A)SCS instance after the VM reboot, make sure to set Autostart parameter in (A)SCS instance start profile described in chapter [Using Autostart for SAP instances](#). Please also read chapter [High Availability for SAP Application Servers](#) for more details.

- *Higher Availability of SAP (A)SCS instance*

Here we utilize Azure VM Restart to protect the VM with installed SAP (A)SCS instance. In the case of planned or unplanned downtime of Azure servers, VMs will be restarted on another available server. As mentioned earlier, Azure VM Restart primarily protects VMs and NOT applications, in this case the (A)SCS instance. Through the VM Restart we'll reach indirectly "higher availability" of SAP (A)SCS instance. To insure automatic start of (A)SCS instance after the VM reboot, make sure to set Autostart parameter in (A)SCS instance start profile described in chapter [Using Autostart for SAP instances](#). This means the (A)SCS instance as a Single Point of Failure (SPOF) running in a single VM will be the determinative factor for the availability of the whole SAP landscape.

- *Higher Availability of DBMS Server*

Here, similar to the SAP (A)SCS instance use case, we utilize Azure VM Restart to protect the VM with installed DBMS software, and we achieve "higher availability" of DBMS software through VM Restart. DBMS running in a single VM is also a SPOF, and it is the determinative factor for the availability of the whole SAP landscape.

SAP Application High Availability on Azure IaaS

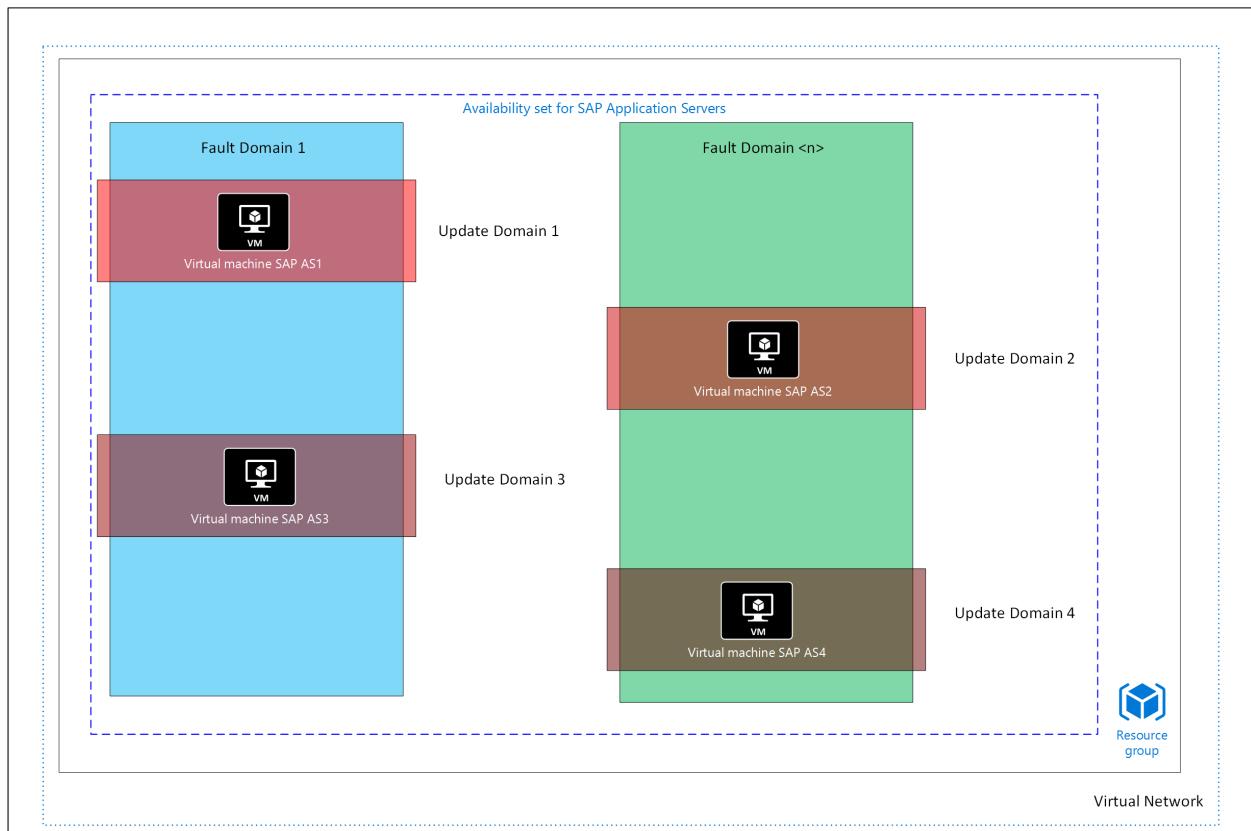
To achieve full SAP system high availability, we need to protect all critical SAP system components, e.g. redundant SAP application servers, and unique components (e.g. Single Point of Failure) like SAP (A)SCS instance and DBMS.

High Availability for SAP Application Servers

For the SAP application servers/dialog instances it's not necessary to think about a specific high availability solution. High availability is simply achieved by redundancy and thereby having enough of them in different virtual machines. They should all be placed in the same Azure Availability Set to avoid that the VMs might be updated at the same time during planned maintenance downtime. The basic functionality which builds on different Upgrade and Fault Domains within an Azure Scale Unit was already introduced in chapter [Upgrade Domains](#). Azure Availability Sets were presented in chapter [Azure Availability Sets](#) of this document.

There is no infinite number of Fault and Upgrade Domains that can be used by an Azure Availability Set within an Azure Scale Unit. This means that putting a number of VMs into one Availability Set sooner or later in the fact that more than one VM ends up in the same Fault or Upgrade Domain

Deploying a few SAP application server instances in their dedicated VMs and assuming that we got 5 Upgrade Domains, the following picture emerges at the end. The actual max number of fault and update domains within an availability set might change in the future :



More details can be found in this documentation: <http://azure.microsoft.com/documentation/articles/virtual-machines-manage-availability>

High Availability for the SAP (A)SCS instance on Windows

Windows Server Failover Cluster (WSFC) is a frequently used solution to protect the SAP (A)SCS instance. It is also integrated into sapinst in form of a "HA installation". At this point in time the Azure infrastructure is not able to provide the functionality to set up the required Windows Server Failover Cluster the same way as it's done on-premises.

As of January 2016 the Azure cloud platform running the Windows operating system does not provide the possibility of using a cluster shared volume on a disk shared between two Azure VMs.

A valid solution though is the usage of 3rd-party software which provides a shared volume by synchronous and transparent disk replication which can be integrated into WSFC. This approach implies that only the active cluster node is able to access one of the disk copies at a point in time. As of January 2016 this HA configuration is supported to protect the SAP (A)SCS instance on Windows guest OS on Azure VMs in combination with 3rd-party software SIOS DataKeeper.

The SIOS DataKeeper solution provides a shared disk cluster resource to Windows Failover Clusters by having:

- An additional Azure VHD attached to each of the virtual machines (VMs) that are in a Windows Cluster configuration
- SIOS DataKeeper Cluster Edition running on both VM nodes
- Having SIOS DataKeeper Cluster Edition configured in a way that it synchronously mirrors the content of the additional VHD attached volume from source VMs to additional VHD attached volume of target VM.
- SIOS DataKeeper is abstracting the source and target local volumes and presenting them to Windows Failover Cluster as a single shared disk.

You can find all details on how to install a Windows Failover Cluster with SIOS Datakeeper and SAP in the [Clustering SAP ASCS Instance using Windows Server Failover Cluster on Azure with SIOS DataKeeper](#) white paper.

High Availability for the SAP (A)SCS instance on Linux

As of Dec 2015 there is also no equivalent to shared disk WSFC for Linux VMs on Azure. Alternative solutions using 3rd-party software like SIOS for Windows are not validated yet for running SAP on Linux on Azure.

High Availability for the SAP database instance

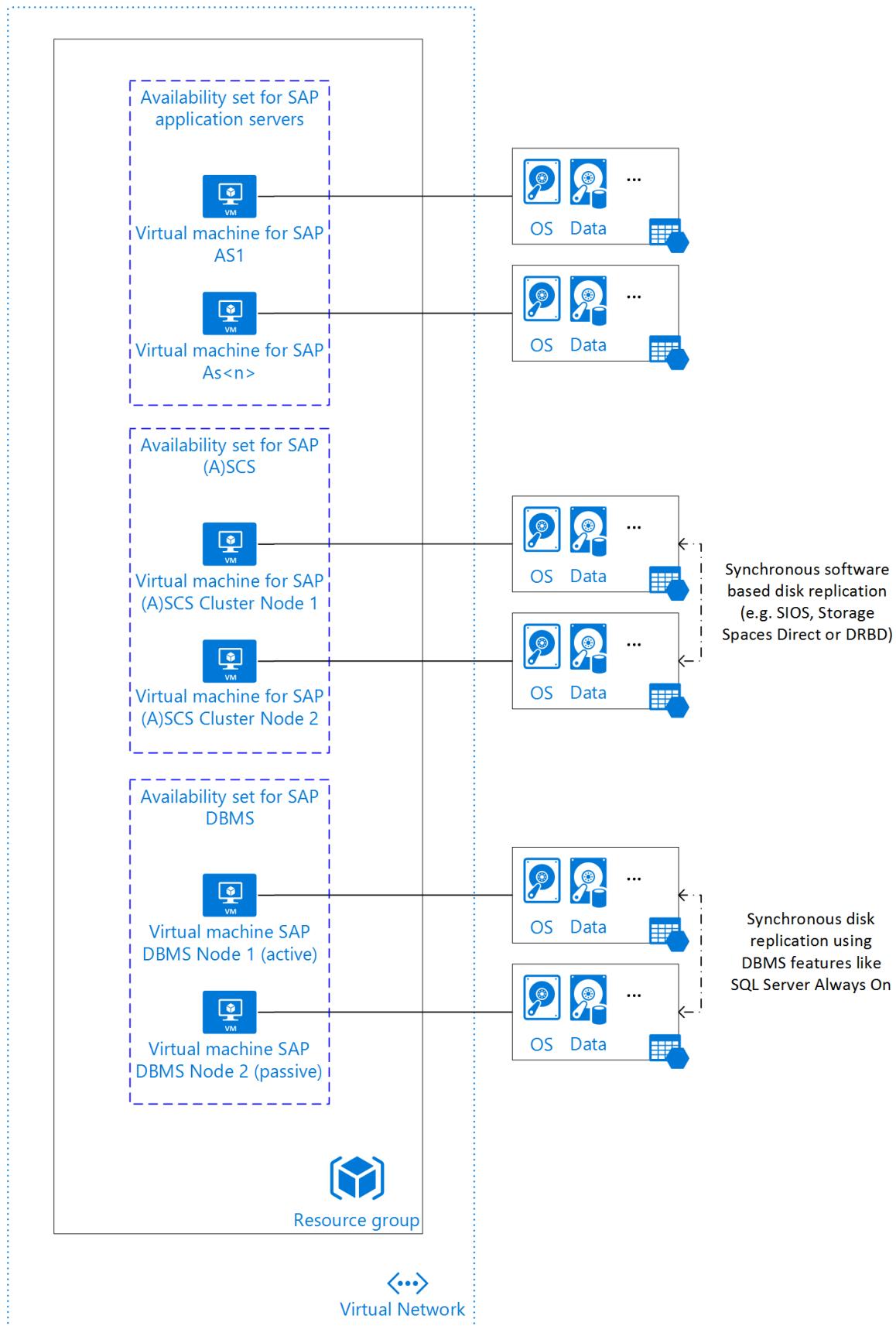
The typical SAP DBMS HA setup is based on two DBMS VMs where DBMS high-availability functionality is used to replicate data from the active DBMS instance to the second VM into a passive DBMS instance.

High Availability and Disaster recovery functionality for DBMS in general as well as specific DBMS are described in the [DBMS Deployment Guide](#).

End-to-End High Availability for the Complete SAP System

Here are two examples of a complete SAP NetWeaver HA architecture in Azure - one for Windows and one for Linux. The concepts as explained below may need to be compromised a bit when you deploy many SAP systems and the number of VMs deployed are exceeding the maximum limit of Storage Accounts per subscription. In such cases, VHDs of VMs need to be combined within one Storage Account. Usually you would do so by combining VHDs of SAP application layer VMs of different SAP systems. We also combined different VHDs of different DBMS VMs of different SAP systems in one Azure Storage Account. Thereby keeping the IOPS limits of Azure Storage Accounts in mind (<https://azure.microsoft.com/documentation/articles/storage-scalability-targets>)





The following Azure constructs are used for the SAP NetWeaver system, to minimize impact by infrastructure issues and host patching:

- The complete system is deployed on Azure (required - DBMS layer, (A)SCS instance and complete application layer need to run in the same location).
- The complete system runs within one Azure subscription (required).

- The complete system runs within one Azure Virtual Network (required).
- The separation of the VMs of one SAP system into three Availability Sets is possible even with all the VMs belonging to the same Virtual Network.
- All VMs running DBMS instances of one SAP system are in one Availability Set. We assume that there is more than one VM running DBMS instances per system since native DBMS high availability features are used, like SQL Server AlwaysOn or Oracle Data Guard.
- All VMs running DBMS instances use their own storage account. DBMS data and log files are replicated from one storage account to another storage account using DBMS high availability functions that synchronize the data. Unavailability of one storage account will cause unavailability of one SQL Windows cluster node, but not the whole SQL Server service.
- All VMs running (A)SCS instance of one SAP system are in one Availability Set. Inside of those VMs is configure Windows Sever Failover Cluster (WSFC) to protect (A)SCS instance.
- All VMs running (A)SCS instances use their own storage account. (A)SCS instance files and SAP global folder are replicated from one storage account to another storage account using SIOS DataKeeper replication. Unavailability of one storage account will cause unavailability of one (A)SCS Windows cluster node, but not the whole (A)SCS service.
- ALL the VMs representing the SAP application server layer are in a third Availability Set.
- ALL the VMs running SAP application servers use their own storage account. Unavailability of one storage account will cause unavailability of one SAP application server, where other SAP AS continue to run.



The architecture for SAP HA on Linux on Azure is basically the same as for Windows as described above. As of Jan 2016 there are two restrictions though :

- only SAP ASE 16 is currently supported on Linux on Azure without any ASE replication features.
- there is no SAP (A)SCS HA solution supported yet on Linux on Azure

As a consequence as of January 2016 a SAP-Linux-Azure system cannot achieve the same availability as a SAP-Windows-Azure system because of missing HA for the (A)SCS instance and the single-instance SAP ASE database.

Using Autostart for SAP instances

SAP offered the functionality to start SAP instances immediately after the start of the OS within the VM. The exact steps were documented in SAP Knowledge Base Article [1909114](#) - How to start SAP instances automatically using parameter Autostart. However, SAP is not recommending to use the setting anymore because there is no control in the order of instance restarts, assuming more than one VM got affected or multiple instances ran per VM. Assuming a typical Azure scenario of one SAP application server instance in a VM and the case of a single VM eventually getting restarted, the Autostart is not really critical and can be enabled by adding this parameter:

```
Autostart = 1
```

Into the start profile of the SAP ABAP and/or Java instance.

NOTE

The Autostart parameter can have some downfalls as well. In more detail, the parameter triggers the start of a SAP ABAP or Java instance when the related Windows/Linux service of the instance is started. That certainly is the case when the operating systems boots up. However, restarts of SAP services are also a common thing for SAP Software Lifecycle Management functionality like SUM or other updates or upgrades. These functionalities are not expecting an instance to be restarted automatically at all. Therefore, the Autostart parameter should be disabled before running such tasks. The Autostart parameter also should not be used for SAP instances that are clustered, like ASCS/SCS/CI.

See additional information regarding autostart for SAP instances here :

- [Start/Stop SAP along with your Unix Server Start/Stop](#)
- [Starting and Stopping SAP NetWeaver Management Agents](#)
- [How to enable auto Start of HANA Database](#)

Larger 3-Tier SAP systems

High-Availability aspects of 3-Tier SAP configurations got discussed in earlier sections already. But what about systems where the DBMS server requirements are too large to have it located in Azure, but the SAP application layer could be deployed into Azure?

Location of 3-Tier SAP configurations

It is not supported to split the application tier itself or the application and DBMS tier between on-premises and Azure. A SAP system is either completely deployed on-premises OR in Azure. It is also not supported to have some of the application servers run on-premises and some others in Azure. That is the starting point of the discussion. We also are not supporting to have the DBMS components of a SAP system and the SAP application server layer deployed in two different Azure Regions. E.g. DBMS in West US and SAP application layer in Central US. Reason for not supporting such configurations is the latency sensitivity of the SAP NetWeaver architecture.

However, over the course of last year data center partners developed co-locations to Azure Regions. These co-locations often are in very close proximity to the physical Azure data centers within an Azure Region. The short distance and connection of assets in the co-location through ExpressRoute into Azure can result in a latency that is less than 2ms. In such cases, to locate the DBMS layer (including storage SAN/NAS) in such a co-location and the SAP application layer in Azure is possible. As of Dec 2015, we don't have any deployments like that. But different customers with non-SAP application deployments are using such approaches already.

Offline Backup of SAP systems

Dependent on the SAP configuration chosen (2-Tier or 3-Tier) there could be a need to backup. The content of the VM itself plus to have a backup of the database. The DBMS related backups are expected to be done with database methods. A detailed description for the different databases, can be found in [DBMS Guide](#). On the other hand, the SAP data can be backed up in an offline manner (including the database content as well) as described in this section or online as described in the next section.

The offline backup would basically require a shutdown of the VM through the Azure Portal and a copy of the base VM disk plus all attached VHDs to the VM. This would preserve a point in time image of the VM and its associated disk. It is recommended to copy the 'backups' into a different Azure Storage Account. Hence the procedure described in chapter [Copying disks between Azure Storage Accounts](#) of this document would apply. Besides the shutdown using the Azure Portal one can also do it via Powershell or CLI as described here :
<https://azure.microsoft.com/documentation/articles/virtual-machines-deploy-rmtemplates-powershell/>

A restore of that state would consist of deleting the base VM as well as the original disks of the base VM and mounted VHDs, copying back the saved VHDs to the original Storage Account and then redeploying the system. This article shows an example how to script this process in Powershell : <http://www.westerndevelopers.com/azure-snapshots/>

Please make sure to install a new SAP license since restoring a VM backup as described above creates a new hardware key.

Online backup of an SAP system

Backup of the DBMS is performed with DBMS specific methods as described in the [DBMS Guide](#).

Other VMs within the SAP system can be backed up using Azure Virtual Machine Backup functionality. Azure Virtual Machine Backup got introduced early in 2015 and meanwhile is a standard method to backup a complete VM in Azure. Azure Backup stores the backups in Azure and allows a restore of a VM again.

NOTE

As of Dec 2015 using VM Backup does NOT keep the unique VM ID which is used for SAP licensing. This means that a restore from a VM backup requires installation of a new SAP license key as the restored VM is considered to be a new VM and not a replacement of the former one which was saved. As of Jan 2016 Azure VM Backup doesn't support VMs that are deployed with Azure Resource Manager yet.



Theoretically VMs that run databases can be backed up in a consistent manner as well if the DBMS systems supports the Windows VSS (Volume Shadow Copy Service)

[https://msdn.microsoft.com/library/windows/desktop/bb968832\(v=vs.85\).aspx](https://msdn.microsoft.com/library/windows/desktop/bb968832(v=vs.85).aspx)) as e.g. SQL Server does. However, be aware that based on Azure VM backups point-in-time restores of databases are not possible. Therefore, the recommendation is to perform backups of databases with DBMS functionality instead of relying on Azure VM Backup

To get familiar with Azure Virtual Machine Backup please start here:

<https://azure.microsoft.com/documentation/articles/backup-azure-vms/>.

Other possibilities are to use a combination of Microsoft Data Protection Manager installed in an Azure VM and Azure Backup to backup/restore databases. More information can be found here:

<https://azure.microsoft.com/documentation/articles/backup-azure-dpm-introduction/>.



There is no equivalent to Windows VSS in Linux. Therefore only file-consistent backups are possible but not application-consistent backups. The SAP DBMS backup should be done using DBMS functionality. The file system which includes the SAP related data can be saved e.g. using tar as described here :

http://help.sap.com/saphelp_nw70ehp2/helpdata/en/d3/c0da3ccb04d35b186041ba6ac301f/content.htm

Azure as DR site for production SAP landscapes

Since Mid 2014, extensions to various components around Hyper-V, System Center and Azure enable the usage of Azure as DR site for VMs running on-premise based on Hyper-V.

A blog detailing how to deploy this solution is documented here:

<http://blogs.msdn.com/b/saponsqlserver/archive/2014/11/19/protecting-sap-solutions-with-azure-site-recovery.aspx>

Summary

The key points of High Availability for SAP systems in Azure are:

- At this point in time, the SAP single point of failure cannot be secured exactly the same way as it can be done in on-premises deployments. The reason is that Shared Disk clusters can't yet be built in Azure without the use of 3rd party software.
- For the DBMS layer you need to use DBMS functionality that does not rely on shared disk cluster technology. Details are documented in the [DBMS Guide](#).
- To minimize the impact of problems within Fault Domains in the Azure infrastructure or host maintenance, you should use Azure Availability Sets:
 - It is recommended to have one Availability Set for the SAP application layer.
 - It is recommended to have a separate Availability Set for the SAP DBMS layer.
 - It is NOT recommended to apply the same Availability set for VMs of different SAP systems.
- For Backup purposes of the SAP DBMS layer, please check the [DBMS Guide](#).
- Backing up SAP Dialog instances makes little sense since it is usually faster to redeploy simple dialog instances.
- Backing up the VM which contains the global directory of the SAP system and with it all the profiles of the different instances, does make sense and should be performed with Windows Backup or e.g. tar on Linux.

Since there are differences between Windows Server 2008 (R2) and Windows Server 2012 (R2), which make it easier to backup using the more recent Windows Server releases, we recommend to run Windows Server 2012 (R2) as Windows guest operating system.

Azure Virtual Machines high availability for SAP NetWeaver

1/17/2017 • 49 min to read • [Edit on GitHub](#)

Azure Virtual Machines is the solution for organizations that need compute, storage, and network resources, in minimal time, and without lengthy procurement cycles. You can use Azure Virtual Machines to deploy classic applications like SAP NetWeaver-based ABAP, Java, and an ABAP+Java stack. Extend reliability and availability without additional on-premises resources. Azure Virtual Machines supports cross-premises connectivity, so you can integrate Azure Virtual Machines into your organization's on-premises domains, private clouds, and SAP system landscape.

In this article, 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:

- Find the right SAP Notes and installation guides, listed in the [Resources](#) section. This article complements SAP installation documentation and SAP Notes, which are the primary resources that can help you install and deploy SAP software on specific platforms.
- Learn the differences between the Azure Resource Manager deployment model and the Azure classic deployment model.
- Learn about Windows Server Failover Clustering quorum modes, so you can select the model that is right for your Azure deployment.
- Learn about Windows Server Failover Clustering shared storage in Azure services.
- Learn how to help protect 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, in Azure.
- Follow a step-by-step example of an installation and configuration of a high-availability SAP system in a Windows Server Failover Clustering cluster in Azure by using Azure Resource Manager.
- Learn about additional steps required to use Windows Server Failover Clustering in Azure, but which are not needed in an on-premises deployment.

To simplify deployment and configuration, in this article, we use the SAP three-tier high-availability Resource Manager templates. The templates automate deployment of the entire infrastructure that you need for a high-availability SAP system. The infrastructure also supports SAP Application Performance Standard (SAPS) sizing of your SAP system.

Prerequisites

Before you start, make sure that you meet the prerequisites that are described in the following sections. Also, be sure to check all resources listed in the [Resources](#) section.

In this article, we use Azure Resource Manager templates for [three-tier SAP NetWeaver](#). For a helpful overview of templates, see [SAP Azure Resource Manager templates](#).

Resources

These articles cover SAP deployments in Azure:

- [Azure Virtual Machines planning and implementation for SAP NetWeaver](#)
- [Azure Virtual Machines deployment for SAP NetWeaver](#)

- [Azure Virtual Machines DBMS deployment for SAP NetWeaver](#)
- [Azure Virtual Machines high availability for SAP NetWeaver \(this guide\)](#)

NOTE

Whenever possible, we give you a link to the referring SAP installation guide (see the [SAP installation guides](#)). For prerequisites and information about the installation process, it's a good idea to read the SAP NetWeaver installation guides carefully. This article covers only specific tasks for SAP NetWeaver-based systems that you can use with Azure Virtual Machines.

These SAP Notes are related to the topic of SAP in Azure:

| NOTE NUMBER | TITLE |
|-------------------------|--|
| 1928533 | SAP Applications on Azure: Supported Products and Sizing |
| 2015553 | SAP on Microsoft Azure: Support Prerequisites |
| 1999351 | Enhanced Azure Monitoring for SAP |
| 2178632 | Key Monitoring Metrics for SAP on Microsoft Azure |
| 1999351 | Virtualization on Windows: Enhanced Monitoring |
| 2243692 | Use of Azure Premium SSD Storage for SAP DBMS Instance |

Learn more about the [limitations of Azure subscriptions](#), including general default limitations and maximum limitations.

High-availability SAP with Azure Resource Manager vs. the Azure classic deployment model

The Azure Resource Manager and Azure classic deployment models are different in the following areas:

- Resource groups
- Azure internal load balancer dependency on the Azure resource group
- Support for SAP multi-SID scenarios

Resource groups

In Azure Resource Manager, you can use resource groups to manage all the application resources in your Azure subscription. An integrated approach, in a resource group, all resources have the same life cycle. For example, all resources are created at the same time and they are deleted at the same time. Learn more about [resource groups](#).

Azure internal load balancer dependency on the Azure resource group

In the Azure classic deployment model, there is a dependency between the Azure internal load balancer (Azure Load Balancer service) and the cloud service group. Every internal load balancer needs one cloud service group.

In Azure Resource Manager, you don't need an Azure resource group to use Azure Load Balancer. The environment is simpler and more flexible.

Support for SAP multi-SID scenarios

In Azure Resource Manager, you can install multiple SAP system identifier (SID) ASCS/SCS instances in one cluster. Multi-SID instances are possible because of support for multiple IP addresses for each Azure internal load balancer.

To use the Azure classic deployment model, follow the procedures described in [SAP NetWeaver in Azure: Clustering SAP ASCS/SCS instances by using Windows Server Failover Clustering in Azure with SIOS DataKeeper](#).

IMPORTANT

We strongly recommend that you use the Azure Resource Manager deployment model for your SAP installations. It offers many benefits that are not available in the classic deployment model. Learn more about Azure [deployment models](#).

Windows Server Failover Clustering

Windows Server Failover Clustering is the foundation of a high-availability SAP ASCS/SCS installation and DBMS in Windows.

A failover cluster is a group of 1+n independent servers (nodes) that work together to increase the availability of applications and services. If a node failure occurs, Windows Server Failover Clustering calculates the number of failures that can occur while maintaining a healthy cluster to provide applications and services. You can choose from different quorum modes to achieve failover clustering.

Quorum modes

You can choose from four quorum modes when you use Windows Server Failover Clustering:

- **Node Majority.** Each node of the cluster can vote. The cluster functions only with a majority of votes, that is, with more than half the votes. We recommend this option for clusters that have an uneven number of nodes. For example, three nodes in a seven-node cluster can fail, and the cluster stills achieves a majority and continues to run.
- **Node and Disk Majority.** Each node and a designated disk (a disk witness) in the cluster storage can vote when they are available and in communication. The cluster functions only with a majority of the votes, that is, with more than half the votes. This mode makes sense in a cluster environment with an even number of nodes. If half the nodes and the disk are online, the cluster remains in a healthy state.
- **Node and File Share Majority.** Each node plus a designated file share (a file share witness) that the administrator creates can vote, regardless of whether the nodes and file share are available and in communication. The cluster functions only with a majority of the votes, that is, with more than half the votes. This mode makes sense in a cluster environment with an even number of nodes. It's similar to the Node and Disk Majority mode, but it uses a witness file share instead of a witness disk. This mode is easy to implement, but if the file share itself is not highly available, it might become a single point of failure.
- **No Majority: Disk Only.** The cluster has a quorum if one node is available and in communication with a specific disk in the cluster storage. Only the nodes that also are in communication with that disk can join the cluster. We recommend that you do not use this mode.

Windows Server Failover Clustering on-premises

Figure 1 shows a cluster of two nodes. If the network connection between the nodes fails and both nodes stay up and running, a quorum disk or file share determines which node will continue to provide the cluster's applications and services. The node that has access to the quorum disk or file share is the node that ensures that services continue.

Because this example uses a two-node cluster, we use the Node and File Share Majority quorum mode. The Node and Disk Majority also is a valid option. In a production environment, we recommend that you use a quorum disk. You can use network and storage system technology to make it highly available.

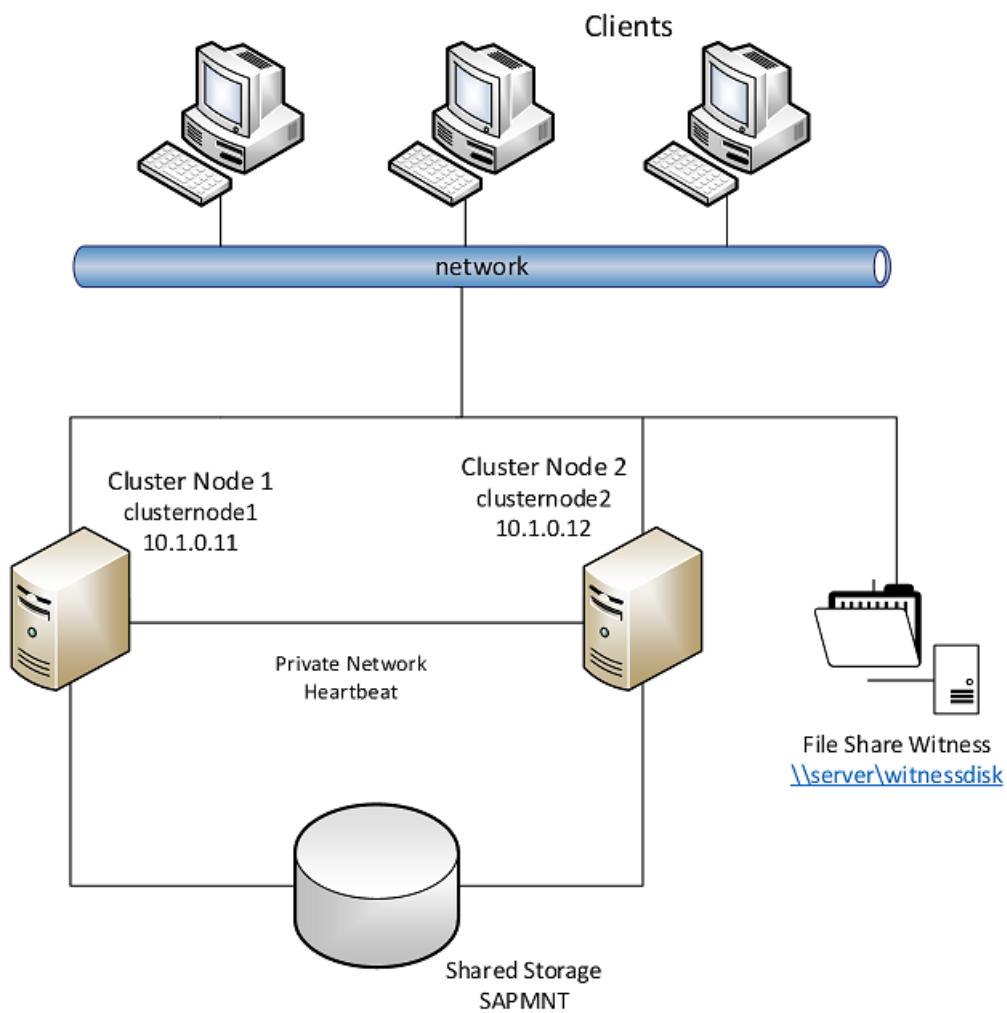


Figure 1: Example of a Windows Server Failover Clustering configuration for SAP ASCS/SCS in Azure

Shared storage

Figure 1 also shows a two-node shared storage cluster. In an on-premises shared storage cluster, all nodes in the cluster detect shared storage. A locking mechanism protects the data from corruption. All nodes can detect if another node fails. If one node fails, the remaining node takes ownership of the storage resources and ensures the availability of services.

NOTE

You don't need shared disks for high availability with some DBMS applications, like with SQL Server. SQL Server Always On replicates DBMS data and log files from the local disk of one cluster node to the local disk of another cluster node. In that case, the Windows cluster configuration doesn't need a shared disk.

Networking and name resolution

Client computers reach the cluster over a virtual IP address and a virtual host name that the DNS server provides. The on-premises nodes and the DNS server can handle multiple IP addresses.

In a typical setup, you use two or more network connections:

- A dedicated connection to the storage
- A cluster-internal network connection for the heartbeat
- A public network that clients use to connect to the cluster

Windows Server Failover Clustering in Azure

Compared to bare metal or private cloud deployments, Azure Virtual Machines requires additional steps to

configure Windows Server Failover Clustering. When you build a shared cluster disk, you need to set several IP addresses and virtual host names for the SAP ASCS/SCS instance.

In this article, we discuss key concepts and the additional steps required to build an SAP high-availability central services cluster in Azure. We show you how to set up the third-party tool SIOS DataKeeper, and how to configure the Azure internal load balancer. You can use these tools to create a Windows failover cluster with a file share witness in Azure.

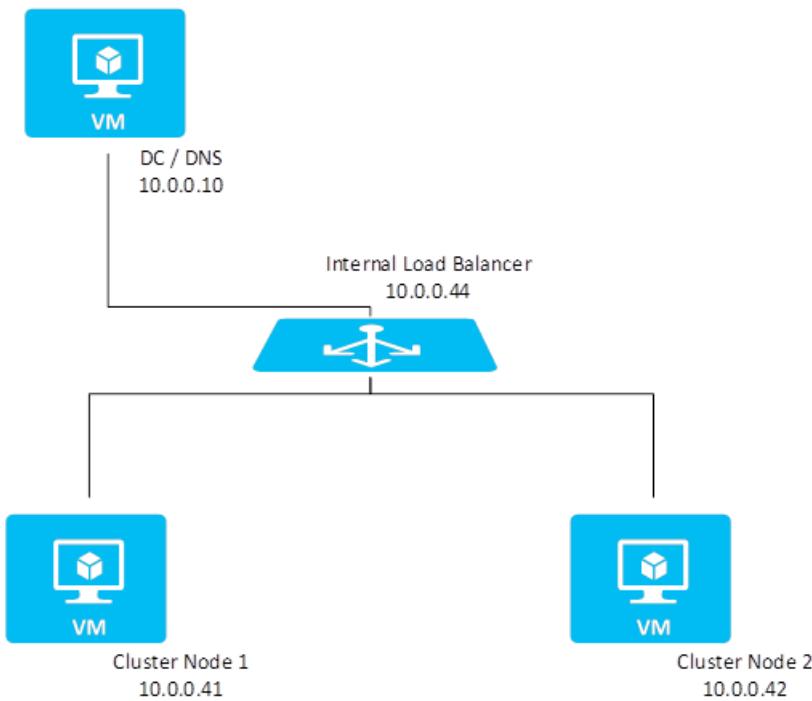


Figure 2: Windows Server Failover Clustering configuration in Azure without a shared disk

Shared disk in Azure with SIOS DataKeeper

You need cluster shared storage for a high-availability SAP ASCS/SCS instance. As of September 2016, Azure doesn't offer shared storage that you can use to create a shared storage cluster. You can use third-party software SIOS DataKeeper Cluster Edition to create a mirrored storage that simulates cluster shared storage. The SIOS solution provides real-time synchronous data replication. This is how you can create a shared disk resource for a cluster:

1. Attach an additional Azure virtual hard disk (VHD) to each of the virtual machines (VMs) in a Windows cluster configuration.
2. Run SIOS DataKeeper Cluster Edition on both virtual machine nodes.
3. Configure SIOS DataKeeper Cluster Edition so that it mirrors the content of the additional VHD attached volume from the source virtual machine to the additional VHD attached volume of the target virtual machine. SIOS DataKeeper abstracts the source and target local volumes, and then presents them to Windows Server Failover Clustering as one shared disk.

Get more information about [SIOS DataKeeper](#).

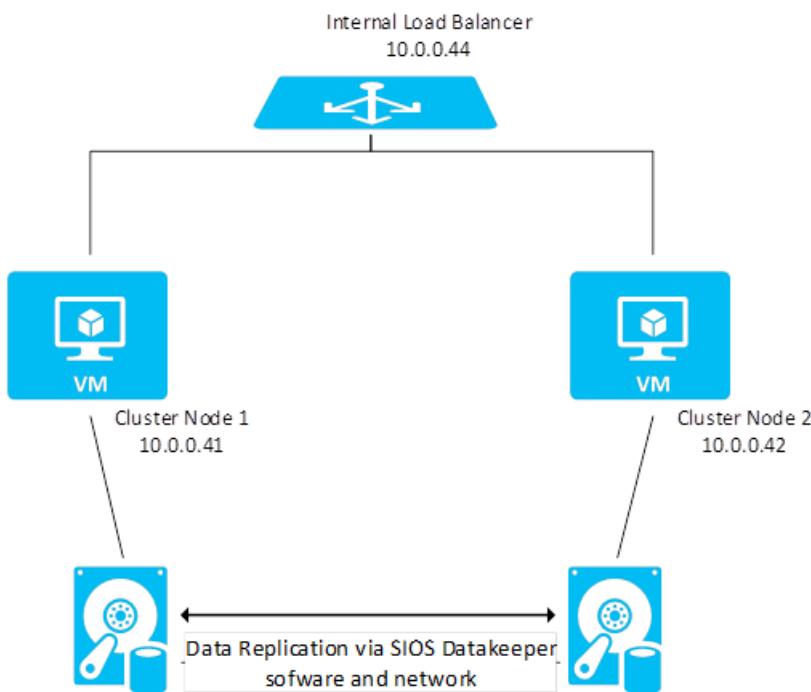


Figure 3: Windows Server Failover Clustering configuration in Azure with SIOS DataKeeper

NOTE

You don't need shared disks for high availability with some DBMS products, like SQL Server. SQL Server Always On replicates DBMS data and log files from the local disk of one cluster node to the local disk of another cluster node. In this case, the Windows cluster configuration doesn't need a shared disk.

Name resolution in Azure

The Azure cloud platform doesn't offer the option to configure virtual IP addresses, such as floating IP addresses. You need an alternative solution to set up a virtual IP address to reach the cluster resource in the cloud. Azure has an internal load balancer in the Azure Load Balancer service. With the internal load balancer, clients reach the cluster over the cluster virtual IP address. You need to deploy the internal load balancer in the resource group that contains the cluster nodes. Then, configure all necessary port forwarding rules with the probe ports of the internal load balancer. The clients can connect via the virtual host name. The DNS server resolves the cluster IP address, and the internal load balancer handles port forwarding to the active node of the cluster.

SAP NetWeaver high availability in Azure Infrastructure-as-a-Service (IaaS)

To achieve SAP application high availability, such as for SAP software components, you need to protect the following components:

- SAP Application Server instance
- SAP ASCS/SCS instance
- DBMS server

For more information about protecting SAP components in high-availability scenarios, see [Azure Virtual Machines planning and implementation for SAP NetWeaver](#).

High-availability SAP Application Server

You usually don't need a specific high-availability solution for the SAP Application Server and dialog instances. You achieve high availability by redundancy, and you'll configure multiple dialog instances in different instances of Azure Virtual Machines. You should have at least two SAP application instances installed in two instances of Azure

Virtual Machines.

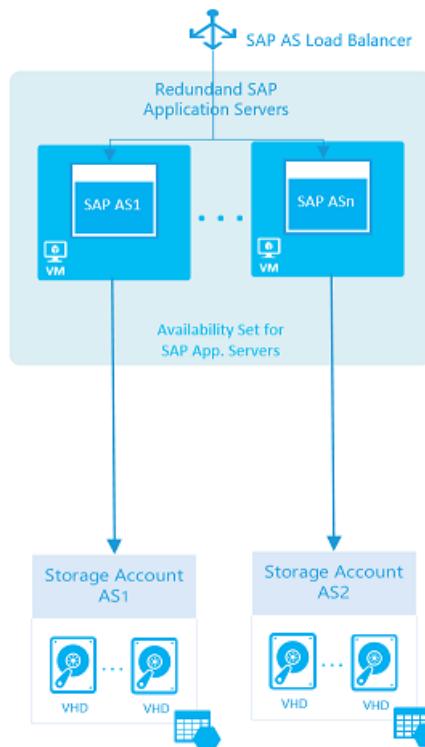


Figure 4: High-availability SAP Application Server

You must place all virtual machines that host SAP Application Server instances in the same Azure availability set. An Azure availability set ensures that:

- All virtual machines are part of the same upgrade domain. An upgrade domain, for example, makes sure that the virtual machines aren't updated at the same time during planned maintenance downtime.
- All virtual machines are part of the same fault domain. A fault domain, for example, makes sure that virtual machines are deployed so that no single point of failure affects the availability of all virtual machines.

Learn more about how to [manage the availability of virtual machines](#).

Because the Azure storage account is a potential single point of failure, it's important to have at least two Azure storage accounts, in which at least two virtual machines are distributed. In an ideal setup, the disks of each virtual machine that is running an SAP dialog instance would be deployed in a different storage account.

High-availability SAP ASCS/SCS instance

Figure 5 is an example of a high-availability SAP ASCS/SCS instance.

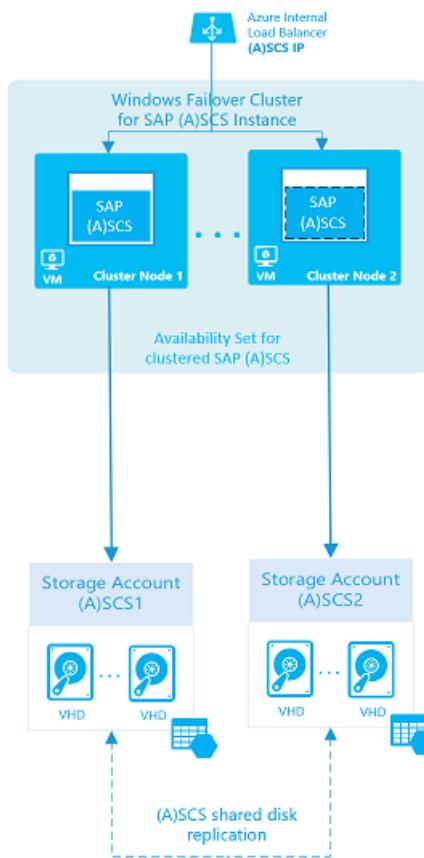


Figure 5: High-availability SAP ASCS/SCS instance

SAP ASCS/SCS instance high availability with Windows Server Failover Clustering in Azure

Compared to bare metal or private cloud deployments, Azure Virtual Machines requires additional steps to configure Windows Server Failover Clustering. To build a Windows failover cluster, you need a shared cluster disk, several IP addresses, several virtual host names, and an Azure internal load balancer for clustering an SAP ASCS/SCS instance. We discuss this in more detail later in the article.

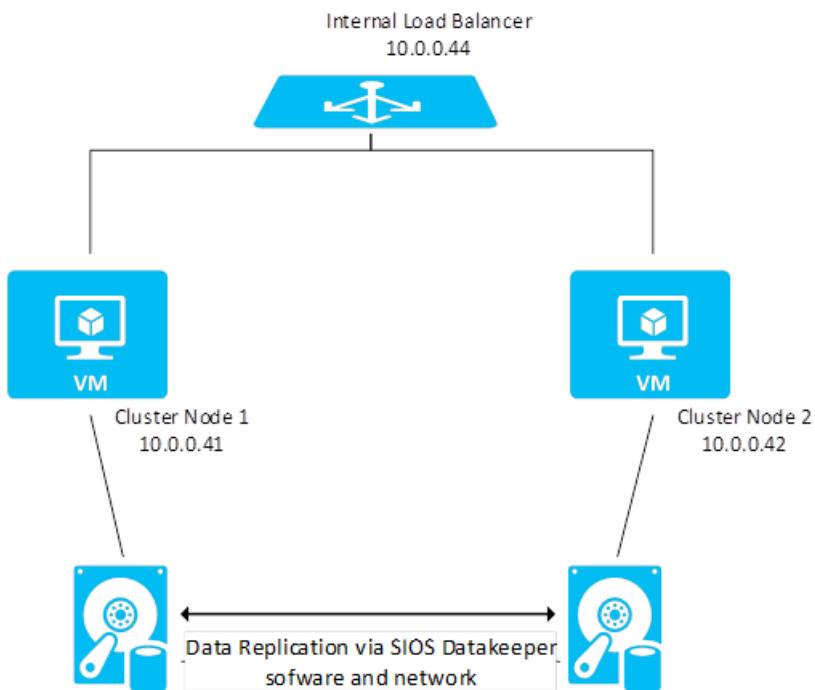


Figure 6: Windows Server Failover Clustering for an SAP ASCS/SCS configuration in Azure with SIOS DataKeeper

High-availability DBMS instance

The DBMS also is a single point of contact in an SAP system. You need to protect it by using a high-availability

solution. Figure 7 shows a SQL Server Always On high-availability solution in Azure, with Windows Server Failover Clustering and the Azure internal load balancer. SQL Server Always On replicates DBMS data and log files by using its own DBMS replication. In this case, you don't need cluster shared disks, which simplifies the entire setup.

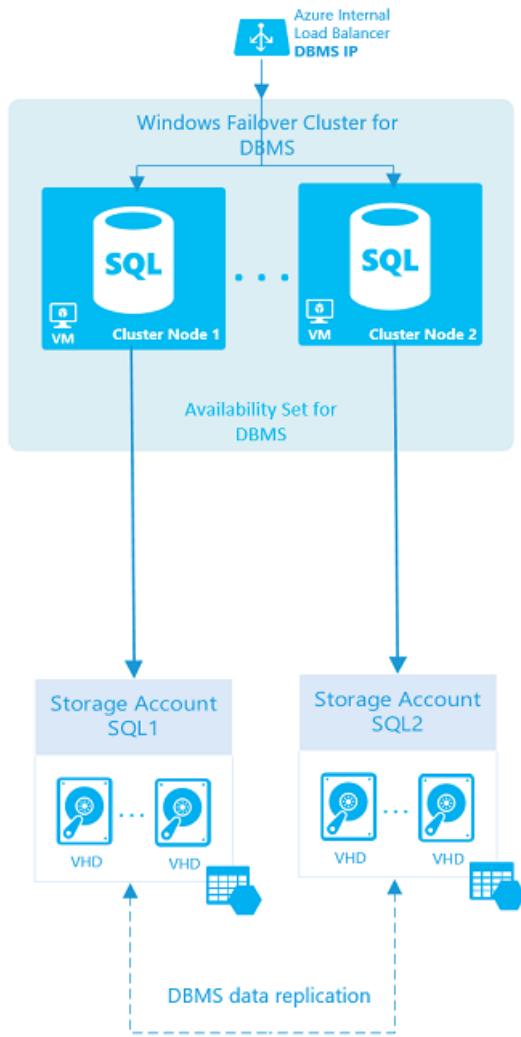


Figure 7: Example of a high-availability SAP DBMS, with SQL Server Always On

For more information about clustering SQL Server in Azure by using the Azure Resource Manager deployment model, see these articles:

- [Configure Always On availability group in Azure Virtual Machines manually by using Resource Manager] [virtual-machines-windows-portal-sql-alwayson-availability-groups-manual]
- [Configure an Azure internal load balancer for an Always On availability group in Azure][virtual-machines-windows-portal-sql-alwayson-int-listener]

End-to-end high-availability deployment scenarios

Deployment scenario using Architectural Template 1

Figure 8 shows an example of an SAP NetWeaver high-availability architecture in Azure for **one** SAP system. This scenario is set up as follows:

- One dedicated cluster is used for the SAP ASCS/SCS instance.
- One dedicated cluster is used for the DBMS instance.
- SAP Application Server instances are deployed in their own dedicated VMs.

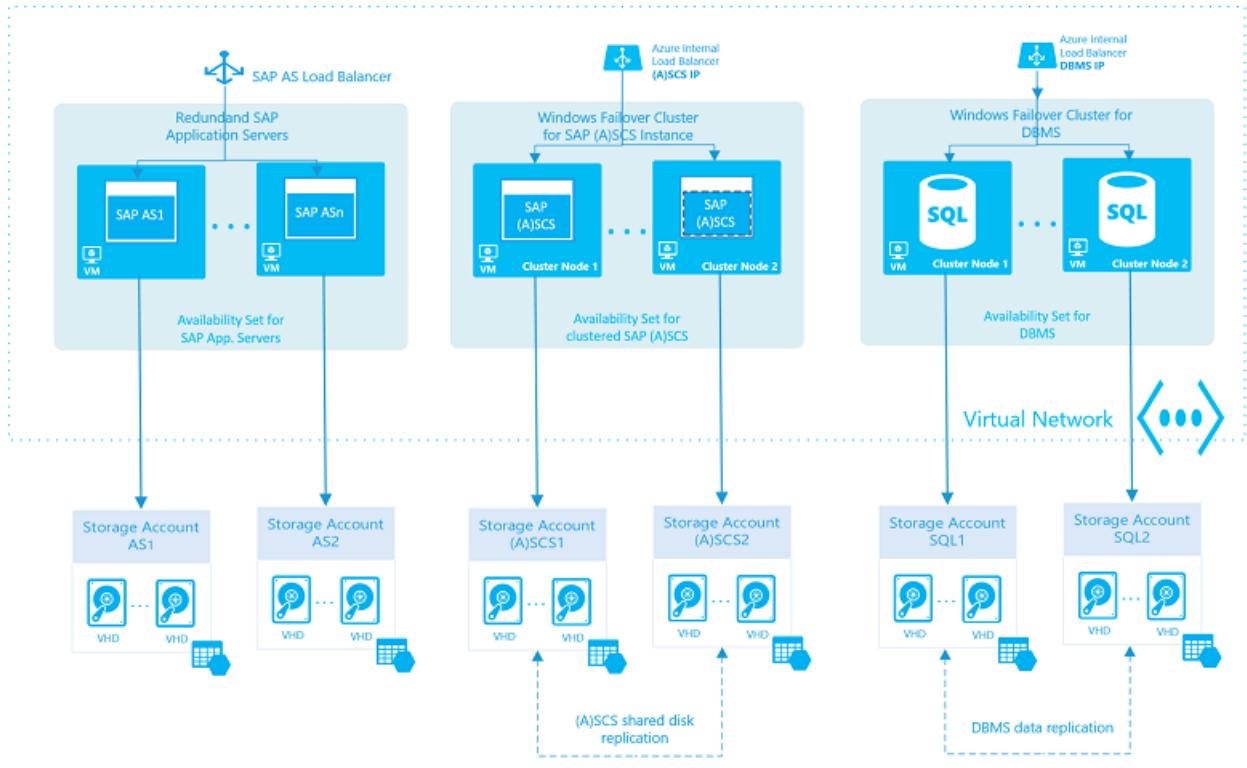


Figure 8: SAP high-availability Architectural Template 1, dedicated clusters for ASCS/SCS and DBMS

Deployment scenario using Architectural Template 2

Figure 9 shows an example of an SAP NetWeaver high-availability architecture in Azure for **one** SAP system. This scenario is set up as follows:

- One dedicated cluster is used for **both** the SAP ASCS/SCS instance and the DBMS.
- SAP Application Server instances are deployed in own dedicated VMs.

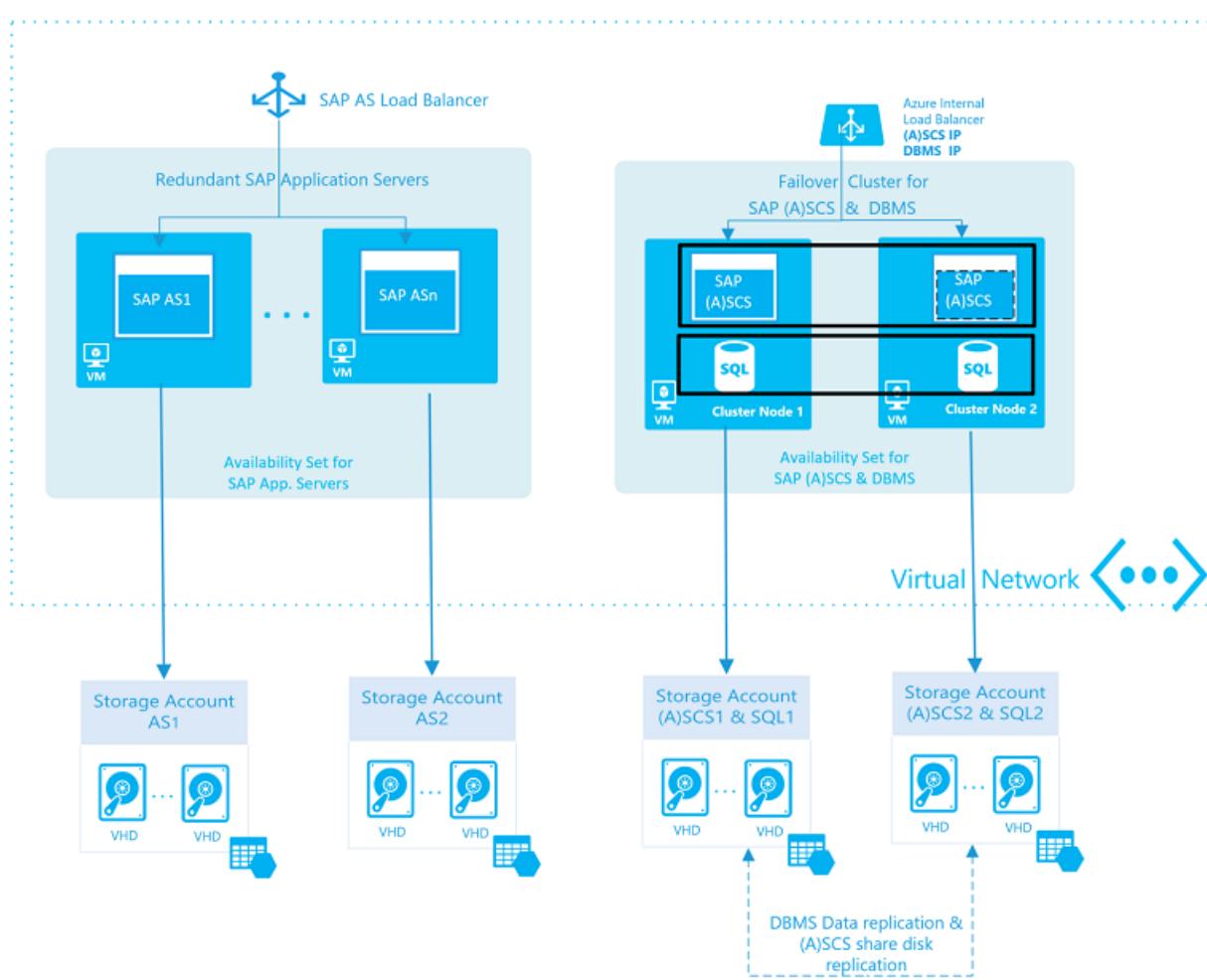


Figure 9: SAP high-availability Architectural Template 2, with a dedicated cluster for ASCS/SCS and a dedicated cluster for DBMS

Deployment scenario using Architectural Template 3

Figure 10 shows an example of an SAP NetWeaver high-availability architecture in Azure for **two** SAP systems, with <SID1> and <SID2>. This scenario is set up as follows:

- One dedicated cluster is used for **both** the SAP ASCS/SCS SID1 instance *and* the SAP ASCS/SCS SID2 instance (one cluster).
- One dedicated cluster is used for DBMS SID1, and another dedicated cluster is used for DBMS SID2 (two clusters).
- SAP Application Server instances for the SAP system SID1 have their own dedicated VMs.
- SAP Application Server instances for the SAP system SID2 have their own dedicated VMs.

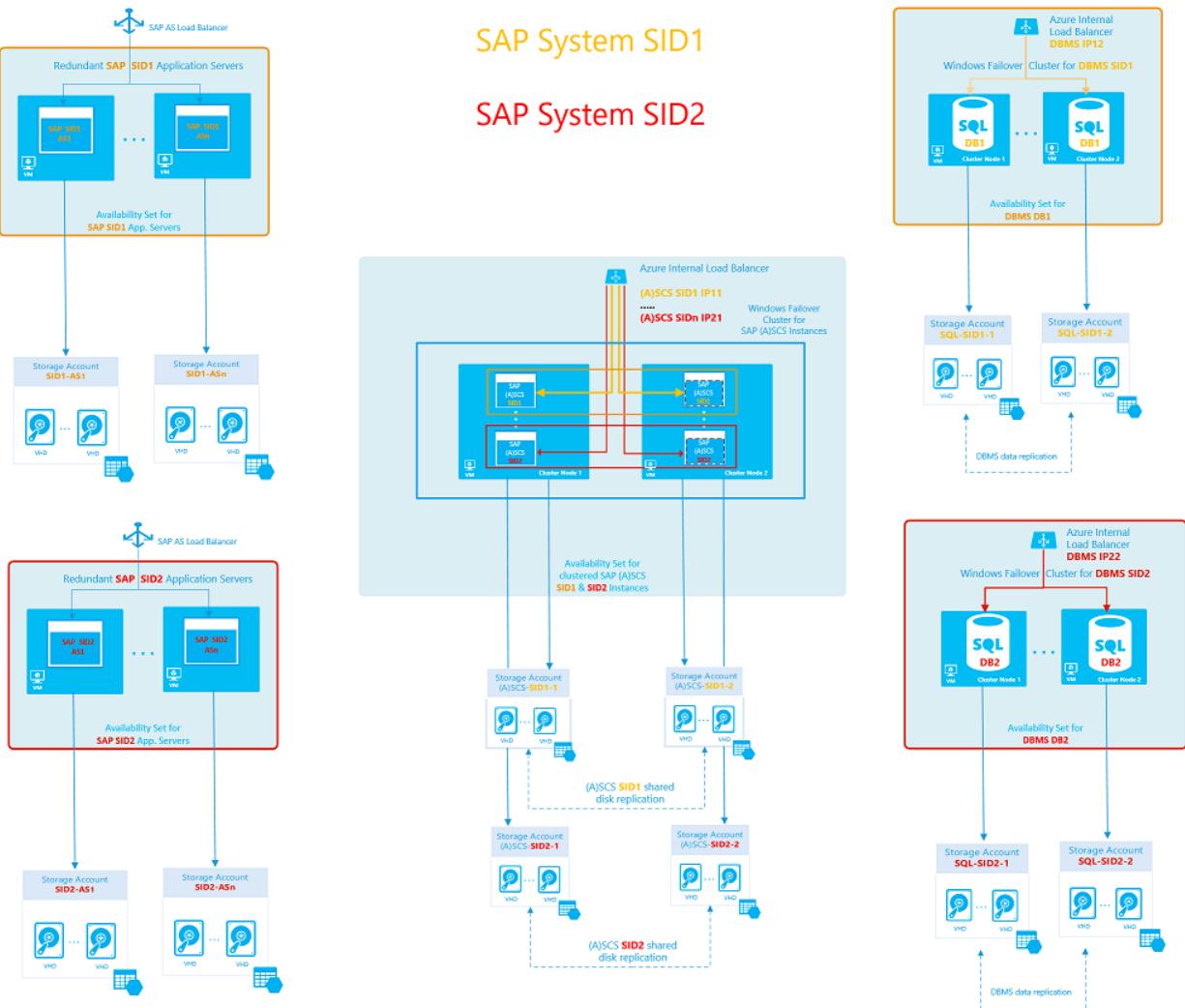


Figure 10: SAP high-availability Architectural Template 3, with a dedicated cluster for different ASCS/SCS instances

Prepare the infrastructure

Prepare the infrastructure for Architectural Template 1

Azure Resource Manager templates for SAP help simplify deployment of required resources.

The three-tier templates in Azure Resource Manager also support high-availability scenarios, such as in Architectural Template 1, which has two clusters. Each cluster is an SAP single point of failure for SAP ASCS/SCS and DBMS.

Here's where you can get Azure Resource Manager templates for the example scenario we describe in this article:

- [Azure Marketplace image](#)
- [Custom image](#)

To prepare the infrastructure for Architectural Template 1:

- In the Azure portal, on the **Parameters** blade, in the **SYSTEMAVAILABILITY** box, select **HA**.

The screenshot shows the 'Parameters' blade of the Azure Resource Manager interface. It displays various parameters for a SAP system deployment:

- SAPSYSTEMID (string)**: PR1
- OSTYPE (string)**: Windows Server 2012 R2 Datacenter
- SAPSYSTEMSIZE (string)**: Large < 180.000 SAPS
- SYSTEMAVAILABILITY (string)**: HA (highlighted with a red border)
- ADMINUSERNAME (string)**: osadmin
- ADMINPASSWORD (securestring)**: (redacted)
- NEWOREXISTINGSUBNET (string)**: existing
- SUBNETID (string)**: /subscriptions/e... (highlighted with a blue border)

Figure 11: Set SAP high-availability Azure Resource Manager parameters

The templates create:

- **Virtual machines:**
 - SAP Application Server virtual machines: <SAPSystemSID>-di-<Number>
 - ASCS/SCS cluster virtual machines: <SAPSystemSID>-asc-s-<Number>
 - DBMS cluster: <SAPSystemSID>-db-<Number>
- **Network cards for all virtual machines, with associated IP addresses:**
 - <SAPSystemSID>-nic-di-<Number>
 - <SAPSystemSID>-nic-asc-s-<Number>
 - <SAPSystemSID>-nic-db-<Number>
- **Azure storage accounts**
- **Availability groups** for:
 - SAP Application Server virtual machines: <SAPSystemSID>-avset-di
 - SAP ASCS/SCS cluster virtual machines: <SAPSystemSID>-avset-asc-s
 - DBMS cluster virtual machines: <SAPSystemSID>-avset-db
- **Azure internal load balancer:**
 - With all ports for the ASCS/SCS instance and IP address <SAPSystemSID>-lb-asc-s
 - With all ports for the SQL Server DBMS and IP address <SAPSystemSID>-lb-db
- **Network security group:** <SAPSystemSID>-nsg-asc-s-0
 - With an open external Remote Desktop Protocol (RDP) port to the <SAPSystemSID>-asc-s-0 virtual

machine

NOTE

All IP addresses of the network cards and Azure internal load balancers are **dynamic** by default. Change them to **static** IP addresses. We describe how to do this later in the article.

Deploy virtual machines with corporate network connectivity (cross-premises) to use in production

For production SAP systems, deploy Azure virtual machines with [corporate network connectivity \(cross-premises\)](#) by using Azure Site-to-Site VPN or Azure ExpressRoute.

NOTE

You can use your Azure Virtual Network instance. The virtual network and subnet have already been created and prepared.

1. In the Azure portal, on the **Parameters** blade, in the **NEWOREXISTINGSUBNET** box, select **existing**.
2. In the **SUBNETID** box, add the full string of your prepared Azure network SubnetID where you plan to deploy your Azure virtual machines.
3. To get a list of all Azure network subnets, run this PowerShell command:

```
(Get-AzureRmVirtualNetwork -Name <azureVnetName> -ResourceGroupName <ResourceGroupOfVNET>).Subnets
```

The **ID** field shows the **SUBNETID**.

4. To get a list of all **SUBNETID** values, run this PowerShell command:

```
(Get-AzureRmVirtualNetwork -Name <azureVnetName> -ResourceGroupName <ResourceGroupOfVNET>).Subnets.Id
```

The **SUBNETID** looks like this:

```
/subscriptions/<SubscriptionId>/resourceGroups/<VPNName>/providers/Microsoft.Network/virtualNetworks/azureVNet/subnets/<SubnetName>
```

Deploy cloud-only SAP instances for test and demo

You can deploy your high-availability SAP system in a cloud-only deployment model. This kind of deployment primarily is useful for demo and test use cases. It's not suited for production use cases.

- In the Azure portal, on the **Parameters** blade, in the **NEWOREXISTINGSUBNET** box, select **new**. Leave the **SUBNETID** field empty.

The SAP Azure Resource Manager template automatically creates the Azure virtual network and subnet.

NOTE

You also need to deploy at least one dedicated virtual machine for Active Directory and DNS in the same Azure Virtual Network instance. The template doesn't create these virtual machines.

Prepare the infrastructure for Architectural Template 2

You can use this Azure Resource Manager template for SAP to help simplify deployment of required infrastructure resources for SAP Architectural Template 2.

Here's where you can get Azure Resource Manager templates for this deployment scenario:

- [Azure Marketplace image](#)
- [Custom image](#)

Prepare the infrastructure for Architectural Template 3

You can prepare the infrastructure and configure SAP for **multi-SID**. For example, you can add an additional SAP ASCS/SCS instance into an *existing* cluster configuration. For more information, see [Configure an additional SAP ASCS/SCS instance into an existing cluster configuration to create an SAP multi-SID configuration in Azure Resource Manager](#).

If you want to create a new multi-SID cluster, you can use the multi-SID [quickstart templates on GitHub](#). To create a new multi-SID cluster, you need to deploy the following three templates:

- [ASCS/SCS template](#)
- [Database template](#)
- [Application servers template](#)

The following sections have more details about the templates and the parameters you need to provide in the templates.

ASCS/SCS template

The ASCS/SCS template deploys two virtual machines that you can use to create a Windows Server failover cluster that hosts multiple ASCS/SCS instances.

To set up the ASCS/SCS multi-SID template, in the [ASCS/SCS multi-SID template](#), enter values for the following parameters:

- **Resource Prefix.** Set the resource prefix, which is used to prefix all resources that are created during the deployment. Because the resources do not belong to only one SAP system, the prefix of the resource is not the SID of one SAP system. The prefix must be between **three and six characters**.
- **Stack Type.** Select the stack type of the SAP system. Depending on the stack type, Azure Load Balancer has one (ABAP or Java only) or two (ABAP+Java) private IP addresses per SAP system.
- **OS Type.** Select the operating system of the virtual machines.
- **SAP System Count.** Select the number of SAP systems you want to install in this cluster.
- **System Availability.** Select **HA**.
- **Admin Username and Admin Password.** Create a new user that can be used to sign in to the machine.
- **New Or Existing Subnet.** Set whether a new virtual network and subnet should be created, or an existing subnet should be used. If you already have a virtual network that is connected to your on-premises network, select **existing**.
- **Subnet Id.** Set the ID of the subnet to which the virtual machines should be connected. Select the subnet of your virtual private network (VPN) or ExpressRoute virtual network to connect the virtual machine to your on-premises network. The ID usually looks like this:

```
/subscriptions/<subscription id>/resourceGroups/<resource group name>/providers/Microsoft.Network/virtualNetworks/<virtual network name>/subnets/<subnet name>
```

The template deploys one Azure Load Balancer instance, which supports multiple SAP systems.

- The ASCS instances are configured for instance number 00, 10, 20...
- The SCS instances are configured for instance number 01, 11, 21...
- The ASCS Enqueue Replication Server (ERS) (Linux only) instances are configured for instance number 02, 12, 22...
- The SCS ERS (Linux only) instances are configured for instance number 03, 13, 23...

The load balancer contains 1 (2 for Linux) VIP(s), 1x VIP for ASCS/SCS and 1x VIP for ERS (Linux only).

The following list contains all load balancing rules (where x is the number of the SAP system, for example, 1, 2, 3...):

- Windows-specific ports for every SAP system: 445, 5985
- ASCS ports (instance number x0): 32x0, 36x0, 39x0, 81x0, 5x013, 5x014, 5x016
- SCS ports (instance number x1): 32x1, 33x1, 39x1, 81x1, 5x113, 5x114, 5x116
- ASCS ERS ports on Linux (instance number x2): 33x2, 5x213, 5x214, 5x216
- SCS ERS ports on Linux (instance number x3): 33x3, 5x313, 5x314, 5x316

The load balancer is configured to use the following probe ports (where x is the number of the SAP system, for example, 1, 2, 3...):

- ASCS/SCS internal load balancer probe port: 620x0
- ERS internal load balancer probe port (Linux only): 621x2

Database template

The database template deploys one or two virtual machines that you can use to install the relational database management system (RDBMS) for one SAP system. For example, if you deploy an ASCS/SCS template for five SAP systems, you need to deploy this template five times.

To set up the database multi-SID template, in the [database multi-SID template](#), enter values for the following parameters:

- **Sap System Id.** Enter the SAP system ID of the SAP system you want to install. The ID will be used as a prefix for the resources that are deployed.
- **Os Type.** Select the operating system of the virtual machines.
- **Dbtype.** Select the type of the database you want to install on the cluster. Select **SQL** if you want to install Microsoft SQL Server. Select **HANA** if you plan to install SAP HANA on the virtual machines. Make sure to select the correct operating system type: select **Windows** for SQL, and select a Linux distribution for HANA. The Azure Load Balancer that is connected to the virtual machines will be configured to support the selected database type:
 - **SQL.** The load balancer will load-balance port 1433. Make sure to use this port for your SQL Server Always On setup.
 - **HANA.** The load balancer will load-balance ports 35015 and 35017. Make sure to install SAP HANA with instance number **50**. The load balancer will use probe port 62550.
- **Sap System Size.** Set the number of SAPS the new system will provide. If you are not sure how many SAPS the system will require, ask your SAP Technology Partner or System Integrator.
- **System Availability.** Select **HA**.
- **Admin Username and Admin Password.** Create a new user that can be used to sign in to the machine.
- **Subnet Id.** Enter the ID of the subnet that you used during the deployment of the ASCS/SCS template, or the ID of the subnet that was created as part of the ASCS/SCS template deployment.

Application servers template

The application servers template deploys two or more virtual machines that can be used as SAP Application Server instances for one SAP system. For example, if you deploy an ASCS/SCS template for five SAP systems, you need to deploy this template five times.

To set up the application servers multi-SID template, in the [application servers multi-SID template](#), enter values for the following parameters:

- **Sap System Id.** Enter the SAP system ID of the SAP system you want to install. The ID will be used as a prefix for the resources that are deployed.
- **Os Type.** Select the operating system of the virtual machines.
- **Sap System Size.** The number of SAPS the new system will provide. If you are not sure how many SAPS the system will require, ask your SAP Technology Partner or System Integrator.

- **System Availability.** Select **HA**.
- **Admin Username and Admin Password.** Create a new user that can be used to sign in to the machine.
- **Subnet Id.** Enter the ID of the subnet that you used during the deployment of the ASCS/SCS template, or the ID of the subnet that was created as part of the ASCS/SCS template deployment.

Azure virtual network

In our example, the address space of the Azure virtual network is 10.0.0.0/16. There is one subnet called **Subnet**, with an address range of 10.0.0.0/24. All virtual machines and internal load balancers are deployed in this virtual network.

IMPORTANT

Don't make any changes to the network settings inside the guest operating system. This includes IP addresses, DNS servers, and subnet. Configure all your network settings in Azure. The Dynamic Host Configuration Protocol (DHCP) service propagates your settings.

DNS IP addresses

To set the required DNS IP addresses, do the following steps.

1. In the Azure portal, on the **DNS servers** blade, make sure that your virtual network **DNS servers** option is set to **Custom DNS**.
2. Select your settings based on the type of network you have. For more information, see the following resources:
 - [Corporate network connectivity \(cross-premises\)](#): Add the IP addresses of the on-premises DNS servers. You can extend on-premises DNS servers to the virtual machines that are running in Azure. In that scenario, you can add the IP addresses of the Azure virtual machines on which you run the DNS service.
 - [Cloud-only deployment](#): Deploy an additional virtual machine in the same Virtual Network instance that serves as a DNS server. Add the IP addresses of the Azure virtual machines that you've set up to run DNS service.

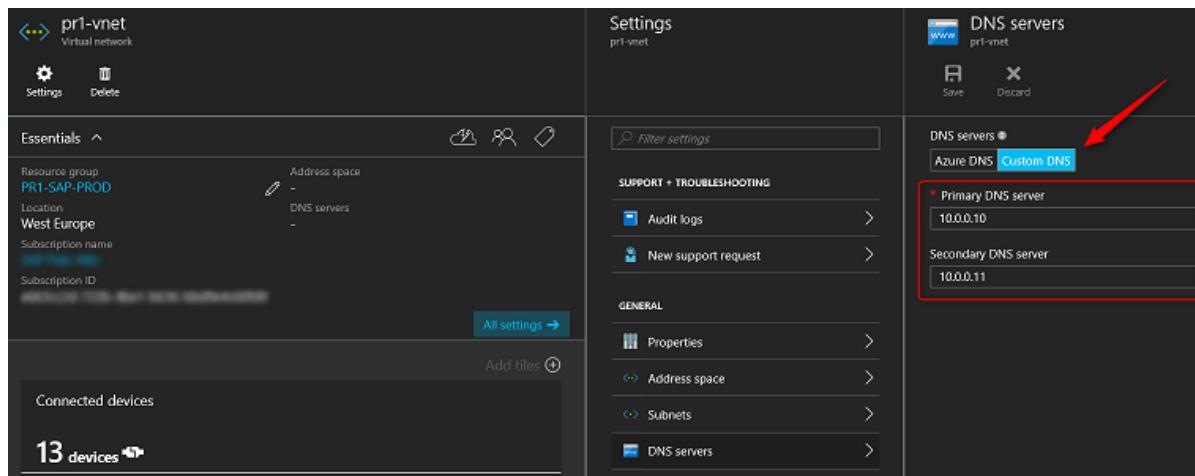


Figure 12: Configure DNS servers for Azure Virtual Network

NOTE

If you change the IP addresses of the DNS servers, you need to restart the Azure virtual machines to apply the change and propagate the new DNS servers.

In our example, the DNS service is installed and configured on these Windows virtual machines:

| VIRTUAL MACHINE ROLE | VIRTUAL MACHINE HOST NAME | NETWORK CARD NAME | STATIC IP ADDRESS |
|----------------------|---------------------------|--------------------|-------------------|
| First DNS server | domcontr-0 | pr1-nic-domcontr-0 | 10.0.0.10 |
| Second DNS server | domcontr-1 | pr1-nic-domcontr-1 | 10.0.0.11 |

Host names and static IP addresses for the SAP ASCS/SCS clustered instance and DBMS clustered instance

For on-premises deployment, you need these reserved host names and IP addresses:

| VIRTUAL HOST NAME ROLE | VIRTUAL HOST NAME | VIRTUAL STATIC IP ADDRESS |
|---|-------------------|---------------------------|
| SAP ASCS/SCS first cluster virtual host name (for cluster management) | pr1-ascs-vir | 10.0.0.42 |
| SAP ASCS/SCS instance virtual host name | pr1-ascs-sap | 10.0.0.43 |
| SAP DBMS second cluster virtual host name (cluster management) | pr1-dbms-vir | 10.0.0.32 |

When you create the cluster, create the virtual host names **pr1-ascs-vir** and **pr1-dbms-vir** and the associated IP addresses that manage the cluster itself. For information about how to do this, see [Collect cluster nodes in a cluster configuration](#).

You can manually create the other two virtual host names, **pr1-ascs-sap** and **pr1-dbms-sap**, and the associated IP addresses, on the DNS server. The clustered SAP ASCS/SCS instance and the clustered DBMS instance use these resources. For information about how to do this, see [Create a virtual host name for a clustered SAP ASCS/SCS instance](#).

Set static IP addresses for the SAP virtual machines

After you deploy the virtual machines to use in your cluster, you need to set static IP addresses for all virtual machines. Do this in the Azure Virtual Network configuration, and not in the guest operating system.

1. In the Azure portal, select **Resource Group > Network Card > Settings > IP Address**.
2. On the **IP addresses** blade, under **Assignment**, select **Static**. In the **IP address** box, enter the IP address that you want to use.

NOTE

If you change the IP address of the network card, you need to restart the Azure virtual machines to apply the change.

Figure 13: Set static IP addresses for the network card of each virtual machine

Repeat this step for all network interfaces, that is, for all virtual machines, including virtual machines that you want to use for your Active Directory/DNS service.

In our example, we have these virtual machines and static IP addresses:

| VIRTUAL MACHINE ROLE | VIRTUAL MACHINE HOST NAME | NETWORK CARD NAME | STATIC IP ADDRESS |
|---|---------------------------|-------------------|-------------------|
| First SAP Application Server instance | pr1-di-0 | pr1-nic-di-0 | 10.0.0.50 |
| Second SAP Application Server instance | pr1-di-1 | pr1-nic-di-1 | 10.0.0.51 |
| ... | ... | ... | ... |
| Last SAP Application Server instance | pr1-di-5 | pr1-nic-di-5 | 10.0.0.55 |
| First cluster node for ASCS/SCS instance | pr1-ascscs-0 | pr1-nic-ascscs-0 | 10.0.0.40 |
| Second cluster node for ASCS/SCS instance | pr1-ascscs-1 | pr1-nic-ascscs-1 | 10.0.0.41 |
| First cluster node for DBMS instance | pr1-db-0 | pr1-nic-db-0 | 10.0.0.30 |
| Second cluster node for DBMS instance | pr1-db-1 | pr1-nic-db-1 | 10.0.0.31 |

Set a static IP address for the Azure internal load balancer

The SAP Azure Resource Manager template creates an Azure internal load balancer that is used for the SAP ASCS/SCS instance cluster and the DBMS cluster.

IMPORTANT

The IP address of the virtual host name of the SAP ASCS/SCS is the same as the IP address of the SAP ASCS/SCS internal load balancer: **pr1-lb-asc**s. The IP address of the virtual name of the DBMS is the same as the IP address of the DBMS internal load balancer: **pr1-lb-dbms**.

To set a static IP address for the Azure internal load balancer:

1. The initial deployment sets the internal load balancer IP address to **Dynamic**. In the Azure portal, on the **IP addresses** blade, under **Assignment**, select **Static**.
2. Set the IP address of the internal load balancer **pr1-lb-asc**s to the IP address of the virtual host name of the SAP ASCS/SCS instance.
3. Set the IP address of the internal load balancer **pr1-lb-dbms** to the IP address of the virtual host name of the DBMS instance.

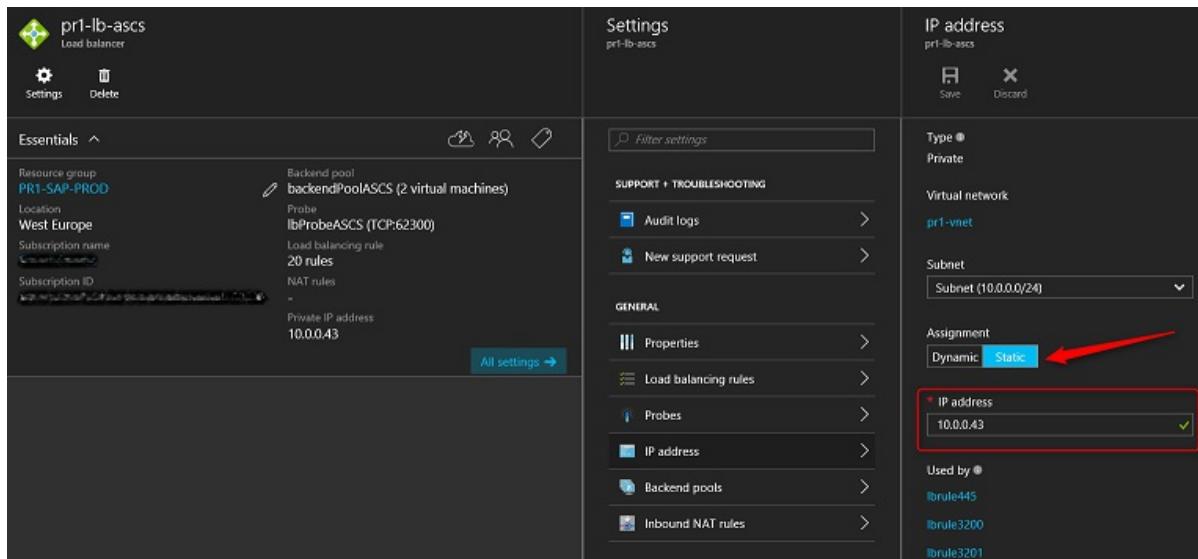


Figure 14: Set static IP addresses for the internal load balancer for the SAP ASCS/SCS instance

In our example, we have two Azure internal load balancers that have these static IP addresses:

| AZURE INTERNAL LOAD BALANCER ROLE | AZURE INTERNAL LOAD BALANCER NAME | STATIC IP ADDRESS |
|--|-----------------------------------|-------------------|
| SAP ASCS/SCS instance internal load balancer | pr1-lb-asc | 10.0.0.43 |
| SAP DBMS internal load balancer | pr1-lb-dbms | 10.0.0.33 |

Default ASCS/SCS load balancing rules for the Azure internal load balancer

The SAP Azure Resource Manager template creates the ports you need:

- An ABAP ASCS instance, with the default instance number **00**
- A Java SCS instance, with the default instance number **01**

When you install your SAP ASCS/SCS instance, you must use the default instance number **00** for your ABAP ASCS instance and the default instance number **01** for your Java SCS instance.

Next, create required internal load balancing endpoints for the SAP NetWeaver ports.

To create required internal load balancing endpoints, first, create these load balancing endpoints for the SAP NetWeaver ABAP ASCS ports:

| SERVICE/LOAD BALANCING RULE NAME | DEFAULT PORT NUMBERS | CONCRETE PORTS FOR (ASCS INSTANCE WITH INSTANCE NUMBER 00) (ERS WITH 10) |
|--|------------------------------|--|
| Enqueue Server / <i>lrule3200</i> | 32< <i>InstanceNumber</i> > | 3200 |
| ABAP Message Server / <i>lrule3600</i> | 36< <i>InstanceNumber</i> > | 3600 |
| Internal ABAP Message / <i>lrule3900</i> | 39< <i>InstanceNumber</i> > | 3900 |
| Message Server HTTP / <i>Lrule8100</i> | 81< <i>InstanceNumber</i> > | 8100 |
| SAP Start Service ASCS HTTP / <i>Lrule50013</i> | 5< <i>InstanceNumber</i> >13 | 50013 |
| SAP Start Service ASCS HTTPS / <i>Lrule50014</i> | 5< <i>InstanceNumber</i> >14 | 50014 |
| Enqueue Replication / <i>Lrule50016</i> | 5< <i>InstanceNumber</i> >16 | 50016 |
| SAP Start Service ERS HTTP <i>Lrule51013</i> | 5< <i>InstanceNumber</i> >13 | 51013 |
| SAP Start Service ERS HTTP <i>Lrule51014</i> | 5< <i>InstanceNumber</i> >14 | 51014 |
| Win RM <i>Lrule5985</i> | | 5985 |
| File Share <i>Lrule445</i> | | 445 |

Table 1: Port numbers of the SAP NetWeaver ABAP ASCS instances

Then, create these load balancing endpoints for the SAP NetWeaver Java SCS ports:

| SERVICE/LOAD BALANCING RULE NAME | DEFAULT PORT NUMBERS | CONCRETE PORTS FOR (SCS INSTANCE WITH INSTANCE NUMBER 01) (ERS WITH 11) |
|---|------------------------------|---|
| Enqueue Server / <i>lrule3201</i> | 32< <i>InstanceNumber</i> > | 3201 |
| Gateway Server / <i>lrule3301</i> | 33< <i>InstanceNumber</i> > | 3301 |
| Java Message Server / <i>lrule3900</i> | 39< <i>InstanceNumber</i> > | 3901 |
| Message Server HTTP / <i>Lrule8101</i> | 81< <i>InstanceNumber</i> > | 8101 |
| SAP Start Service SCS HTTP / <i>Lrule50113</i> | 5< <i>InstanceNumber</i> >13 | 50113 |
| SAP Start Service SCS HTTPS / <i>Lrule50114</i> | 5< <i>InstanceNumber</i> >14 | 50114 |
| Enqueue Replication / <i>Lrule50116</i> | 5< <i>InstanceNumber</i> >16 | 50116 |

| SERVICE/LOAD BALANCING RULE NAME | DEFAULT PORT NUMBERS | CONCRETE PORTS FOR (SCS INSTANCE WITH INSTANCE NUMBER 01) (ERS WITH 11) |
|---|----------------------|---|
| SAP Start Service ERS HTTP
<i>Lrule51113</i> | 5<InstanceNumber>13 | 51113 |
| SAP Start Service ERS HTTP
<i>Lrule51114</i> | 5<InstanceNumber>14 | 51114 |
| Win RM <i>Lrule5985</i> | | 5985 |
| File Share <i>Lrule445</i> | | 445 |

Table 2: Port numbers of the SAP NetWeaver Java SCS instances

| NAME | LOAD BALANCER | BACKEND POOL | PROBE |
|------------|-----------------------|-----------------|-------------|
| lrule3200 | lrule3200 (TCP/32...) | backendPoolASCS | lbProbeASCS |
| lrule3201 | lrule3201 (TCP/32...) | backendPoolASCS | lbProbeASCS |
| lrule3301 | lrule3301 (TCP/33...) | backendPoolASCS | lbProbeASCS |
| lrule3600 | lrule3600 (TCP/36...) | backendPoolASCS | lbProbeASCS |
| lrule3900 | lrule3900 (TCP/39...) | backendPoolASCS | lbProbeASCS |
| lrule3901 | lrule3901 (TCP/39...) | backendPoolASCS | lbProbeASCS |
| lrule445 | lrule445 (TCP/445) | backendPoolASCS | lbProbeASCS |
| lrule50013 | lrule50013 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule50014 | lrule50014 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule50016 | lrule50016 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule50113 | lrule50113 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule50114 | lrule50114 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule50116 | lrule50116 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule51013 | lrule51013 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule51014 | lrule51014 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule51113 | lrule51113 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule51114 | lrule51114 (TCP/5...) | backendPoolASCS | lbProbeASCS |
| lrule5985 | lrule5985 (TCP/59...) | backendPoolASCS | lbProbeASCS |
| lrule8100 | lrule8100 (TCP/81...) | backendPoolASCS | lbProbeASCS |
| lrule8101 | lrule8101 (TCP/81...) | backendPoolASCS | lbProbeASCS |

Figure 15: Default ASCS/SCS load balancing rules for the Azure internal load balancer

Set the IP address of the load balancer **pr1-lb-dbms** to the IP address of the virtual host name of the DBMS instance.

Change the ASCS/SCS default load balancing rules for the Azure internal load balancer

If you want to use different numbers for the SAP ASCS or SCS instances, you must change the names and values of their ports from default values.

1. In the Azure portal, select **<SID>-lb-ascs load balancer > Load Balancing Rules**.
2. For all load balancing rules that belong to the SAP ASCS or SCS instance, change these values:

- Name
- Port
- Back-end port

For example, if you want to change the default ASCS instance number from 00 to 31, you need to make the changes for all ports listed in Table 1.

Here's an example of an update for port *lbrule3200*.

The screenshot shows the Azure portal interface for managing load balancing rules. On the left, a list of existing rules is shown, including lbrule3200, lbrule3201, lbrule3301, lbrule3600, lbrule3900, lbrule3901, lbrule445, lbrule50013, lbrule50014, lbrule50016, lbrule50113, lbrule50114, lbrule50116, lbrule51013, lbrule51014, lbrule51113, and lbrule51114. On the right, a detailed view of the lbrule3200 rule is displayed. The rule is configured for TCP port 3200, using the backend pool 'backendPoolASCS' (which contains 2 virtual machines). The probe is set to 'lbProbeASCS'. Session persistence is disabled, and the idle timeout is set to 30 minutes.

Figure 16: Change the ASCS/SCS default load balancing rules for the Azure internal load balancer

Add Windows virtual machines to the domain

After you assign a static IP address to the virtual machines, add the virtual machines to the domain.

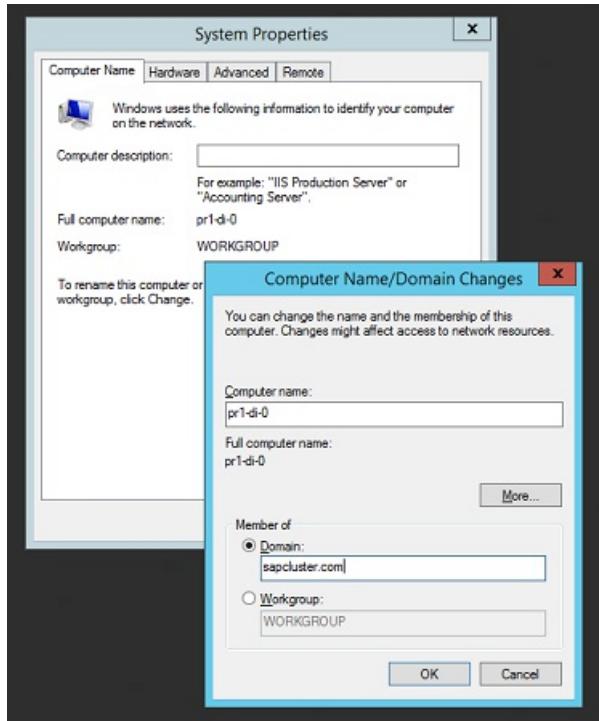


Figure 17: Add a virtual machine to a domain

Add registry entries on both cluster nodes of the SAP ASCS/SCS instance

Azure Load Balancer has an internal load balancer that closes connections when the connections are idle for a set period of time (an idle timeout). SAP work processes in dialog instances open connections to the SAP enqueue process as soon as the first enqueue/dequeue request needs to be sent. These connections usually remain

established until the work process or the enqueue process restarts. However, if the connection is idle for a set period of time, the Azure internal load balancer closes the connections. This isn't a problem because the SAP work process reestablishes the connection to the enqueue process if it no longer exists. These activities are documented in the developer traces of SAP processes, but they create a large amount of extra content in those traces. It's a good idea to change the TCP/IP `KeepAliveTime` and `KeepAliveInterval` on both cluster nodes. Combine these changes in the TCP/IP parameters with SAP profile parameters, described later in the article.

To add registry entries on both cluster nodes of the SAP ASCS/SCS instance, first, add these Windows registry entries on both Windows cluster nodes for SAP ASCS/SCS:

| PATH | HKLM\SYSTEM\CURRENTCONTROLSET\SERVICES\TCPPIP\PARAMETERS |
|-----------------------|---|
| Variable name | <code>KeepAliveTime</code> |
| Variable type | REG_DWORD (Decimal) |
| Value | 120000 |
| Link to documentation | https://technet.microsoft.com/en-us/library/cc957549.aspx |

Table 3: Change the first TCP/IP parameter

Then, add this Windows registry entries on both Windows cluster nodes for SAP ASCS/SCS:

| PATH | HKLM\SYSTEM\CURRENTCONTROLSET\SERVICES\TCPPIP\PARAMETERS |
|-----------------------|---|
| Variable name | <code>KeepAliveInterval</code> |
| Variable type | REG_DWORD (Decimal) |
| Value | 120000 |
| Link to documentation | https://technet.microsoft.com/en-us/library/cc957548.aspx |

Table 4: Change the second TCP/IP parameter

To apply the changes, restart both cluster nodes.

Set up a Windows Server Failover Clustering cluster for an SAP ASCS/SCS instance

Setting up a Windows Server Failover Clustering cluster for an SAP ASCS/SCS instance involves these tasks:

- Collecting the cluster nodes in a cluster configuration
- Configuring a cluster file share witness

Collect the cluster nodes in a cluster configuration

1. In the Add Role and Features Wizard, add failover clustering to both cluster nodes.
2. Set up the failover cluster by using Failover Cluster Manager. In Failover Cluster Manager, select **Create Cluster**, and then add only the name of the first cluster, node A. Do not add the second node yet; you'll add the second node in a later step.

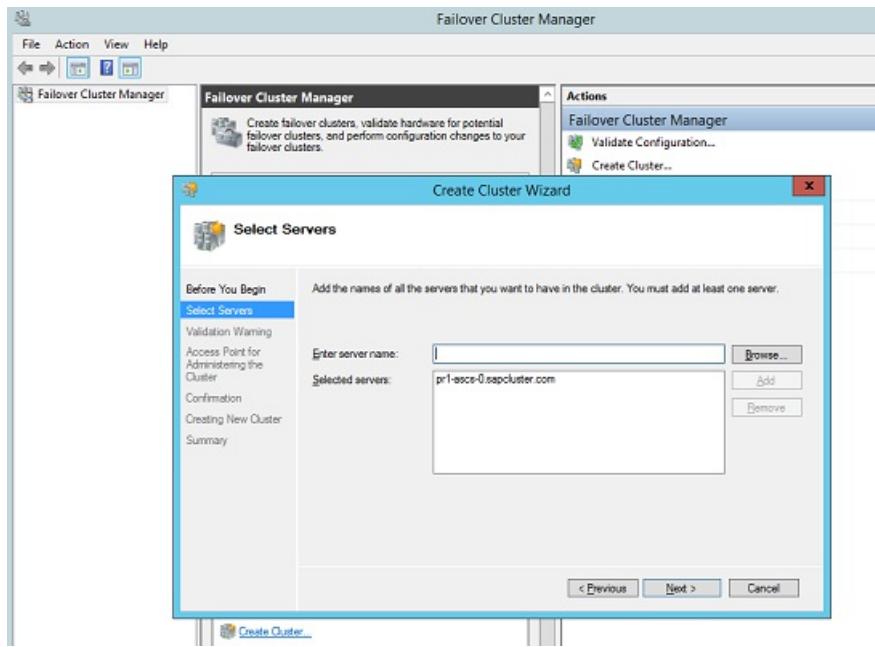


Figure 18: Add the server or virtual machine name of the first cluster node

3. Enter the network name (virtual host name) of the cluster.

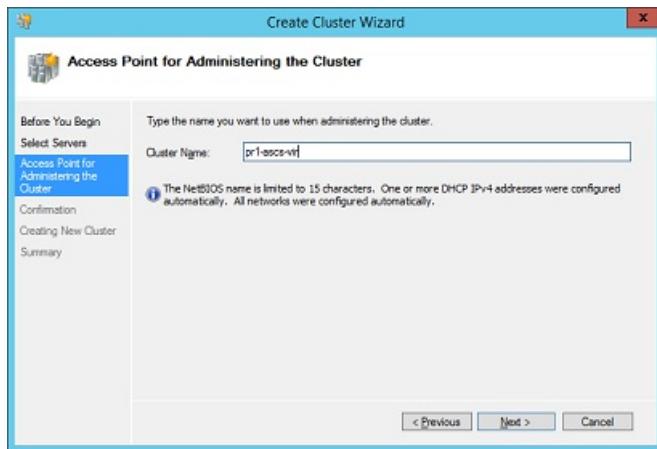


Figure 19: Enter the cluster name

4. After you've created the cluster, run a cluster validation test.

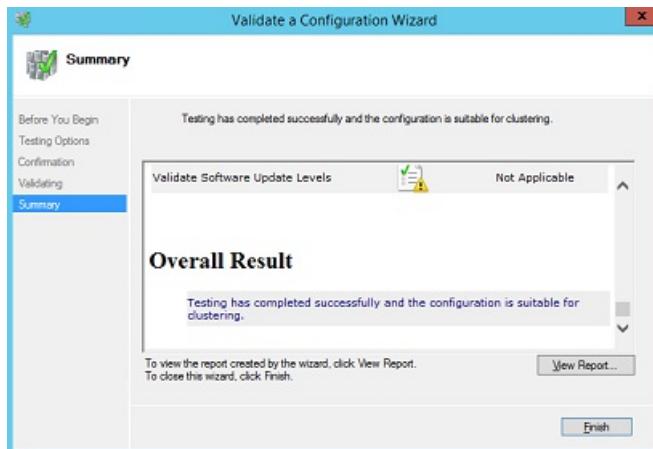


Figure 20: Run the cluster validation check

You can ignore any warnings about disks at this point in the process. You'll add a file share witness and the SIOS shared disks later. At this stage, you don't need to worry about having a quorum.

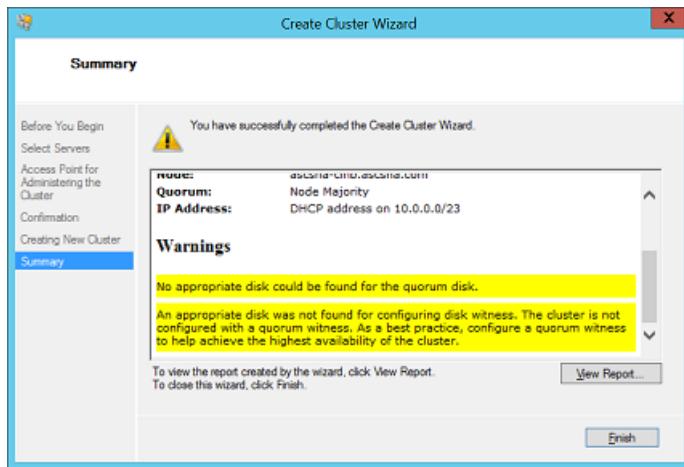


Figure 21: No quorum disk is found

This screenshot shows the 'Cluster Core Resources' management interface. On the left, there's a navigation pane with icons for Roles, Networks, and Nodes. Under 'Cluster Core Resources', a table lists resources. One resource, 'pr1-ascs-vir', is shown with its 'Name' and 'IP Address' (10.0.0.40). The status is listed as 'Offline' and 'Failed'. To the right of the table, there are 'Status' and 'Information' columns.

| Name | Status | Information |
|-----------------------|---------|-------------|
| Server Name | | |
| pr1-ascs-vir | Offline | Failed |
| IP Address: 10.0.0.40 | | |

Figure 22: Core cluster resource needs a new IP address

5. Change the IP address of the core cluster service. The cluster can't start until you change the IP address of the core cluster service, because the IP address of the server points to one of the virtual machine nodes. Do this on the **Properties** page of the core cluster service's IP resource.

For example, we need to assign an IP address (in our example, **10.0.0.42**) for the cluster virtual host name **pr1-ascs-vir**.

This screenshot shows the 'Properties' dialog box for the 'pr1-ascs-vir' resource. The left pane lists 'Roles', 'Networks', and 'Nodes'. The 'Nodes' pane is expanded, showing actions like 'Bring Online', 'Take Offline', 'Information Details...', 'Show Critical Events', 'More Actions', 'Remove', and 'Properties'. The 'Properties' button is selected. The right pane shows 'Status' and 'Information' columns. The 'IP Address' field is currently set to '10.0.0.40'. A dropdown menu is open over this field, with the option '10.0.0.42' highlighted.

Figure 23: In the **Properties** dialog box, change the IP address

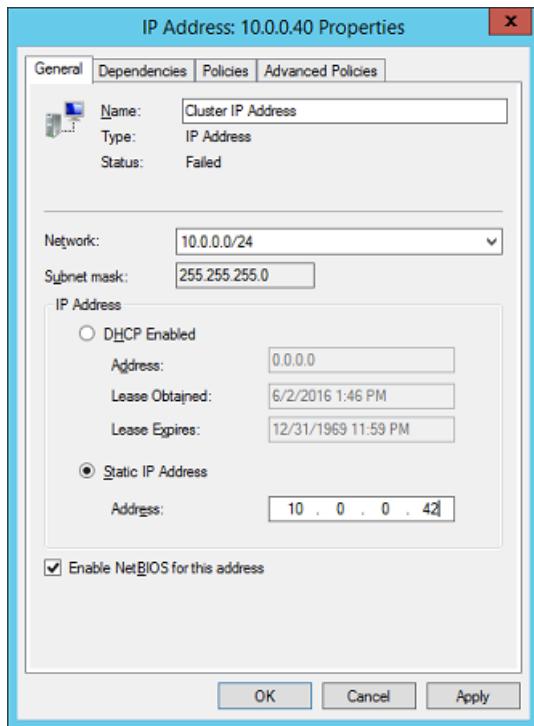


Figure 24: Assign the IP address that is reserved for the cluster

6. Bring the cluster virtual host name online.

| Name | Status | Information |
|-----------------------|--------|-------------|
| Server Name | | |
| Name: pr1-ascs-vir | Online | |
| IP Address: 10.0.0.42 | Online | |

Figure 25: Cluster core service is up and running, and with the correct IP address

7. Add the second cluster node.

Now that the core cluster service is up and running, you can add the second cluster node.

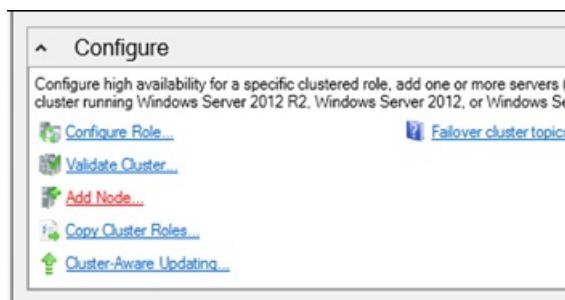


Figure 26: Add the second cluster node

8. Enter a name for the second cluster node host.

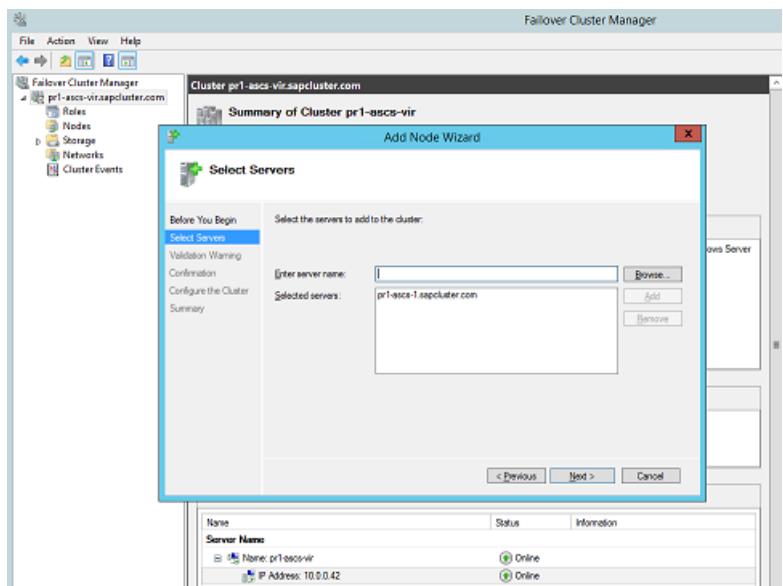


Figure 27: Enter the second cluster node host name

IMPORTANT

Be sure that the **Add all eligible storage to the cluster** check box is **NOT** selected.

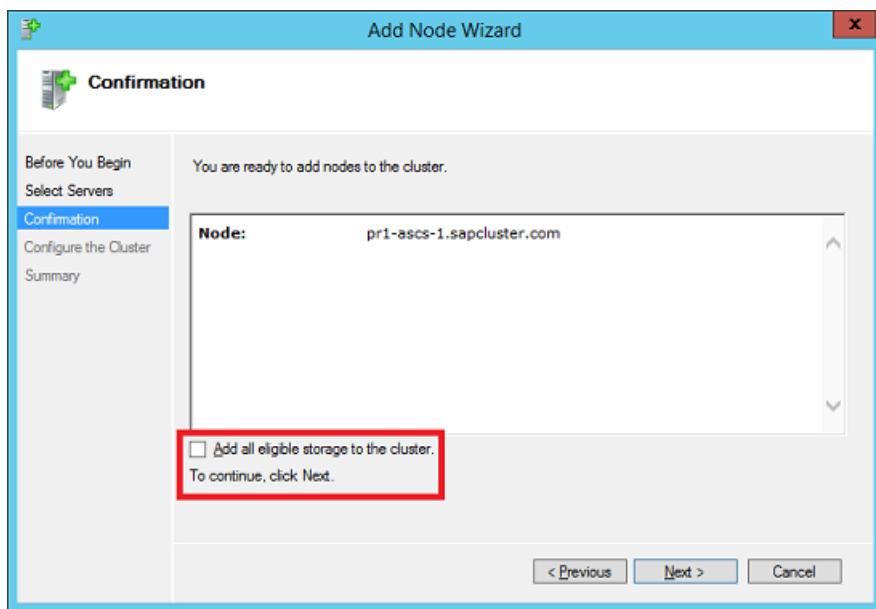


Figure 28: Do not select the check box

You can ignore warnings about quorum and disks. You'll set the quorum and share the disk later, as described in [Installing SIOS DataKeeper Cluster Edition for SAP ASCS/SCS cluster share disk](#).

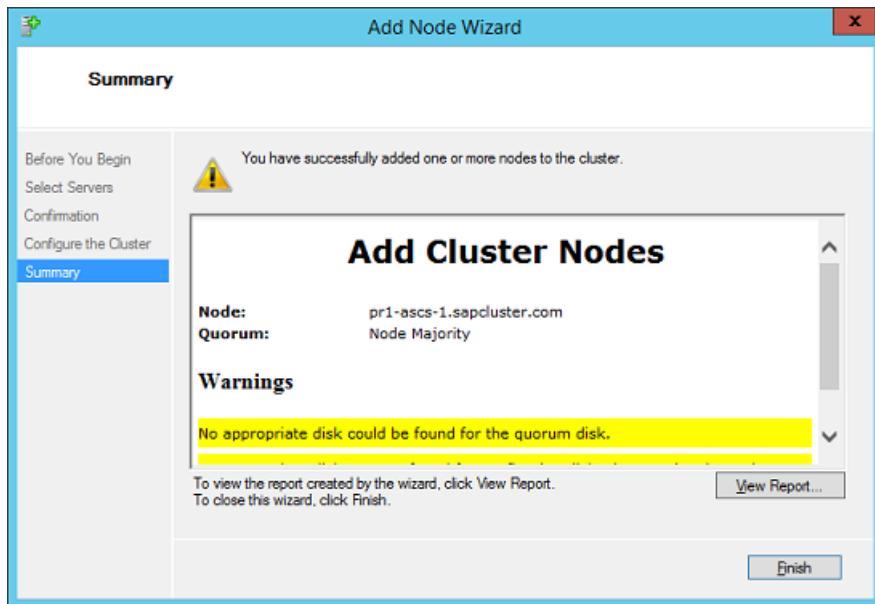


Figure 29: Ignore warnings about the disk quorum

Configure a cluster file share witness

Configuring a cluster file share witness involves these tasks:

- Creating a file share
- Setting the file share witness quorum in Failover Cluster Manager

Create a file share

1. Select a file share witness instead of a quorum disk. SIOS DataKeeper supports this option.

In the examples in this article, the file share witness is on the Active Directory/DNS server that is running in Azure. The file share witness is called **domcontr-0**. Because you would have configured a VPN connection to Azure (via Site-to-Site VPN or Azure ExpressRoute), your Active Directory/DNS service is on-premises and isn't suitable to run a file share witness.

NOTE

If your Active Directory/DNS service runs only on-premises, don't configure your file share witness on the Active Directory/DNS Windows operating system that is running on-premises. Network latency between cluster nodes running in Azure and Active Directory/DNS on-premises might be too large and cause connectivity issues. Be sure to configure the file share witness on an Azure virtual machine that is running close to the cluster node.

The quorum drive needs at least 1,024 MB of free space. We recommend 2,048 MB of free space for the quorum drive.

2. Add the cluster name object.

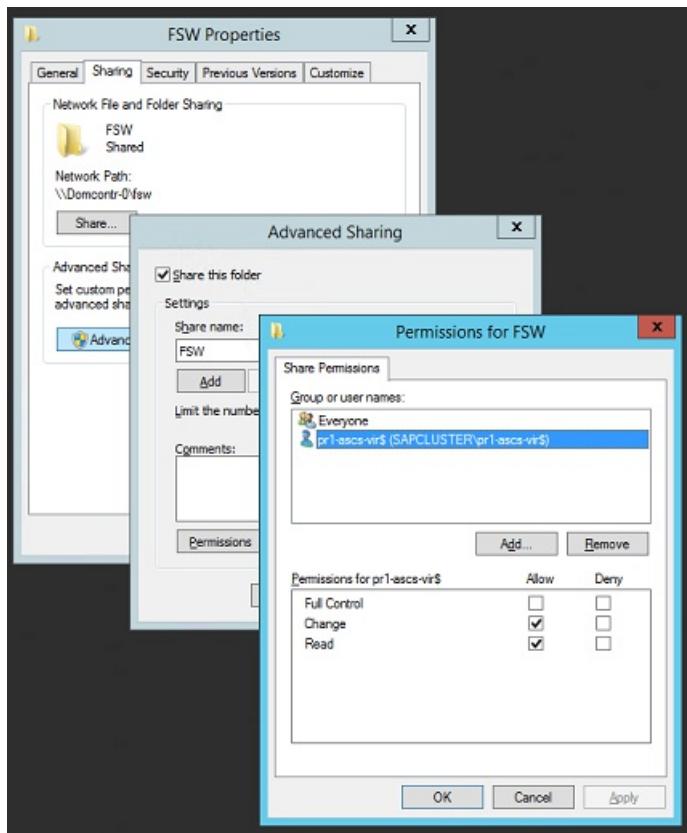


Figure 30: Assign the permissions on the share for the cluster name object

Be sure that the permissions include the authority to change data in the share for the cluster name object (in our example, **pr1-ascs-vir\$**).

3. To add the cluster name object to the list, select **Add**. Change the filter to check for computer objects, in addition to those shown in Figure 31.

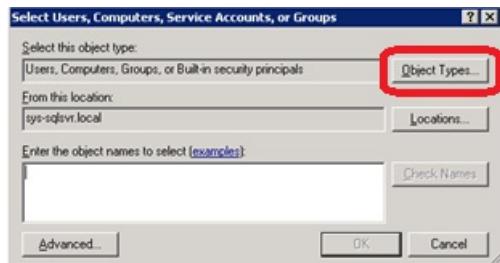


Figure 31: Change the Object Types to include computers

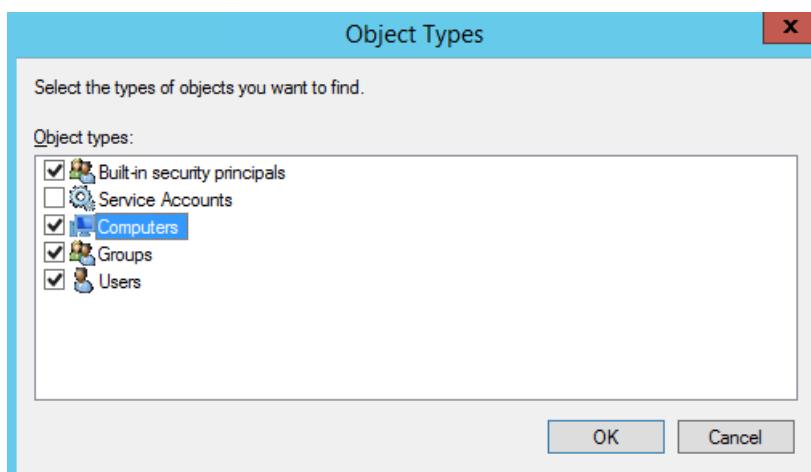


Figure 32: Select the **Computers** check box

4. Enter the cluster name object as shown in Figure 31. Because the record has already been created, you can

change the permissions, as shown in Figure 30.

5. Select the **Security** tab of the share, and then set more detailed permissions for the cluster name object.

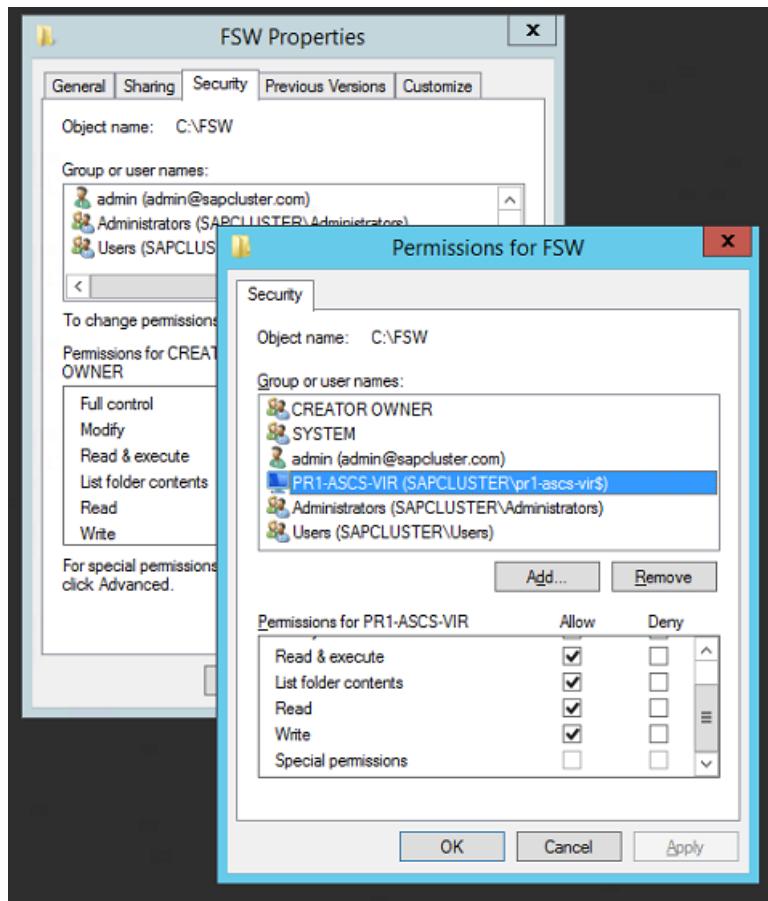


Figure 33: Set the security attributes for the cluster name object on the file share quorum

Set the file share witness quorum in Failover Cluster Manager

1. Open the Configure Quorum Setting Wizard.

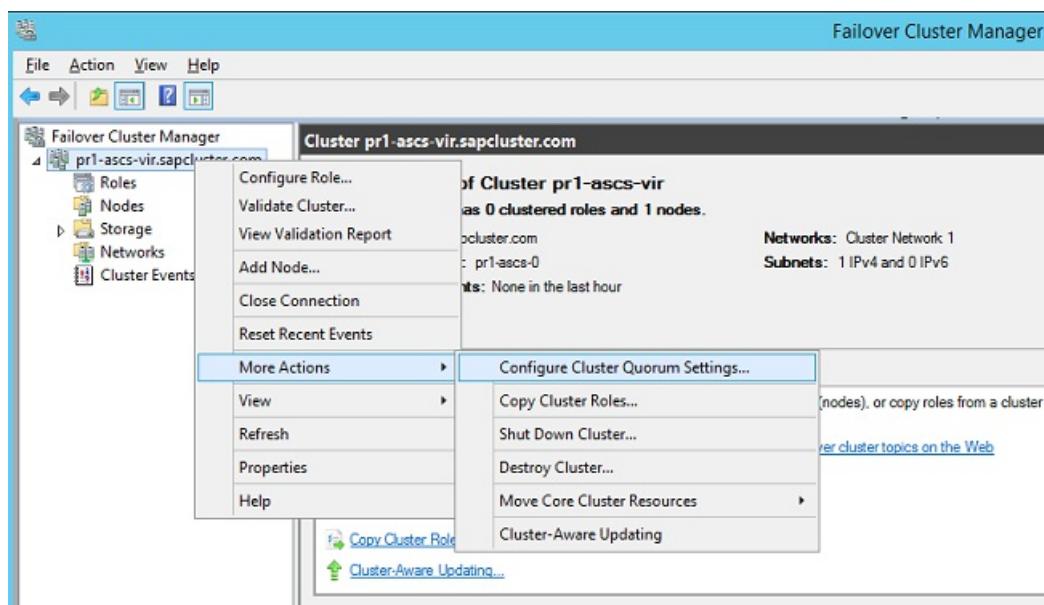


Figure 34: Start the Configure Cluster Quorum Setting Wizard

2. On the **Select Quorum Configuration** page, select **Select the quorum witness**.

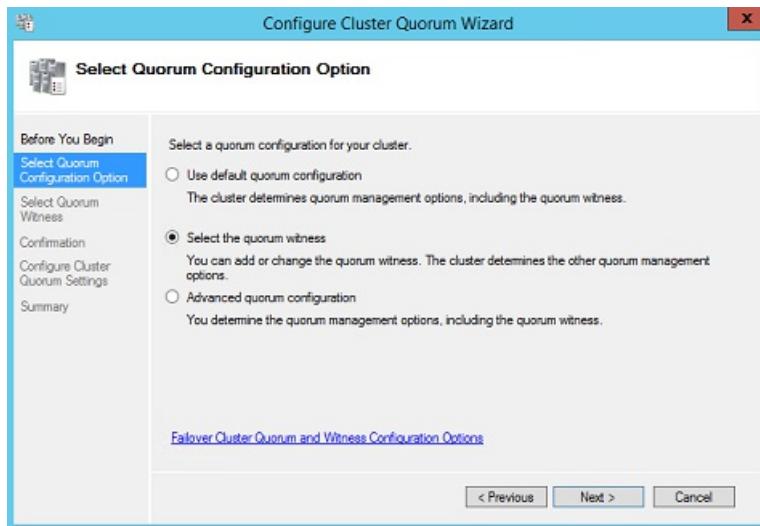


Figure 35: Quorum configurations you can choose from

3. On the **Select Quorum Witness** page, select **Configure a file share witness**.

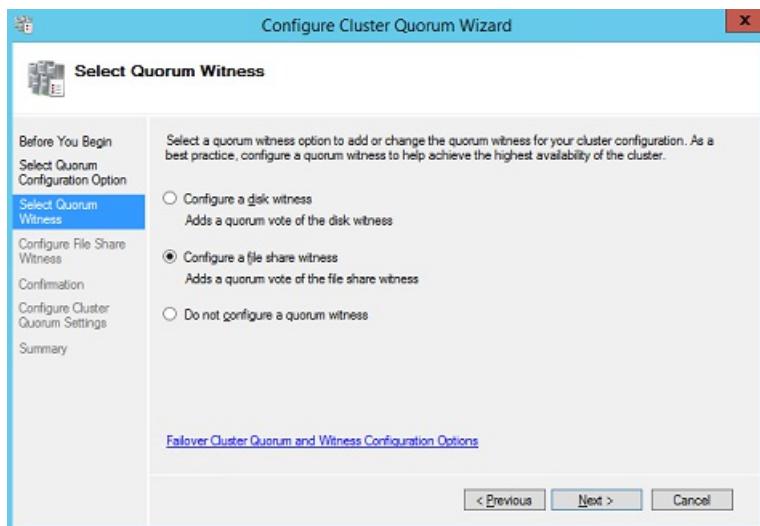


Figure 36: Select the file share witness

4. Enter the UNC path to the file share (in our example, \domcontr-0\FSW). To see a list of the changes you can make, select **Next**.

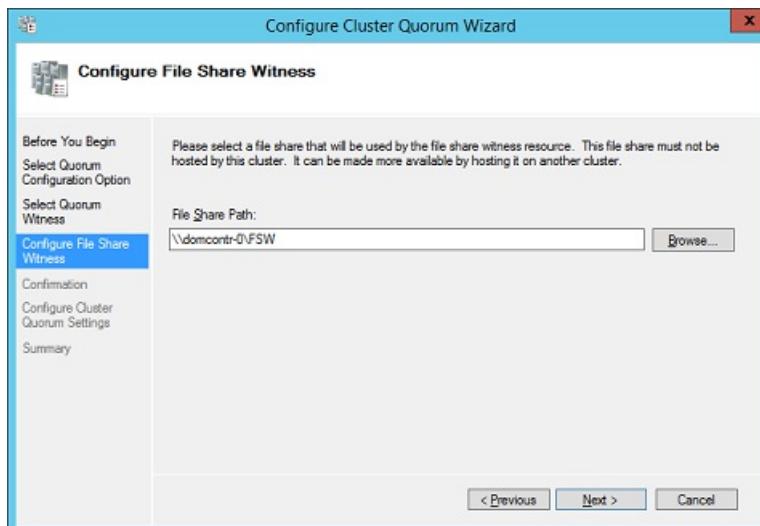


Figure 37: Define the file share location for the witness share

5. Select the changes you want, and then select **Next**. You need to successfully reconfigure the cluster

configuration as shown in Figure 38.

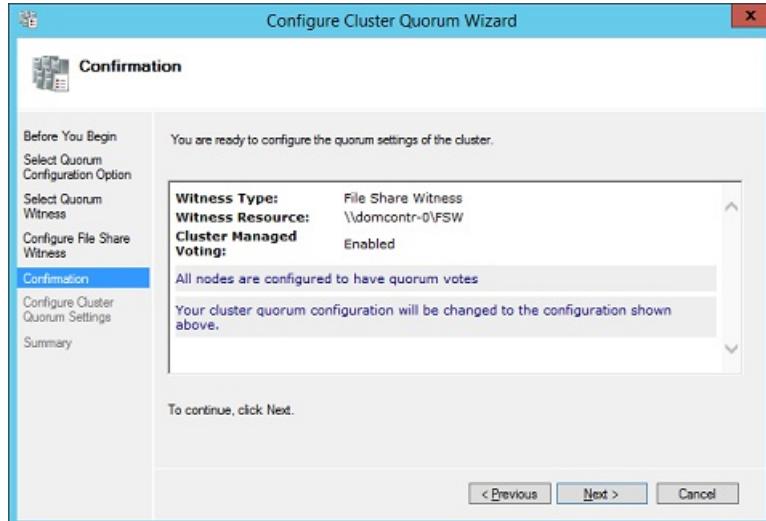


Figure 38: Confirmation that you've reconfigured the cluster

Install SIOS DataKeeper Cluster Edition for the SAP ASCS/SCS cluster share disk

You now have a working Windows Server Failover Clustering configuration in Azure. But, to install an SAP ASCS/SCS instance, you need a shared disk resource. You cannot create the shared disk resources you need in Azure. SIOS DataKeeper Cluster Edition is a third-party solution you can use to create shared disk resources.

Installing SIOS DataKeeper Cluster Edition for the SAP ASCS/SCS cluster share disk involves these tasks:

- Adding the .NET Framework 3.5
- Installing SIOS DataKeeper
- Setting up SIOS DataKeeper

Add the .NET Framework 3.5

The Microsoft .NET Framework 3.5 isn't automatically activated or installed on Windows Server 2012 R2. Because SIOS DataKeeper requires the .NET Framework to be on all nodes that you install DataKeeper on, you must install the .NET Framework 3.5 on the guest operating system of all virtual machines in the cluster.

There are two ways to add the .NET Framework 3.5:

- Use the Add Roles and Features Wizard in Windows as shown in Figure 39.

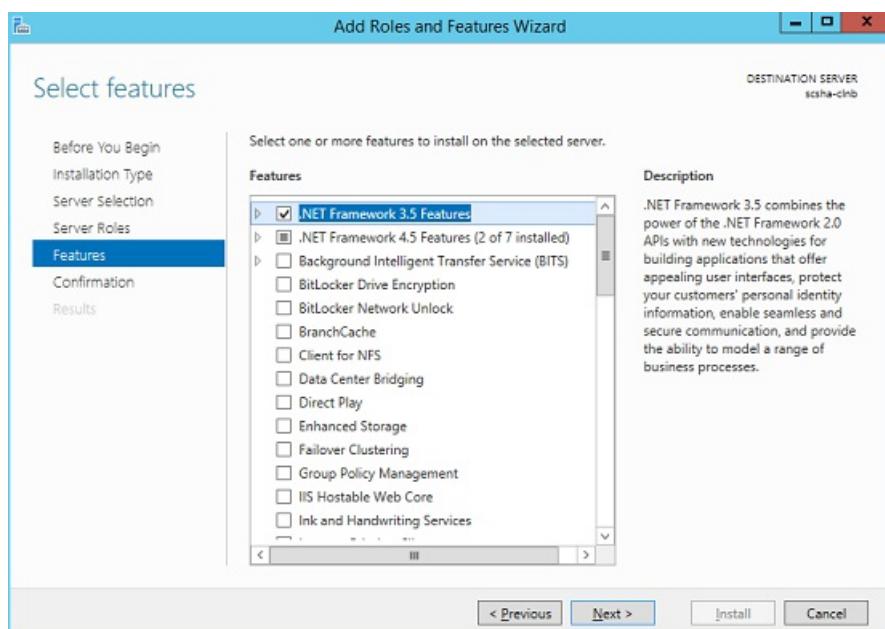


Figure 39: Install the .NET Framework 3.5 by using the Add Roles and Features Wizard

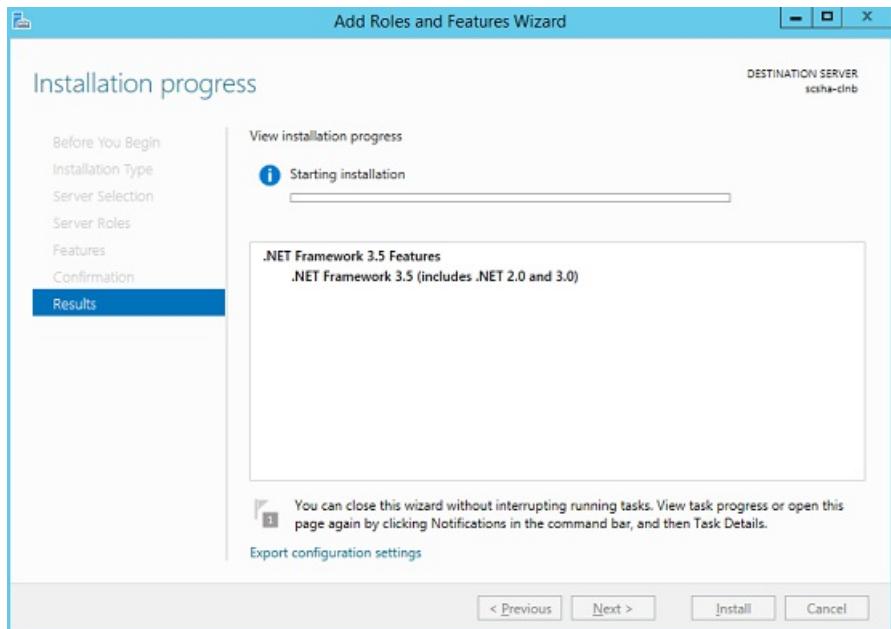


Figure 40: Installation progress bar when you install the .NET Framework 3.5 by using the Add Roles and Features Wizard

- Use the command-line tool dism.exe. For this type of installation, you need to access the SxS directory on the Windows installation media. At an elevated command prompt, type:

```
Dism /online /enable-feature /featurename:NetFx3 /All /Source:installation_media_drive:\sources\sxs  
/LimitAccess
```

Install SIOS DataKeeper

Install SIOS DataKeeper Cluster Edition on each node in the cluster. To create virtual shared storage with SIOS DataKeeper, create a synced mirror and then simulate cluster shared storage.

Before you install the SIOS software, create the domain user **DataKeeperSvc**.

NOTE

Add the **DataKeeperSvc** user to the **Local Administrator** group on both cluster nodes.

To install SIOS DataKeeper:

1. Install the SIOS software on both cluster nodes.

DK-8.2.0-Setup 8/21/2014 4:44 PM Application

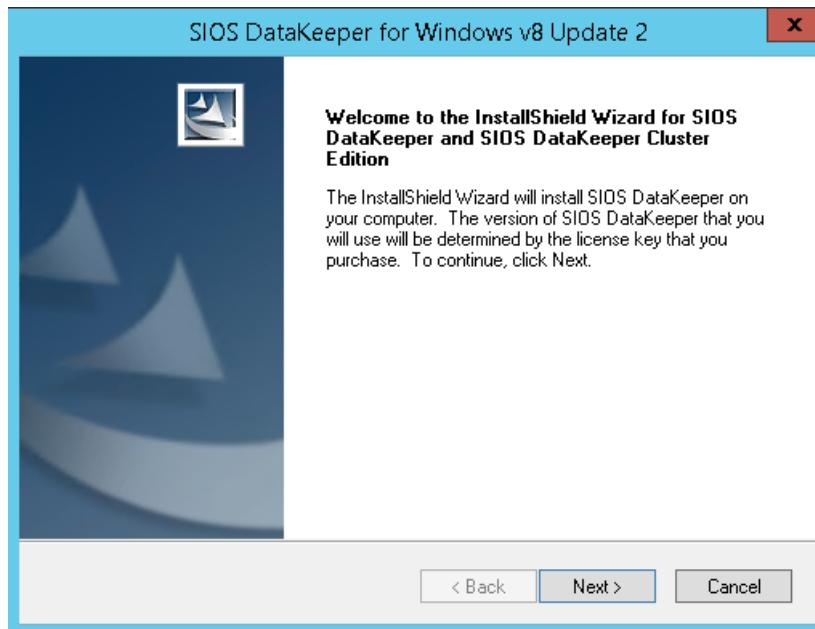


Figure 41: First page of the SIOS DataKeeper installation

2. In the dialog box shown in Figure 42, select **Yes**.

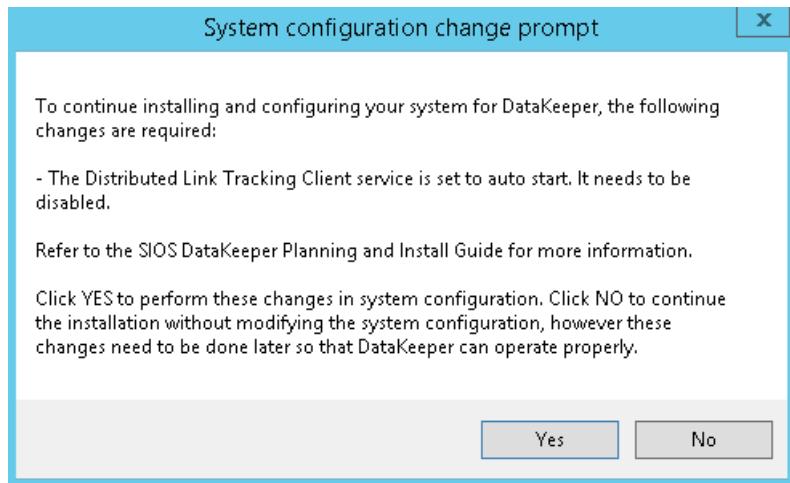


Figure 42: DataKeeper informs you that a service will be disabled

3. In the dialog box shown in Figure 43, we recommend that you select **Domain or Server account**.

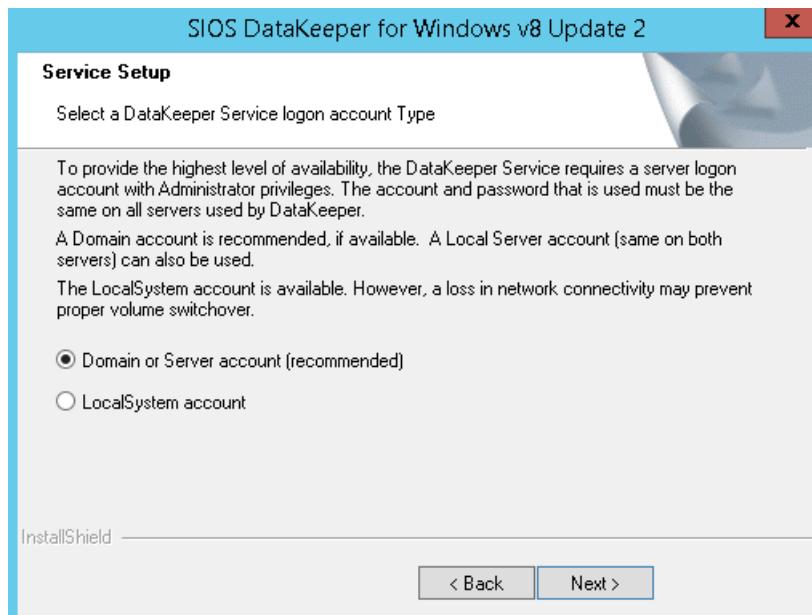


Figure 43: User selection for SIOS DataKeeper

- Enter the domain account user name and passwords that you created for SIOS DataKeeper.

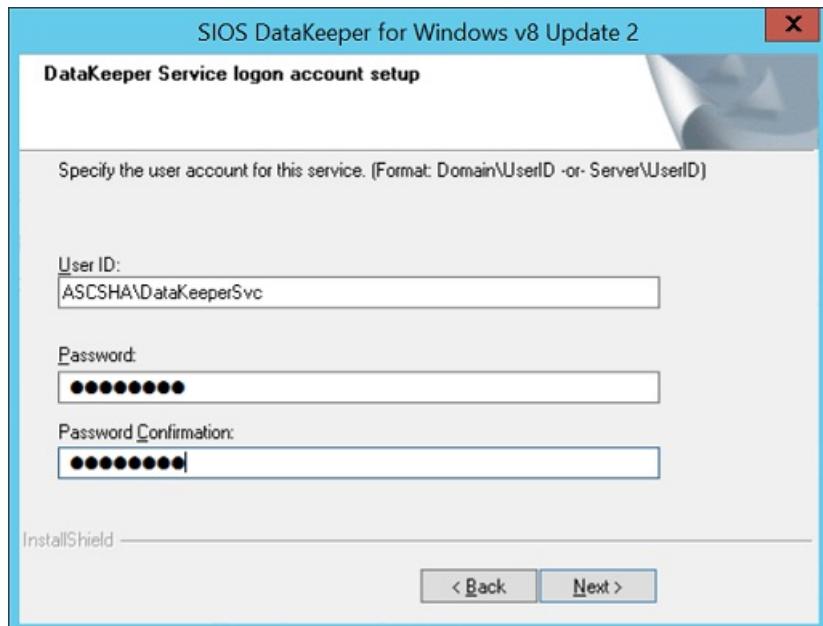


Figure 44: Enter the domain user name and password for the SIOS DataKeeper installation

- Install the license key for your SIOS DataKeeper instance as shown in Figure 45.

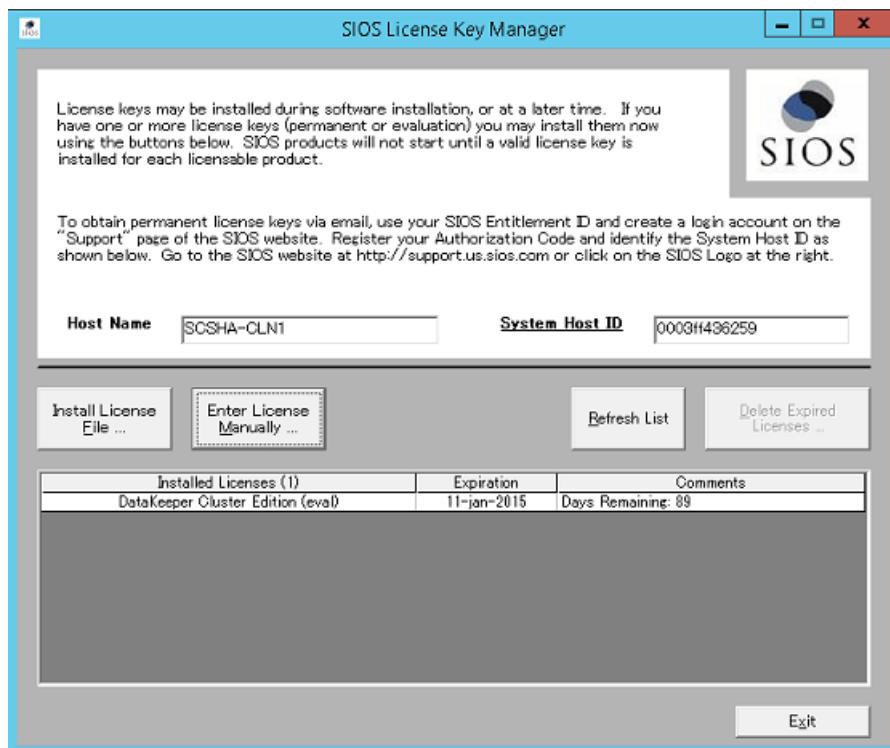


Figure 45: Enter your SIOS DataKeeper license key

- When prompted, restart the virtual machine.

Set up SIOS DataKeeper

After you install SIOS DataKeeper on both nodes, you need to start the configuration. The goal of the configuration is to have synchronous data replication between the additional VHDs attached to each of the virtual machines.

- Start the DataKeeper Management and Configuration tool, and then select **Connect Server**. (In Figure 46, this option is circled in red.)

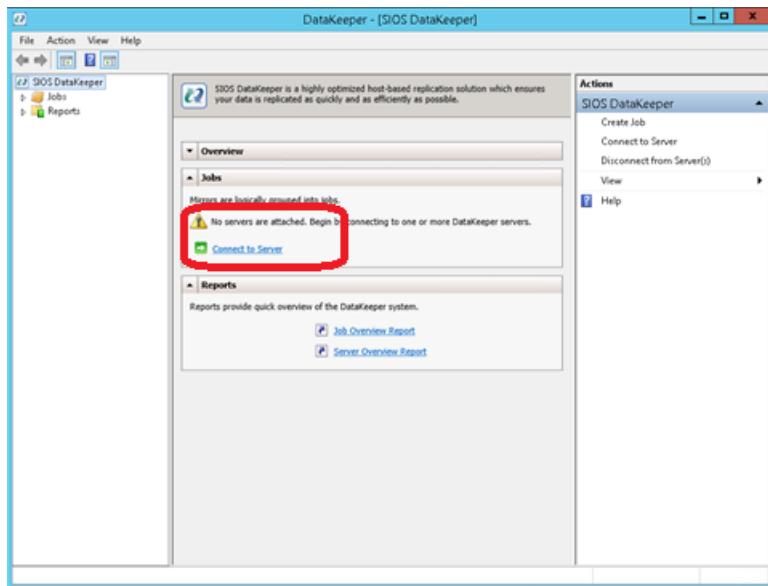


Figure 46: SIOS DataKeeper Management and Configuration tool

2. Enter the name or TCP/IP address of the first node the Management and Configuration tool should connect to, and, in a second step, the second node.

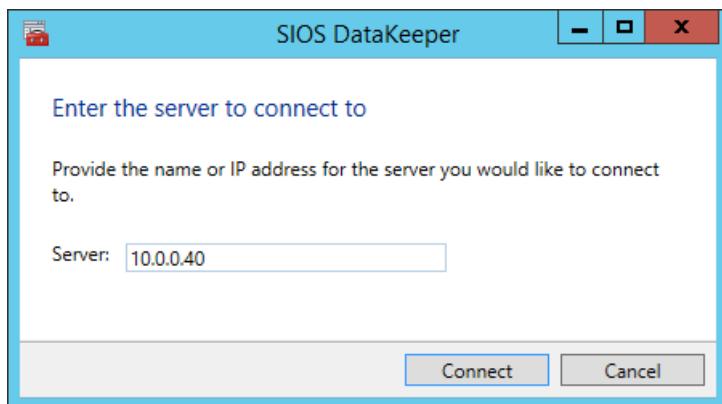


Figure 47: Insert the name or TCP/IP address of the first node the Management and Configuration tool should connect to, and in a second step, the second node

3. Create the replication job between the two nodes.



Figure 48: Create a replication job

A wizard guides you through the process of creating a replication job.

4. Define the name, TCP/IP address, and disk volume of the source node.

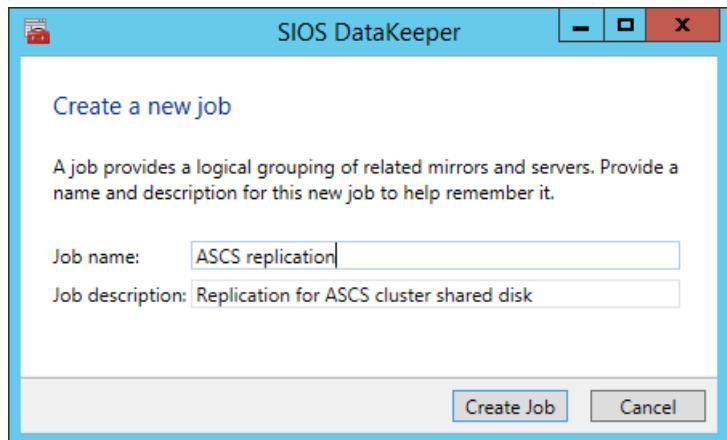


Figure 49: Define the name of the replication job

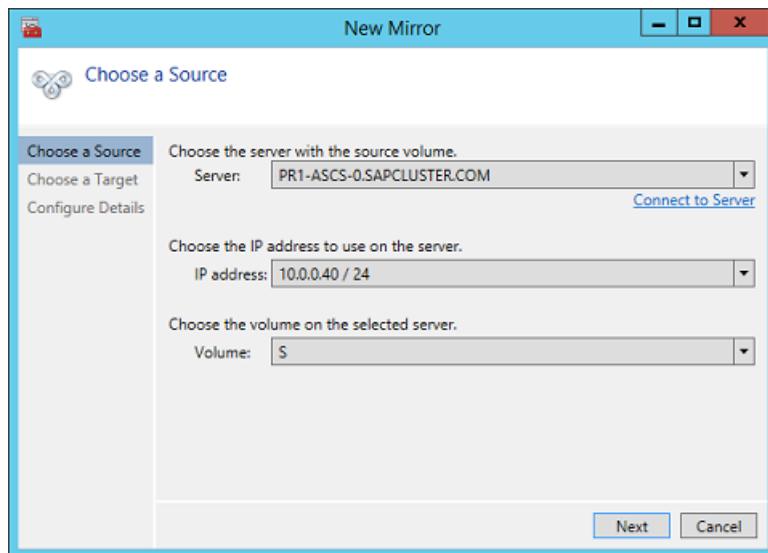


Figure 50: Define the base data for the node, which should be the current source node

5. Define the name, TCP/IP address, and disk volume of the target node.

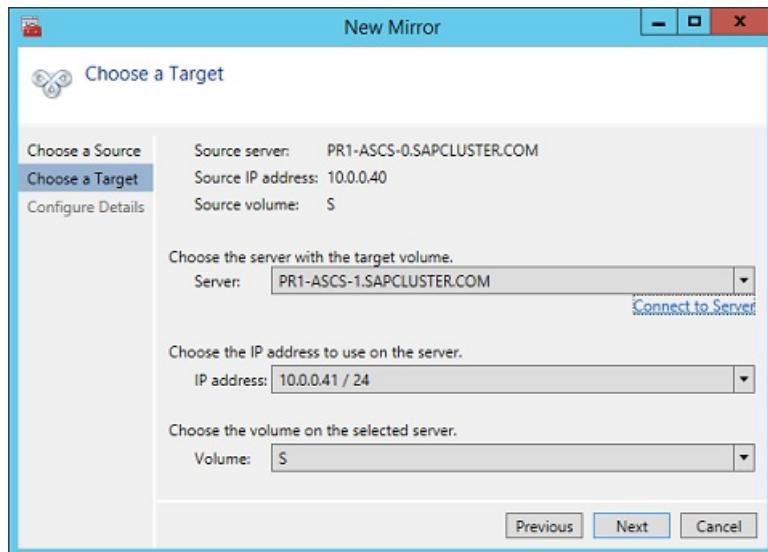


Figure 51: Define the base data for the node, which should be the current target node

6. Define the compression algorithms. In our example, we recommend that you compress the replication stream. Especially in resynchronization situations, the compression of the replication stream dramatically reduces resynchronization time. Note that compression uses the CPU and RAM resources of a virtual machine. As the compression rate increases, so does the volume of CPU resources used. You also can adjust this setting later.

7. Another setting you need to check is whether the replication occurs asynchronously or synchronously.
When you protect SAP ASCS/SCS configurations, you must use synchronous replication.

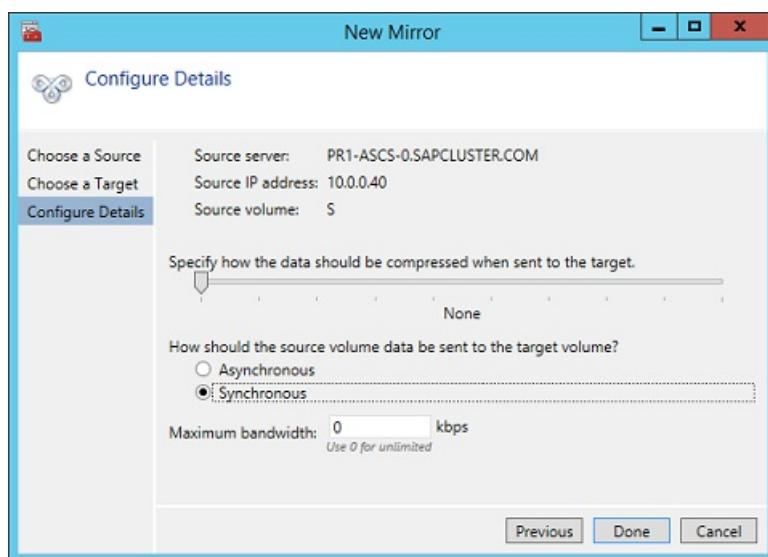


Figure 52: Define replication details

8. Define whether the volume that is replicated by the replication job should be represented to a Windows Server Failover Clustering cluster configuration as a shared disk. For the SAP ASCS/SCS configuration, select **Yes** so that the Windows cluster sees the replicated volume as a shared disk that it can use as a cluster volume.

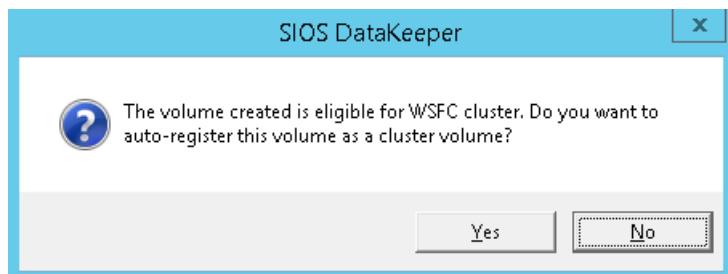


Figure 53: Select **Yes** to set the replicated volume as a cluster volume

After the volume is created, the DataKeeper Management and Configuration tool shows that the replication job is active.

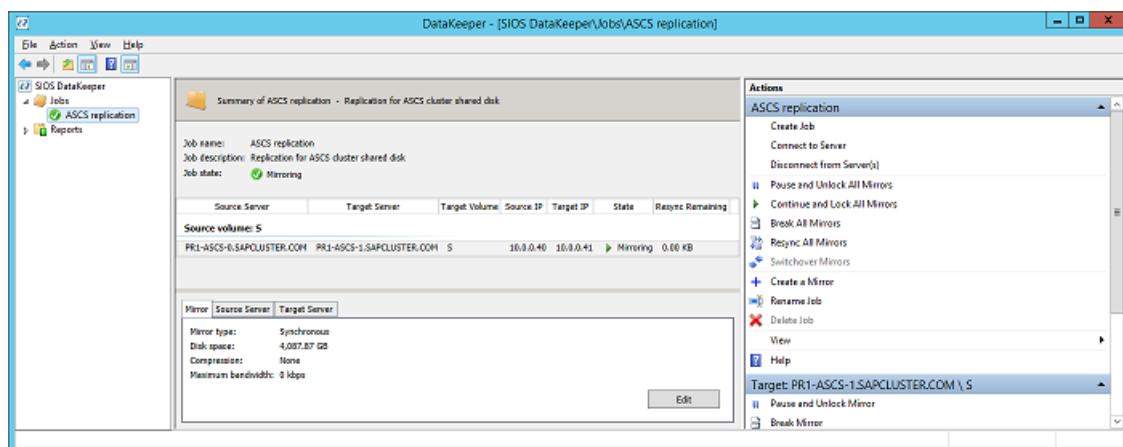


Figure 54: DataKeeper synchronous mirroring for the SAP ASCS/SCS share disk is active

Failover Cluster Manager now shows the disk as a DataKeeper disk, as shown in Figure 55.

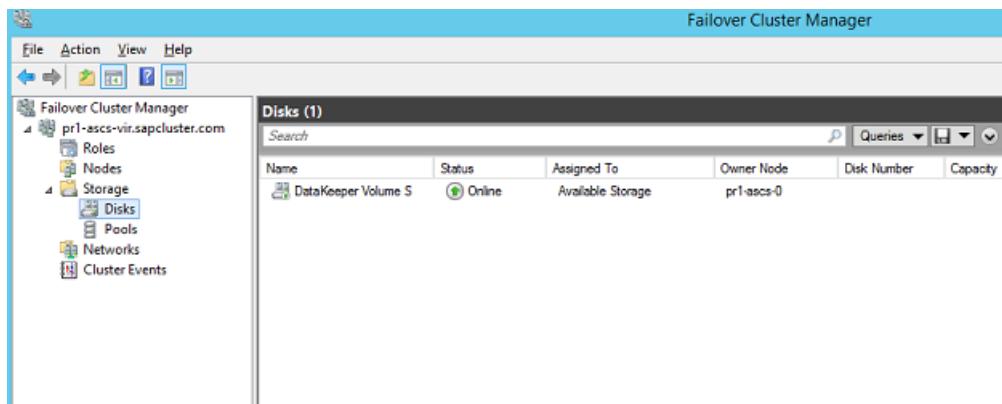


Figure 55: Failover Cluster Manager shows the disk that DataKeeper replicated

Install the SAP NetWeaver system

We won't describe the DBMS setup because setups vary depending on the DBMS system you use. However, we assume that high-availability concerns with the DBMS are addressed with the functionalities the different DBMS vendors support for Azure. For example, Always On or database mirroring for SQL Server, and Oracle Data Guard for Oracle databases. In the scenario we use in this article, we didn't add more protection to the DBMS.

There are no special considerations when different DBMS services interact with this kind of clustered SAP ASCS/SCS configuration in Azure.

NOTE

The installation procedures of SAP NetWeaver ABAP systems, Java systems, and ABAP+Java systems are almost identical. The most significant difference is that an SAP ABAP system has one ASCS instance. The SAP Java system has one SCS instance. The SAP ABAP+Java system has one ASCS instance and one SCS instance running in the same Microsoft failover cluster group. Any installation differences for each SAP NetWeaver installation stack are explicitly mentioned. You can assume that all other parts are the same.

Install SAP with a high-availability ASCS/SCS instance

IMPORTANT

Be sure not to place your page file on DataKeeper mirrored volumes. DataKeeper does not support mirrored volumes. You can leave your page file on the temporary drive D of an Azure virtual machine, which is the default. If it's not already there, move the Windows page file to drive D of your Azure virtual machine.

Installing SAP with a high-availability ASCS/SCS instance involves these tasks:

- Creating a virtual host name for the clustered SAP ASCS/SCS instance
- Installing the SAP first cluster node
- Modifying the SAP profile of the ASCS/SCS instance
- Adding a probe port
- Opening the Windows firewall probe port

Create a virtual host name for the clustered SAP ASCS/SCS instance

1. In the Windows DNS manager, create a DNS entry for the virtual host name of the ASCS/SCS instance.

IMPORTANT

The IP address that you assign to the virtual host name of the ASCS/SCS instance must be the same as the IP address that you assigned to Azure Load Balancer (**<SID>-lb-ascs**).

The IP address of the virtual SAP ASCS/SCS host name (**pr1-ascs-sap**) is the same as the IP address of Azure Load Balancer (**pr1-lb-ascs**).

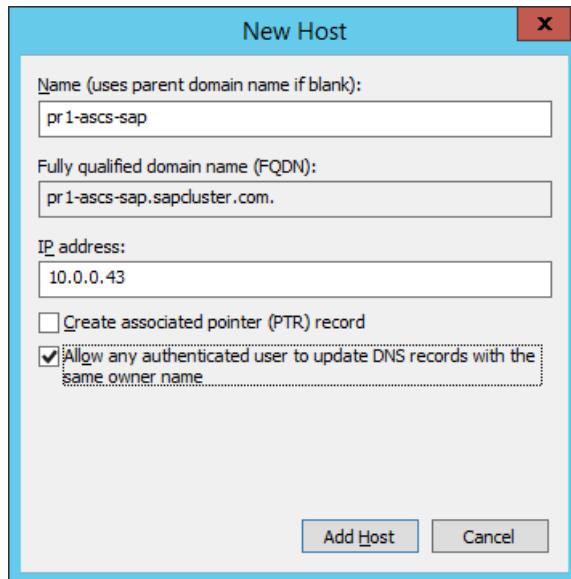


Figure 56: Define the DNS entry for the SAP ASCS/SCS cluster virtual name and TCP/IP address

2. To define the IP address assigned to the virtual host name, select **DNS Manager > Domain**.

| DNS Manager | | | | |
|------------------------|--|-------------------------|--------------------------|--------------------------------|
| | | Name | Type | Data |
| | | | | Timestamp |
| DNS | | _msdcs | | |
| DOMCONTR-0 | | _sites | | |
| Global Logs | | _tcp | | |
| Forward Lookup Zones | | _udp | | |
| _msdcs.sapcluster.co | | DomainDnsZones | | |
| sapcluster.com | | ForestDnsZones | | |
| Reverse Lookup Zones | | (same as parent folder) | Start of Authority (SOA) | [28], domcontr-0.sapclust... |
| Trust Points | | (same as parent folder) | Name Server (NS) | domcontr-0.sapcluster.co... |
| Conditional Forwarders | | (same as parent folder) | Host (A) | 10.0.0.10 6/2/2016 10:00:00 AM |
| DOMCONTR-0 | | domcontr-0 | Host (A) | 10.0.0.10 static |
| pr1-ascs-0 | | pr1-ascs-0 | Host (A) | 10.0.0.40 6/2/2016 10:00:00 AM |
| pr1-ascs-1 | | pr1-ascs-1 | Host (A) | 10.0.0.41 6/2/2016 10:00:00 AM |
| pr1-ascs-vir | | pr1-ascs-vir | Host (A) | 10.0.0.42 static |
| pr1-db-0 | | pr1-db-0 | Host (A) | 10.0.0.30 6/2/2016 12:00:00 PM |
| pr1-db-1 | | pr1-db-1 | Host (A) | 10.0.0.31 6/2/2016 12:00:00 PM |
| pr1-di-0 | | pr1-di-0 | Host (A) | 10.0.0.50 6/2/2016 1:00:00 PM |
| pr1-ascs-sap | | pr1-ascs-sap | Host (A) | 10.0.0.43 |

Figure 57: New virtual name and TCP/IP address for SAP ASCS/SCS cluster configuration

Install the SAP first cluster node

1. Execute the first cluster node option on cluster node A. For example, on the **pr1-ascs-0** host.
2. To keep the default ports for the Azure internal load balancer, select:

- **ABAP system: ASCS** instance number **00**
- **Java system: SCS** instance number **01**
- **ABAP+Java system: ASCS** instance number **00** and **SCS** instance number **01**

To use instance numbers other than 00 for the ABAP ASCS instance and 01 for the Java SCS instance, first you need to change the Azure internal load balancer default load balancing rules, described in [Change the](#)

ASCS/SCS default load balancing rules for the Azure internal load balancer.

The next few tasks aren't described in the standard SAP installation documentation.

NOTE

The SAP installation documentation describes how to install the first ASCS/SCS cluster node.

Modify the SAP profile of the ASCS/SCS instance

You need to add a new profile parameter. The profile parameter prevents connections between SAP work processes and the enqueue server from closing when they are idle for too long. We mentioned the problem scenario in [Add registry entries on both cluster nodes of the SAP ASCS/SCS instance](#). In that section, we also introduced two changes to some basic TCP/IP connection parameters. In a second step, you need to set the enqueue server to send a `keep_alive` signal so that the connections don't hit the Azure internal load balancer's idle threshold.

To modify the SAP profile of the ASCS/SCS instance:

1. Add this profile parameter to the SAP ASCS/SCS instance profile:

```
enqueue/encni/set_so_keepalive = true
```

In our example, the path is:

```
<ShareDisk>:\usr\sap\PR1\SYS\profile\PR1_ASCS00_pr1-ascs-sap
```

For example, to the SAP SCS instance profile and corresponding path:

```
<ShareDisk>:\usr\sap\PR1\SYS\profile\PR1_SCS01_pr1-ascs-sap
```

2. To apply the changes, restart the SAP ASCS /SCS instance.

Add a probe port

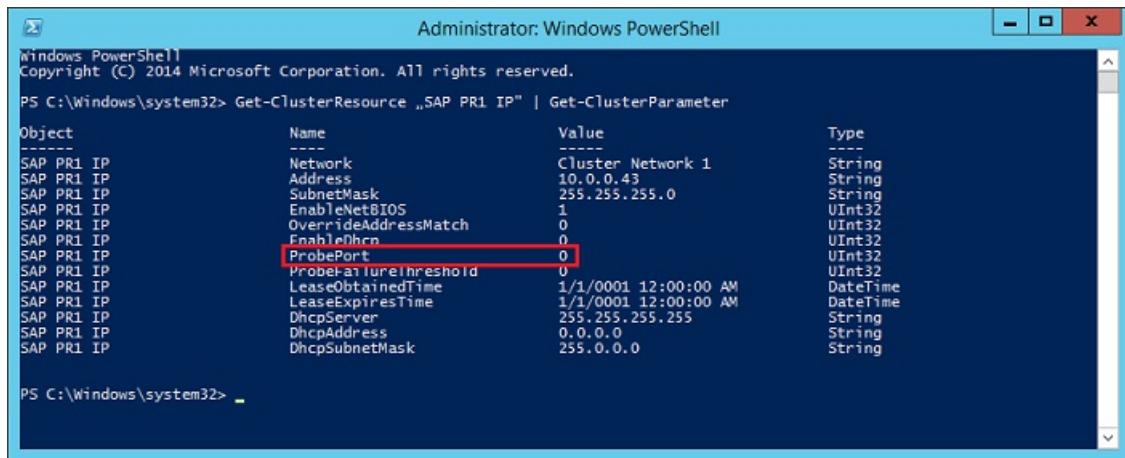
Use the internal load balancer's probe functionality to make the entire cluster configuration work with Azure Load Balancer. The Azure internal load balancer usually distributes the incoming workload equally between participating virtual machines. However, this won't work in some cluster configurations because only one instance is active. The other instance is passive and can't accept any of the workload. A probe functionality helps when the Azure internal load balancer assigns work only to an active instance. With the probe functionality, the internal load balancer can detect which instances are active, and then target only the instance with the workload.

To add a probe port:

1. Check the current **ProbePort** setting by running the following PowerShell command. Execute it from within one of the virtual machines in the cluster configuration.

```
$SAPSID = "PR1"      # SAP <SID>  
  
$SAPNetworkIPClusterName = "SAP $SAPSID IP"  
Get-ClusterResource $SAPNetworkIPClusterName | Get-ClusterParameter
```

2. Define a probe port. The default probe port number is **0**. In our example, we use probe port **62000**.



The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command run is "PS C:\Windows\system32> Get-ClusterResource „SAP PR1 IP“ | Get-ClusterParameter". The output is a table with columns "Object", "Name", "Value", and "Type". The "ProbePort" row is highlighted with a red box. The "Value" column for "ProbePort" is 0.

| Object | Name | Value | Type |
|------------|-----------------------|----------------------|----------|
| SAP PR1 IP | Network | Cluster Network 1 | String |
| SAP PR1 IP | Address | 10.0.0.43 | String |
| SAP PR1 IP | SubnetMask | 255.255.255.0 | String |
| SAP PR1 IP | EnableNetBIOS | 1 | UInt32 |
| SAP PR1 IP | OverrideAddressMatch | 0 | UInt32 |
| SAP PR1 IP | EnableDhcp | 0 | UInt32 |
| SAP PR1 IP | ProbePort | 0 | UInt32 |
| SAP PR1 IP | ProbeFailureThreshold | 0 | UInt32 |
| SAP PR1 IP | LeaseObtainedTime | 1/1/0001 12:00:00 AM | DateTime |
| SAP PR1 IP | LeaseExpiresTime | 1/1/0001 12:00:00 AM | DateTime |
| SAP PR1 IP | DhcpServer | 255.255.255.255 | String |
| SAP PR1 IP | DhcpAddress | 0.0.0.0 | String |
| SAP PR1 IP | DhcpSubnetMask | 255.0.0.0 | String |

Figure 58: The default cluster configuration probe port is 0

The port number is defined in SAP Azure Resource Manager templates. You can assign the port number in PowerShell.

To set a new ProbePort value for the **SAP <SID> IP** cluster resource, run the following PowerShell script. Update the PowerShell variables for your environment. After the script runs, you'll be prompted to restart the SAP cluster group to activate the changes.

```

$SAPSID = "PR1"      # SAP <SID>
$ProbePort = 62000    # ProbePort of the Azure Internal Load Balancer

Clear-Host
$SAPClusterRoleName = "SAP $SAPSID"
$SAPIPResourceName = "SAP $SAPSID IP"
$SAPIPResourceClusterParameters = Get-ClusterResource $SAPIPResourceName | Get-ClusterParameter
$IPAddress = ($SAPIPResourceClusterParameters | Where-Object {$_.Name -eq "Address"}).Value
$NetworkName = ($SAPIPResourceClusterParameters | Where-Object {$_.Name -eq "Network"}).Value
$SubnetMask = ($SAPIPResourceClusterParameters | Where-Object {$_.Name -eq "SubnetMask"}).Value
$OverrideAddressMatch = ($SAPIPResourceClusterParameters | Where-Object {$_.Name -eq "OverrideAddressMatch"}).Value
$EnableDhcp = ($SAPIPResourceClusterParameters | Where-Object {$_.Name -eq "EnableDhcp"}).Value
$OldProbePort = ($SAPIPResourceClusterParameters | Where-Object {$_.Name -eq "ProbePort"}).Value

$var = Get-ClusterResource | Where-Object { $_.name -eq $SAPIPResourceName }

Write-Host "Current configuration parameters for SAP IP cluster resource '$SAPIPResourceName' are:" -
ForegroundColor Cyan
Get-ClusterResource -Name $SAPIPResourceName | Get-ClusterParameter

Write-Host
Write-Host "Current probe port property of the SAP cluster resource '$SAPIPResourceName' is
'$OldProbePort'." -ForegroundColor Cyan
Write-Host
Write-Host "Setting the new probe port property of the SAP cluster resource '$SAPIPResourceName' to
'$ProbePort' ..." -ForegroundColor Cyan
Write-Host

$var | Set-ClusterParameter -Multiple
@{ "Address"=$IPAddress;"ProbePort"=$ProbePort;"Subnetmask"=$SubnetMask;"Network"=$NetworkName;"OverrideAddressMatch"=$OverrideAddressMatch;"EnableDhcp"=$EnableDhcp}

Write-Host

$ActivateChanges = Read-Host "Do you want to take restart SAP cluster role '$SAPClusterRoleName', to
activate the changes (yes/no)?"

if($ActivateChanges -eq "yes"){
Write-Host
Write-Host "Activating changes..." -ForegroundColor Cyan

Write-Host
write-host "Taking SAP cluster IP resource '$SAPIPResourceName' offline ..." -ForegroundColor Cyan
Stop-ClusterResource -Name $SAPIPResourceName
sleep 5

Write-Host "Starting SAP cluster role '$SAPClusterRoleName' ..." -ForegroundColor Cyan
Start-ClusterGroup -Name $SAPClusterRoleName

Write-Host "New ProbePort parameter is active." -ForegroundColor Green
Write-Host

Write-Host "New configuration parameters for SAP IP cluster resource '$SAPIPResourceName': " -
ForegroundColor Cyan
Write-Host
Get-ClusterResource -Name $SAPIPResourceName | Get-ClusterParameter
}else
{
Write-Host "Changes are not activated."
}

```

After you bring the **SAP <SID>** cluster role online, verify that **ProbePort** is set to the new value.

```
$SAPSID = "PR1"      # SAP <SID>

$SAPNetworkIPClusterName = "SAP $SAPSID IP"
Get-ClusterResource $SAPNetworkIPClusterName | Get-ClusterParameter
```

| Object | Name | Value | Type |
|------------|-----------------------|----------------------|----------|
| SAP PR1 IP | Network | Cluster Network 1 | String |
| SAP PR1 IP | Address | 10.0.0.43 | String |
| SAP PR1 IP | SubnetMask | 255.255.255.0 | String |
| SAP PR1 IP | EnableNetBIOS | 1 | UInt32 |
| SAP PR1 IP | OverrideAddressMatch | 1 | UInt32 |
| SAP PR1 IP | EnableDhcp | 0 | UInt32 |
| SAP PR1 IP | ProbePort | 62300 | UInt32 |
| SAP PR1 IP | ProbeFailureThreshold | 0 | UInt32 |
| SAP PR1 IP | LeaseObtainedTime | 1/1/0001 12:00:00 AM | Datetime |
| SAP PR1 IP | LeaseExpiresTime | 1/1/0001 12:00:00 AM | Datetime |
| SAP PR1 IP | DhcpServer | 255.255.255.255 | String |
| SAP PR1 IP | DhcpAddress | 0.0.0.0 | String |
| SAP PR1 IP | DhcpSubnetMask | 255.0.0.0 | String |

Figure 59: Probe the cluster port after you set the new value

Open the Windows firewall probe port

You need to open a Windows firewall probe port on both cluster nodes. Use the following script to open a Windows firewall probe port. Update the PowerShell variables for your environment.

```
$ProbePort = 62000  # ProbePort of the Azure Internal Load Balancer

New-NetFirewallRule -Name AzureProbePort -DisplayName "Rule for Azure Probe Port" -Direction Inbound -Action
Allow -Protocol TCP -LocalPort $ProbePort
```

The **ProbePort** is set to **62000**. Now you can access the file share **\\\ascsha-clsap\sapmnt** from other hosts, such as from **ascsha-dbas**.

Install the database instance

To install the database instance, follow the process described in the SAP installation documentation.

Install the second cluster node

To install the second cluster, follow the steps in the SAP installation guide.

Change the start type of the SAP ERS Windows service instance

Change the start type of the SAP ERS Windows service to **Automatic (Delayed Start)** on both cluster nodes.

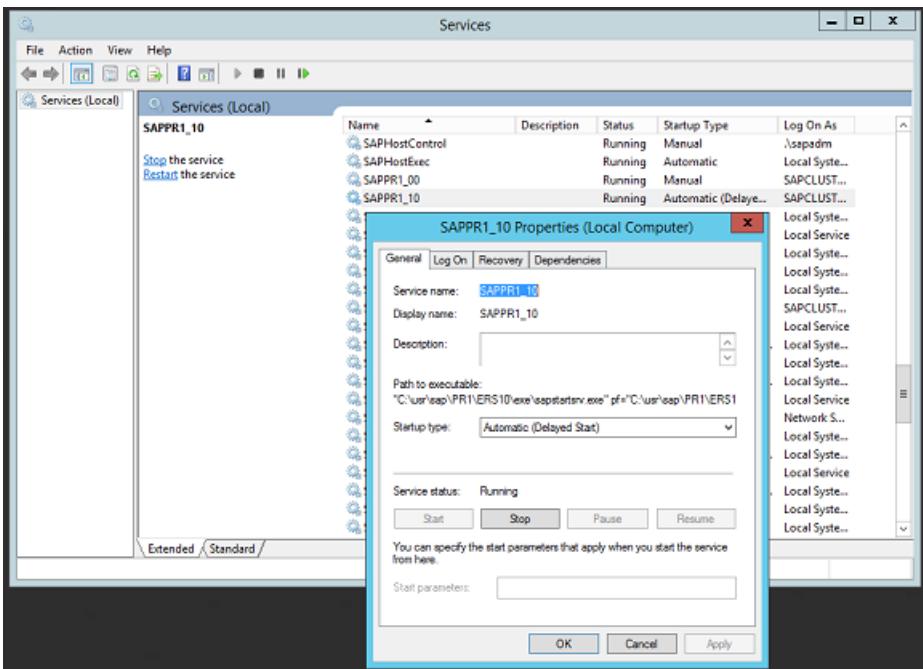


Figure 60: Change the service type for the SAP ERS instance to delayed automatic

Install the SAP Primary Application Server

Install the Primary Application Server (PAS) instance <SID>-di-0 on the virtual machine that you've designated to host the PAS. There are no dependencies on Azure or DataKeeper-specific settings.

Install the SAP Additional Application Server

Install an SAP Additional Application Server (AAS) on all the virtual machines that you've designated to host an SAP Application Server instance. For example, on <SID>-di-1 to <SID>-di-<n>.

NOTE

This finishes the installation of a high-availability SAP NetWeaver system. Next, proceed with failover testing.

Test the SAP ASCS/SCS instance failover and SIOS replication

It's easy to test and monitor an SAP ASCS/SCS instance failover and SIOS disk replication by using Failover Cluster Manager and the SIOS DataKeeper Management and Configuration tool.

SAP ASCS/SCS instance is running on cluster node A

The **SAP PR1** cluster group is running on cluster node A. For example, on **pr1-ascs-0**. Assign the shared disk drive S, which is part of the **SAP PR1** cluster group, and which the ASCS/SCS instance uses, to cluster node A.

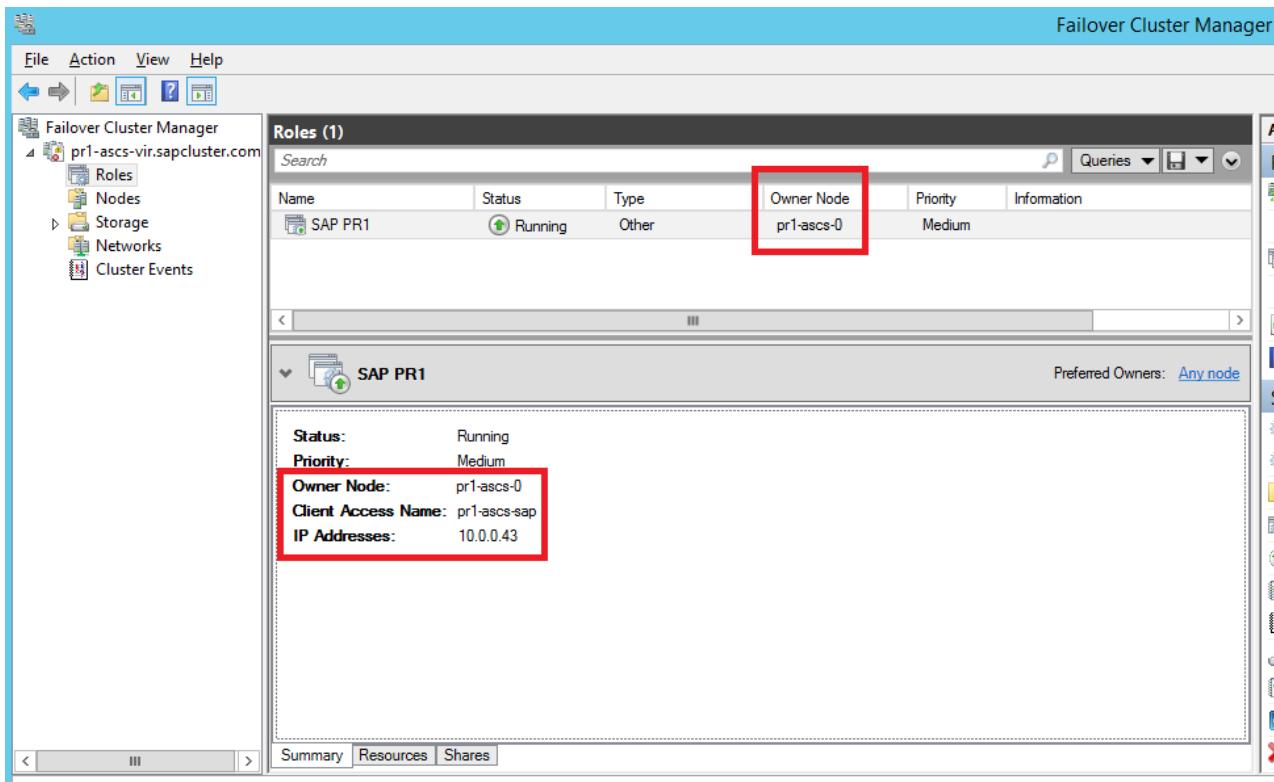


Figure 61: Failover Cluster Manager: The SAP <SID> cluster group is running on cluster node A

In the SIOS DataKeeper Management and Configuration tool, you can see that the shared disk data is synchronously replicated from the source volume drive S on cluster node A to the target volume drive S on cluster node B. For example, it's replicated from **pr1-ascs-0 [10.0.0.40]** to **pr1-ascs-1 [10.0.0.41]**.

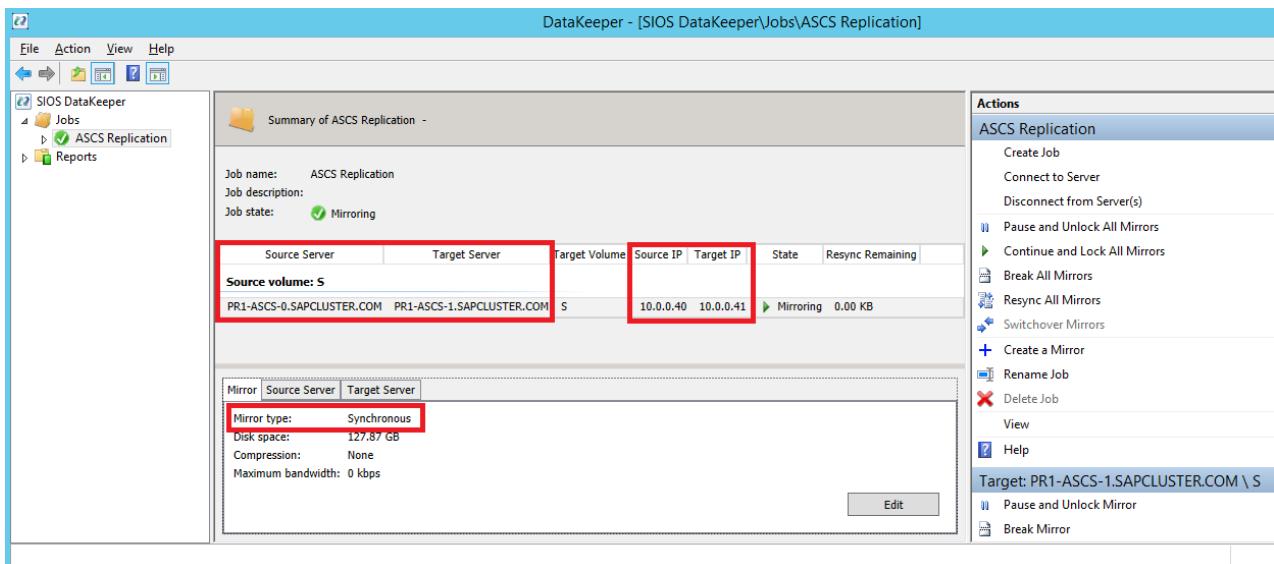


Figure 62: In SIOS DataKeeper, replicate the local volume from cluster node A to cluster node B

Failover from node A to node B

1. Choose one of these options to initiate a failover of the SAP <SID> cluster group from cluster node A to cluster node B:

- Use Failover Cluster Manager
- Use Failover Cluster PowerShell

```

$SAPSID = "PR1"      # SAP <SID>

$SAPClusterGroup = "SAP $SAPSID"
Move-ClusterGroup -Name $SAPClusterGroup

```

2. Restart cluster node A within the Windows guest operating system (this initiates an automatic failover of the SAP <SID> cluster group from node A to node B).
3. Restart cluster node A from the Azure portal (this initiates an automatic failover of the SAP <SID> cluster group from node A to node B).
4. Restart cluster node A by using Azure PowerShell (this initiates an automatic failover of the SAP <SID> cluster group from node A to node B).

After failover, the SAP <SID> cluster group is running on cluster node B. For example, it's running on **pr1-ascs-1**.

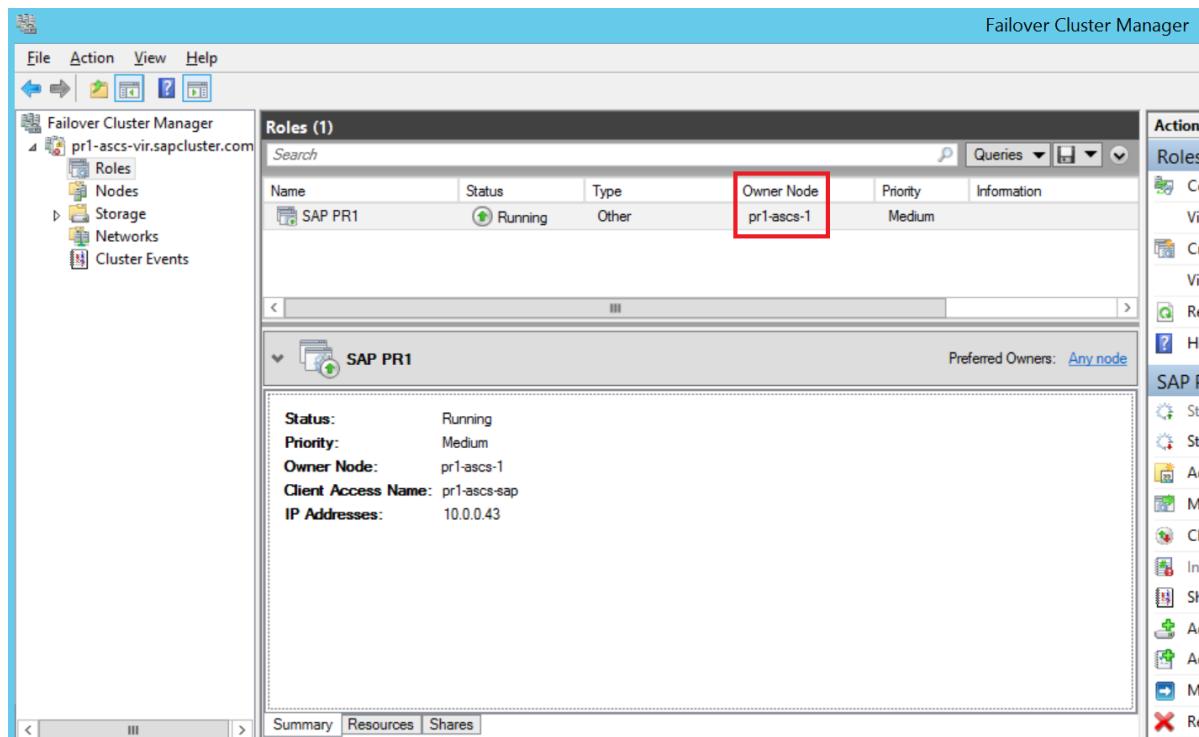


Figure 63: In Failover Cluster Manager, the SAP <SID> cluster group is running on cluster node B

The shared disk is now mounted on cluster node B. SIOS DataKeeper is replicating data from source volume drive S on cluster node B to target volume drive S on cluster node A. For example, it's replicating from **pr1-ascs-1 [10.0.0.41]** to **pr1-ascs-0 [10.0.0.40]**.

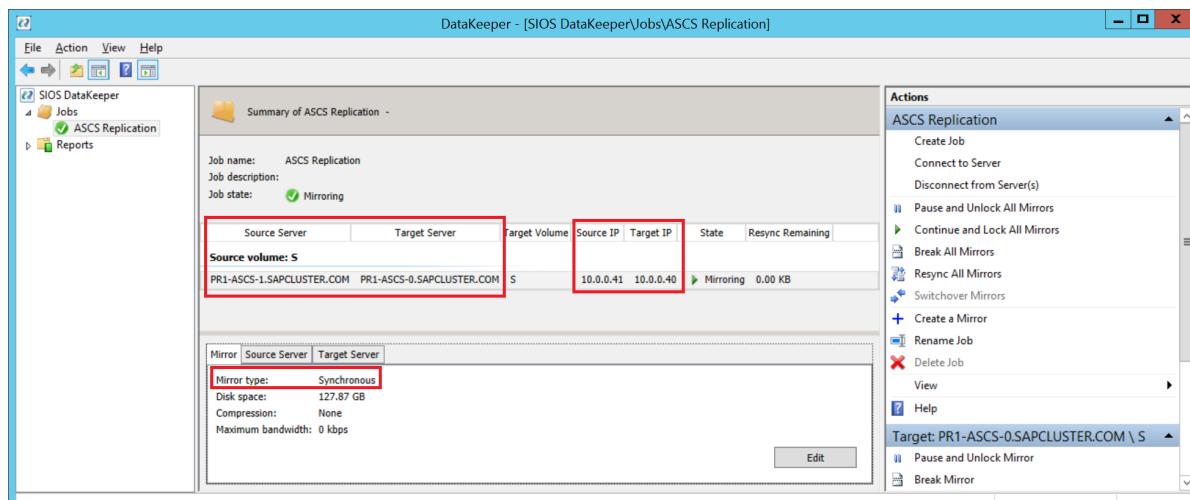


Figure 64: SIOS DataKeeper replicates the local volume from cluster node B to cluster node A

SAP NetWeaver on Azure Virtual Machines (VMs) – DBMS Deployment Guide

1/17/2017 • 90 min to read • [Edit on GitHub](#)

This guide is part of the documentation on implementing and deploying the SAP software on Microsoft Azure. Before reading this guide, please read the [Planning and Implementation Guide](#). This document covers the deployment of various Relational Database Management Systems (RDBMS) and related products in combination with SAP on Microsoft Azure Virtual Machines (VMs) using the Azure Infrastructure as a Service (IaaS) capabilities.

The paper complements the SAP Installation Documentation and SAP Notes which represent the primary resources for installations and deployments of SAP software on given platforms

General considerations

In this chapter, considerations of running SAP related DBMS systems in Azure VMs are introduced. There are few references to specific DBMS systems in this chapter. Instead the specific DBMS systems are handled within this paper, after this chapter.

Definitions upfront

Throughout the document we will use the following terms:

- IaaS: Infrastructure as a Service.
- PaaS: Platform as a Service.
- SaaS: Software as a Service.
- SAP Component: an individual SAP application such as ECC, BW, Solution Manager or EP. SAP components can be based on traditional ABAP or Java technologies or a non-NetWeaver based application such as Business Objects.
- SAP Environment: one or more SAP components logically grouped to perform a business function such as Development, QAS, Training, DR or Production.
- SAP Landscape: This refers to the entire SAP assets in a customer's IT landscape. The SAP landscape includes all production and non-production environments.
- SAP System: The combination of DBMS layer and application layer of e.g. an SAP ERP development system, SAP BW test system, SAP CRM production system, etc. In Azure deployments it is not supported to divide these two layers between on-premises and Azure. This means an SAP system is either deployed on-premises or it is deployed in Azure. However, you can deploy the different systems of an SAP landscape in Azure or on-premises. For example, you could deploy the SAP CRM development and test systems in Azure but the SAP CRM production system on-premises.
- Cloud-Only deployment: A deployment where the Azure subscription is not connected via a site-to-site or ExpressRoute connection to the on-premises network infrastructure. In common Azure documentation these kinds of deployments are also described as 'Cloud-Only' deployments. Virtual Machines deployed with this method are accessed through the Internet and public Internet endpoints assigned to the VMs in Azure. The on-premises Active Directory (AD) and DNS is not extended to Azure in these types of deployments. Hence the VMs are not part of the on-premises Active Directory. Note: Cloud-Only deployments in this document are defined as complete SAP landscapes which are running exclusively in Azure without extension of Active Directory or name resolution from on-premises into public cloud. Cloud-Only configurations are not supported for production SAP systems or configurations where SAP STMS or other on-premises resources need to be used between SAP systems hosted on Azure and resources residing

on-premises.

- Cross-Premises: Describes a scenario where VMs are deployed to an Azure subscription that has site-to-site, multi-site or ExpressRoute connectivity between the on-premises datacenter(s) and Azure. In common Azure documentation, these kinds of deployments are also described as Cross-Premises scenarios. The reason for the connection is to extend on-premises domains, on-premises Active Directory and on-premises DNS into Azure. The on-premises landscape is extended to the Azure assets of the subscription. Having this extension, the VMs can be part of the on-premises domain. Domain users of the on-premises domain can access the servers and can run services on those VMs (like DBMS services). Communication and name resolution between VMs deployed on-premises and VMs deployed in Azure is possible. We expect this to be the most common scenario for deploying SAP assets on Azure. See [this article](#) and [this](#) for more information.

NOTE

Cross-Premises deployments of SAP systems where Azure Virtual Machines running SAP systems are members of an on-premises domain are supported for production SAP systems. Cross-Premises configurations are supported for deploying parts or complete SAP landscapes into Azure. Even running the complete SAP landscape in Azure requires having those VMs being part of on-premises domain and ADS. In former versions of the documentation we talked about Hybrid-IT scenarios, where the term 'Hybrid' is rooted in the fact that there is a cross-premises connectivity between on-premises and Azure. In this case 'Hybrid' also means that the VMs in Azure are part of the on-premises Active Directory.

Some Microsoft documentation describes Cross-Premises scenarios a bit differently, especially for DBMS HA configurations. In the case of the SAP related documents, the Cross-Premises scenario just boils down to having a site-to-site or private (ExpressRoute) connectivity and to the fact that the SAP landscape is distributed between on-premises and Azure.

Resources

The following guides are available for the topic of SAP deployments on Azure:

- [SAP NetWeaver on Azure Virtual Machines \(VMs\) – Planning and Implementation Guide](#)
- [SAP NetWeaver on Azure Virtual Machines \(VMs\) – Deployment Guide](#)
- [SAP NetWeaver on Azure Virtual Machines \(VMs\) – DBMS Deployment Guide \(this document\)](#)
- [SAP NetWeaver on Azure Virtual Machines \(VMs\) – High Availability Deployment Guide](#)

The following SAP Notes are related to the topic of SAP on Azure:

| NOTE NUMBER | TITLE |
|-------------|--|
| 1928533 | SAP Applications on Azure: Supported Products and Azure VM types |
| 2015553 | SAP on Microsoft Azure: Support Prerequisites |
| 1999351 | Troubleshooting Enhanced Azure Monitoring for SAP |
| 2178632 | Key Monitoring Metrics for SAP on Microsoft Azure |
| 1409604 | Virtualization on Windows: Enhanced Monitoring |
| 2191498 | SAP on Linux with Azure: Enhanced Monitoring |
| 2039619 | SAP Applications on Microsoft Azure using the Oracle Database: Supported Products and Versions |

| NOTE NUMBER | TITLE |
|-------------|--|
| 2233094 | DB6: SAP Applications on Azure Using IBM DB2 for Linux, UNIX, and Windows - Additional Information |
| 2243692 | Linux on Microsoft Azure (IaaS) VM: SAP license issues |
| 1984787 | SUSE LINUX Enterprise Server 12: Installation notes |
| 2002167 | Red Hat Enterprise Linux 7.x: Installation and Upgrade |

Please also read the [SCN Wiki](#) that contains all SAP Notes for Linux.

You should have a working knowledge about the Microsoft Azure Architecture and how Microsoft Azure Virtual Machines are deployed and operated. You can find more information here

<https://azure.microsoft.com/documentation/>

NOTE

We are **not** discussing Microsoft Azure Platform as a Service (PaaS) offerings of the Microsoft Azure Platform. This paper is about running a database management system (DBMS) in Microsoft Azure Virtual Machines (IaaS) just as you would run the DBMS in your on-premises environment. Database capabilities and functionalities between these two offers are very different and should not be mixed up with each other. See also: <https://azure.microsoft.com/services/sql-database/>

Since we are discussing IaaS, in general the Windows, Linux and DBMS installation and configuration are essentially the same as any virtual machine or bare metal machine you would install on-premises. However, there are some architecture and system management implementation decisions which will be different when utilizing IaaS. The purpose of this document is to explain the specific architectural and system management differences that you must be prepared for when using IaaS.

In general, the overall areas of difference that this paper will discuss are:

- Planning the proper VM/VHD layout of SAP systems to ensure you have the proper data file layout and can achieve enough IOPS for your workload.
- Networking considerations when using IaaS.
- Specific database features to use in order to optimize the database layout.
- Backup and restore considerations in IaaS.
- Utilizing different types of images for deployment.
- High Availability in Azure IaaS.

Structure of a RDBMS Deployment

In order to follow this chapter, it is necessary to understand what was presented in [this](#) chapter of the [Deployment Guide](#). Knowledge about the different VM-Series and their differences and differences of Azure Standard and Premium Storage should be understood and known before reading this chapter.

Until March 2015, Azure VHDs which contain an operating system were limited to 127 GB in size. This limitation got lifted in March 2015 (for more information check

<https://azure.microsoft.com/blog/2015/03/25/azure-vm-os-drive-limit-octupled/>). From there on VHDs containing the operating system can have the same size as any other VHD. Nevertheless, we still prefer a structure of deployment where the operating system, DBMS and eventual SAP binaries are separate from the database files. Therefore, we expect SAP systems running in Azure Virtual Machines will have the base VM (or VHD) installed with the operating system, database management system executables and SAP executables. The

DBMS data and log files will be stored in Azure Storage (Standard or Premium Storage) in separate VHD files and attached as logical disks to the original Azure operating system image VM.

Dependent on leveraging Azure Standard or Premium Storage (e.g. by using the DS-series or GS-series VMs) there are other quotas in Azure which are documented [here](#). When planning your Azure VHDs, you'll need to find the best balance of the quotas for the following:

- The number of data files.
- The number of VHDs which contain the files.
- The IOPS quotas of a single VHD.
- The data throughput per VHD.
- The number of additional VHDs possible per VM size.
- The overall storage throughput a VM can provide.

Azure will enforce an IOPS quota per VHD drive. These quotas are different for VHDs hosted on Azure Standard Storage and Premium Storage. I/O latencies will also be very different between the two storage types with Premium Storage delivering factors better I/O latencies. Each of the different VM types have a limited number of VHDs that you are able to attach. Another restriction is that only certain VM types can leverage Azure Premium Storage. This means the decision for a certain VM type might not only be driven by the CPU and memory requirements, but also by the IOPS, latency and disk throughput requirements that usually are scaled with the number of VHDs or the type of Premium Storage disks. Especially with Premium Storage the size of a VHD also might be dictated by the number of IOPS and throughput that needs to be achieved by each VHD.

The fact that the overall IOPS rate, the number of VHDs mounted, and the size of the VM are all tied together, might cause an Azure configuration of an SAP system to be different than its on-premises deployment. The IOPS limits per LUN are usually configurable in on-premises deployments. Whereas with Azure Storage those limits are fixed or as in Premium Storage dependent on the disk type. So with on-premises deployments we see customer configurations of database servers which are using many different volumes for special executables like SAP and the DBMS or special volumes for temporary databases or table spaces. When such an on-premises system is moved to Azure it might lead to a waste of potential IOPS bandwidth by wasting a VHD for executables or databases which do not perform any or not a lot of IOPS. Therefore, in Azure VMs we recommend that the DBMS and SAP executables be installed on the OS disk if possible.

The placement of the database files and log files and the type of Azure Storage used, should be defined by IOPS, latency and throughput requirements. In order to have enough IOPS for the transaction log, you might be forced to leverage multiple VHDs for the transaction log file or use a larger Premium Storage disk. In such a case one would simply build a software RAID (e.g. Windows Storage Pool for Windows or MDADM and LVM (Logical Volume Manager) for Linux) with the VHDs which will contain the transaction log.

Windows

Drive D:\ in an Azure VM is a non-persisted drive which is backed by some local disks on the Azure compute node. Because it is non-persisted, this means that any changes made to the content on the D:\ drive is lost when the VM is rebooted. By "any changes", we mean saved files, directories created, applications installed, etc.

Linux

Linux Azure VMs automatically mount a drive at /mnt/resource which is a non-persisted drive backed by local disks on the Azure compute node. Because it is non-persisted, this means that any changes made to content in /mnt/resource is lost when the VM is rebooted. By any changes, we mean files saved, directories created, applications installed, etc.

Dependent on the Azure VM-series, the local disks on the compute node show different performance which can

be categorized like:

- A0-A7: Very limited performance. Not usable for anything beyond windows page file
- A8-A11: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput
- D-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput
- DS-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput
- G-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput
- GS-Series: Very good performance characteristics with some ten thousand IOPS and >1GB/sec throughput

Statements above are applying to the VM types that are certified with SAP. The VM-series with excellent IOPS and throughput qualify for leverage by some DBMS features, like tempdb or temporary table space.

Caching for VMs and VHDs

When we create those disks/VHDs through the portal or when we mount uploaded VHDs to VMs, we can choose whether the I/O traffic between the VM and those VHDs located in Azure storage are cached. Azure Standard and Premium Storage use two different technologies for this type of cache. In both cases the cache itself would be disk backed on the same drives used by the temporary disk (D:\ on Windows or /mnt/resource on Linux) of the VM.

For Azure Standard Storage the possible cache types are:

- No caching
- Read caching
- Read and Write caching

In order to get consistent and deterministic performance, you should set the caching on Azure Standard Storage for all VHDs containing **DBMS related data files, log files and table space to 'NONE'**. The caching of the VM can remain with the default.

For Azure Premium Storage the following caching options exist:

- No caching
- Read caching

Recommendation for Azure Premium Storage is to leverage **Read caching for data files** of the SAP database and chose **No caching for the VHD(s) of log file(s)**.

Software RAID

As already stated above, you will need to balance the number of IOPS needed for the database files across the number of VHDs you can configure and the maximum IOPS an Azure VM will provide per VHD or Premium Storage disk type. Easiest way to deal with the IOPS load over VHDs is to build a software RAID over the different VHDs. Then place a number of data files of the SAP DBMS on the LUNS carved out of the software RAID. Dependent on the requirements you might want to consider the usage of Premium Storage as well since two of the three different Premium Storage disks provide higher IOPS quota than VHDs based on Standard Storage. Besides the significant better I/O latency provided by Azure Premium Storage.

Same applies to the transaction log of the different DBMS systems. With a lot of them just adding more Tlog files does not help since the DBMS systems write into one of the files at a time only. If higher IOPS rates are needed than a single Standard Storage based VHD can deliver, you can stripe over multiple Standard Storage VHDs or you can use a larger Premium Storage disk type that beyond higher IOPS rates also delivers factors lower latency for the write I/Os into the transaction log.

Situations experienced in Azure deployments which would favor using a software RAID are:

- Transaction Log/Redo Log require more IOPS than Azure provides for a single VHD. As mentioned above this can be solved by building a LUN over multiple VHDs using a software RAID.

- Uneven I/O workload distribution over the different data files of the SAP database. In such cases one can experience one data file hitting the quota rather often. Whereas other data files are not even getting close to the IOPS quota of a single VHD. In such cases the easiest solution is to build one LUN over multiple VHDs using a software RAID.
- You don't know what the exact I/O workload per data file is and only roughly know what the overall IOPS workload against the DBMS is. Easiest to do is to build one LUN with the help of a software RAID. The sum of quotas of multiple VHDs behind this LUN should then fulfill the known IOPS rate.

Windows

Usage of Windows Server 2012 or higher Storage Spaces is preferable since it is more efficient than Windows Striping of earlier Windows versions. Please be aware that you might need to create the Windows Storage Pools and Storage Spaces by PowerShell commands when using Windows Server 2012 as Operating System. The PowerShell commands can be found here <https://technet.microsoft.com/library/jj851254.aspx>

Linux

Only MDADM and LVM (Logical Volume Manager) are supported to build a software RAID on Linux. For more information, read the following articles:

- [Configure Software RAID on Linux](#) (for MDADM)
- [Configure LVM on a Linux VM in Azure](#)

Considerations for leveraging VM-series which are able to work with Azure Premium Storage usually are:

- Demands for I/O latencies that are close to what SAN/NAS devices deliver.
- Demand for factors better I/O latency than Azure Standard Storage can deliver.
- Higher IOPS per VM than what could be achieved with multiple Standard Storage VHDs against a certain VM type.

Since the underlying Azure Storage replicates each VHD to at least three storage nodes, simple RAID 0 striping can be used. There is no need to implement RAID5 or RAID1.

Microsoft Azure Storage

Microsoft Azure Storage will store the base VM (with OS) and VHDs or BLOBs to at least 3 separate storage nodes. When creating a storage account, there is a choice of protection as shown here:

| P Premium Locally Re... | L Locally Redundant | G Geo-Redundant | R Read-Access Geo-R... | Z Zone Redundant |
|---|---|---|---|---|
| 3 Local replicas | 3 Local replicas | 3 Local replicas | 3 Local replicas | 3 Replicas across multip... |
|  Page blob
 5000 Max IOPS per disk
 99.9% SLA |  Block and page blobs
 Table
 Queue
 500 Max IOPS per disk
 99.9% SLA |  Block and page blobs
 Table
 Queue
 500 Max IOPS per disk
 99.9% SLA |  Block and page blobs
 Table
 Queue
 500 Max IOPS per disk
 99.9% SLA |  Block blob
 99.9% SLA
 Read access to sec... |

Azure Storage Local Replication (Locally Redundant) provides levels of protection against data loss due to infrastructure failure that few customers could afford to deploy. As shown above there are 4 different options with a fifth being a variation of one of the first three. Looking closer at them we can distinguish:

- **Premium Locally Redundant Storage (LRS):** Azure Premium Storage delivers high-performance, low-latency disk support for virtual machines running I/O-intensive workloads. There are 3 replicas of the data

within the same Azure datacenter of an Azure region. The copies will be in different Fault and Upgrade Domains (for concepts see [this chapter](#) in the [Planning Guide](#)). In case of a replica of the data going out of service due to a storage node failure or disk failure, a new replica is generated automatically.

- **Locally Redundant Storage (LRS):** In this case there are 3 replicas of the data within the same Azure datacenter of an Azure region. The copies will be in different Fault and Upgrade Domains (for concepts see [this chapter](#) in the [Planning Guide](#)). In case of a replica of the data going out of service due to a storage node failure or disk failure, a new replica is generated automatically.
- **Geo Redundant Storage (GRS):** In this case there is an asynchronous replication that will feed an additional 3 replicas of the data in another Azure Region which is in most of the cases in the same geographical region (like North Europe and West Europe). This will result in 3 additional replicas, so that there are 6 replicas in sum. A variation of this is an addition where the data in the geo replicated Azure region can be used for read purposes (Read-Access Geo-Redundant).
- **Zone Redundant Storage (ZRS):** In this case the 3 replicas of the data remain in the same Azure Region. As explained in [this chapter](#) of the [Planning Guide](#) an Azure region can be a number of datacenters in close proximity. In the case of LRS the replicas would be distributed over the different datacenters that make one Azure region.

More information can be found [here](#).

NOTE

For DBMS deployments, the usage of Geo Redundant Storage is not recommended

Azure Storage Geo-Replication is asynchronous. Replication of individual VHDs mounted to a single VM are not synchronized in lock step. Therefore, it is not suitable to replicate DBMS files that are distributed over different VHDs or deployed against a software RAID based on multiple VHDs. DBMS software requires that the persistent disk storage is precisely synchronized across different LUNs and underlying disks/VHDs/spindles. DBMS software uses various mechanisms to sequence IO write activities and a DBMS will report that the disk storage targeted by the replication is corrupted if these vary even by a few milliseconds. Hence if one really wants a database configuration with a database stretched across multiple VHDs geo-replicated, such a replication needs to be performed with database means and functionality. One should not rely on Azure Storage Geo-Replication to perform this job.

The problem is simplest to explain with an example system. Let's assume you have an SAP system uploaded into Azure which has 8 VHDs containing data files of the DBMS plus one VHD containing the transaction log file. Each one of these 9 VHDs will have data written to them in a consistent method according to the DBMS, whether the data is being written to the data or transaction log files.

In order to properly geo-replicate the data and maintain a consistent database image, the content of all nine VHDs would have to be geo-replicated in the exact order the I/O operations were executed against the nine different VHDs. However, Azure Storage geo-replication does not allow to declare dependencies between VHDs. This means Microsoft Azure Storage geo-replication doesn't know about the fact that the content in these nine different VHDs are related to each other and that the data changes are consistent only when replicating in the order the I/O operations happened across all the 9 VHDs.

Besides chances being high that the geo-replicated images in the scenario do not provide a consistent database image, there also is a performance penalty that shows up with geo redundant storage that can severely impact performance. In summary do not use this type of storage redundancy for DBMS type workloads.

Mapping VHDs into Azure Virtual Machine Service Storage Accounts

An Azure Storage Account is not only an administrative construct, but also a subject of limitations. Whereas the limitations vary on whether we talk about an Azure Standard Storage Account or an Azure Premium Storage Account. The exact capabilities and limitations are listed [here](#)

So for Azure Standard Storage it is important to note there is a limit on the IOPS per storage account (Row containing 'Total Request Rate' in [the article](#)). In addition, there is an initial limit of 100 Storage Accounts per Azure subscription (as of July 2015). Therefore, it is recommended to balance IOPS of VMs between multiple

storage accounts when using Azure Standard Storage. Whereas a single VM ideally uses one storage account if possible. So if we talk about DBMS deployments where each VHD that is hosted on Azure Standard Storage could reach its quota limit, you should only deploy 30-40 VHDs per Azure Storage Account that uses Azure Standard Storage. On the other hand, if you leverage Azure Premium Storage and want to store large database volumes, you might be fine in terms of IOPS. But an Azure Premium Storage Account is way more restrictive in data volume than an Azure Standard Storage Account. As a result, you can only deploy a limited number of VHDs within an Azure Premium Storage Account before hitting the data volume limit. At the end think of an Azure Storage Account as a "Virtual SAN" that has limited capabilities in IOPS and/or capacity. As a result, the task remains, as in on-premises deployments, to define the layout of the VHDs of the different SAP systems over the different 'imaginary SAN devices' or Azure Storage Accounts.

For Azure Standard Storage it is not recommended to present storage from different storage accounts to a single VM if possible.

Whereas using the DS or GS-series of Azure VMs it is possible to mount VHDs out of Azure Standard Storage Accounts and Premium Storage Accounts. Use cases like writing backups into Standard Storage backed VHDs whereas having DBMS data and log files on Premium Storage come to mind where such heterogeneous storage could be leveraged.

Based on customer deployments and testing around 30 to 40 VHDs containing database data files and log files can be provisioned on a single Azure Standard Storage Account with acceptable performance. As mentioned earlier, the limitation of an Azure Premium Storage Account is likely to be the data capacity it can hold and not IOPS.

As with SAN devices on-premises, sharing requires some monitoring in order to eventually detect bottlenecks on an Azure Storage Account. The Azure Monitoring Extension for SAP and the Azure Portal are tools that can be used to detect busy Azure Storage Accounts that may be delivering suboptimal IO performance. If this situation is detected it is recommended to move busy VMs to another Azure Storage Account. Please refer to the [Deployment Guide](#) for details on how to activate the SAP host monitoring capabilities.

Another article summarizing best practices around Azure Standard Storage and Azure Standard Storage Accounts can be found here <https://blogs.msdn.com/b/mast/archive/2014/10/14/configuring-azure-virtual-machines-for-optimal-storage-performance.aspx>

Moving deployed DBMS VMs from Azure Standard Storage to Azure Premium Storage

We encounter quite some scenarios where you as customer want to move a deployed VM from Azure Standard Storage into Azure Premium Storage. This is not possible without physically moving the data. There are several ways to achieve the goal:

- You could simply copy all VHDs, base VHD as well as data VHDs into a new Azure Premium Storage Account. Very often you chose the number of VHDs in Azure Standard Storage not because of the fact that you needed the data volume. But you needed that many VHDs because of the IOPS. Now that you move to Azure Premium Storage you could go with way less VHDs to achieve the same IOPS throughput. Given the fact that in Azure Standard Storage you pay for the used data and not the nominal disk size, the number of VHDs did not really matter in terms of costs. However, with Azure Premium Storage, you would pay for the nominal disk size. Therefore, most of the customer try to keep the number of Azure VHDs in Premium Storage at the number needed to achieve the IOPS throughput necessary. So, most customers decide against the way of a simple 1:1 copy.
- If not yet mounted, you mount a single VHD that can contain a database backup of your SAP database. After the backup, you unmount all VHDs including the VHD containing the backup and copy the base VHD and the VHD with the backup into an Azure Premium Storage account. You would then deploy the VM based on the base VHD and mount the VHD with the backup. Now you create additional empty Premium Storage Disks for the VM that are used to restore the database into. This assumes that the DBMS allows you to change paths to the data and log files as part of the restore process.
- Another possibility is a variation of the former process, where you just copy the backup VHD into Azure

Premium Storage and attach it against a VM that you newly deployed and installed.

- The fourth possibility you would choose when you are in need to change the number of data files of your database. In such a case you would perform a SAP homogenous system copy using export/import. Put those export files into a VHD that is copied into an Azure Premium Storage Account and attach it to a VM that you use to run the import processes. Customers use this possibility mainly when they want to decrease the number of data files.

Deployment of VMs for SAP in Azure

Microsoft Azure offers multiple ways to deploy VMs and associated disks. Thereby it is very important to understand the differences since preparations of the VMs might differ dependent on the way of deployment. In general, we look into the scenarios described in the following chapters.

Deploying a VM from the Azure Marketplace

You like to take a Microsoft or 3rd party provided image from the Azure Marketplace to deploy your VM. After you deployed your VM in Azure, you follow the same guidelines and tools to install the SAP software inside your VM as you would do in an on-premises environment. For installing the SAP software inside the Azure VM, SAP and Microsoft recommend to upload and store the SAP installation media in Azure VHDs or to create an Azure VM working as a 'File server' which contains all the necessary SAP installation media.

Deploying a VM with a customer specific generalized image

Due to specific patch requirements in regards to your OS or DBMS version, the provided images in the Azure Marketplace might not fit your needs. Therefore, you might need to create a VM using your own 'private' OS/DBMS VM image which can be deployed several times afterwards. To prepare such a 'private' image for duplication, the OS must be generalized on the on-premises VM. Please refer to the [Deployment Guide](#) for details on how to generalize a VM.

If you have already installed SAP content in your on-premises VM (especially for 2-Tier systems), you can adapt the SAP system settings after the deployment of the Azure VM through the instance rename procedure supported by the SAP Software Provisioning Manager (SAP Note [1619720](#)). Otherwise you can install the SAP software later after the deployment of the Azure VM.

As of the database content used by the SAP application, you can either generate the content freshly by an SAP installation or you can import your content into Azure by using a VHD with a DBMS database backup or by leveraging capabilities of the DBMS to directly backup into Microsoft Azure Storage. In this case, you could also prepare VHDs with the DBMS data and log files on-premises and then import those as Disks into Azure. But the transfer of DBMS data which is getting loaded from on-premises to Azure would work over VHD disks that need to be prepared on-premises.

Moving a VM from on-premises to Azure with a non-generalized disk

You plan to move a specific SAP system from on-premises to Azure (lift and shift). This can be done by uploading the VHD which contains the OS, the SAP binaries and eventual DBMS binaries plus the VHDs with the data and log files of the DBMS to Azure. In opposite to scenario #2 above, you keep the hostname, SAP SID and SAP user accounts in the Azure VM as they were configured in the on-premises environment. Therefore, generalizing the image is not necessary. This case will mostly apply for Cross-Premises scenarios where a part of the SAP landscape is run on-premises and parts on Azure.

High Availability and Disaster Recovery with Azure VMs

Azure offers the following High Availability (HA) and Disaster Recovery (DR) functionalities which apply to different components we would use for SAP and DBMS deployments

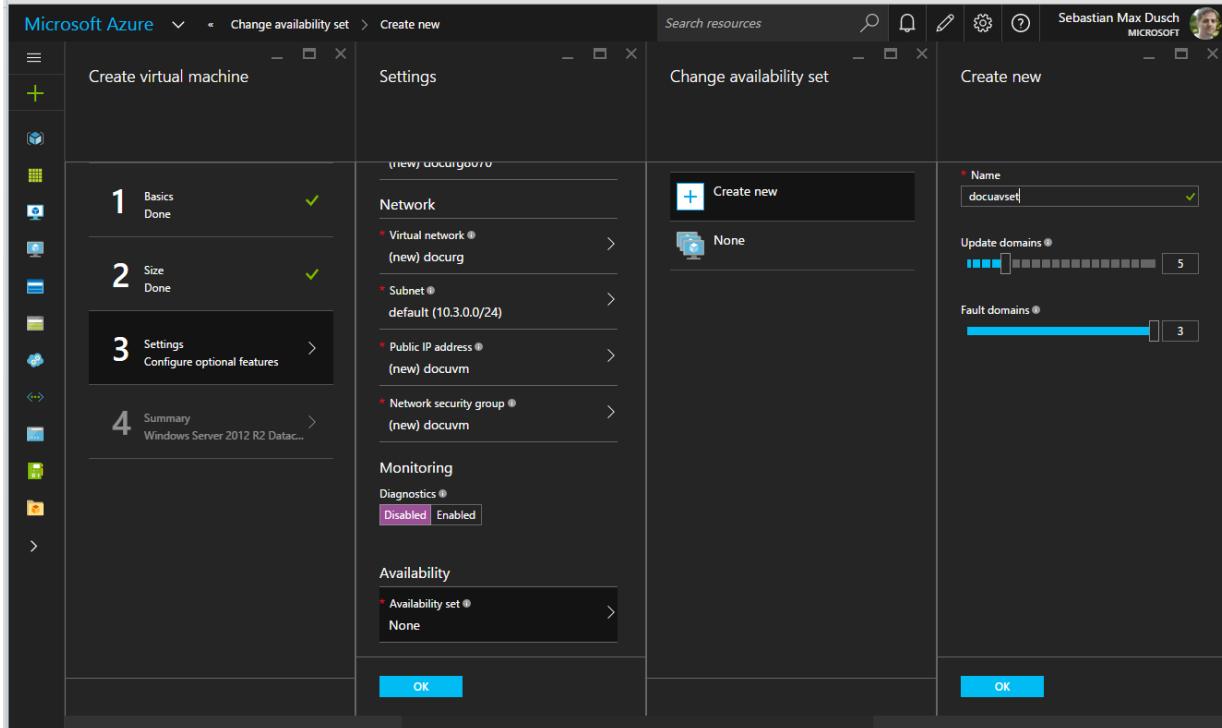
VMs deployed on Azure Nodes

The Azure Platform does not offer features such as Live Migration for deployed VMs. This means if there is maintenance necessary on a server cluster on which a VM is deployed, the VM needs to get stopped and restarted. Maintenance in Azure is performed using so called Upgrade Domains within clusters of servers. Only

one Upgrade Domain at a time is being maintained. During such a restart there will be an interruption of service while the VM is shut down, maintenance is performed and VM restarted. Most DBMS vendors however provide High Availability and Disaster Recovery functionality that will quickly restart the DBMS services on another node if the primary node is unavailable. The Azure Platform offers functionality to distribute VMs, Storage and other Azure services across Upgrade Domains to ensure that planned maintenance or infrastructure failures would only impact a small subset of VMs or services. With careful planning it is possible to achieve availability levels comparable to on-premises infrastructures.

Microsoft Azure Availability Sets are a logical grouping of VMs or Services that ensures VMs and other services are distributed to different Fault and Upgrade Domains within a cluster such that there would only be one node shutdown at any one point in time (read [this article](#) for more details).

It needs to be configured by purpose when rolling out VMs as seen here:



If we want to create highly available configurations of DBMS deployments (independent of the individual DBMS HA functionality used), the DBMS VMs would need to:

- Add the VMs to the same Azure Virtual Network (<https://azure.microsoft.com/documentation/services/virtual-network/>)
- The VMs of the HA configuration should also be in the same subnet. Name resolution between the different subnets is not possible in Cloud-Only deployments, only IP resolution will work. Using site-to-site or ExpressRoute connectivity for Cross-Premises deployments, a network with at least one subnet will be already established. Name resolution will be done according to the on-premises AD policies and network infrastructure.

IP Addresses

It is highly recommended to setup the VMs for HA configurations in a resilient way. Relying on IP addresses to address the HA partner(s) within the HA configuration is not reliable in Azure unless static IP addresses are used. There are two "Shutdown" concepts in Azure:

- Shut down through Azure Portal or Azure PowerShell cmdlet Stop-AzureRmVM: In this case the Virtual Machine gets shutdown and de-allocated. Your Azure account will no longer be charged for this VM so the only charges that will incur are for the storage used. However, if the private IP address of the network interface was not static, the IP address is released and it is not guaranteed that the network interface gets the old IP address assigned again after a restart of the VM. Performing the shut down through the Azure

Portal or by calling Stop-AzureRmVM will automatically cause de-allocation. If you do not want to deallocate the machine use Stop-AzureRmVM -StayProvisioned

- If you shut down the VM from an OS level, the VM gets shut down and NOT de-allocated. However, in this case, your Azure account will still be charged for the VM, despite the fact that it is shutdown. In such a case, the assignment of the IP address to a stopped VM will remain intact. Shutting down the VM from within will not automatically force de-allocation.

Even for Cross-Premises scenarios, by default a shutdown and de-allocation will mean de-assignment of the IP addresses from the VM, even if on-premises policies in DHCP settings are different.

- The exception is if one assigns a static IP address to a network interface as described [here](#).
- In such a case the IP address remains fixed as long as the network interface is not deleted.

IMPORTANT

In order to keep the whole deployment simple and manageable, the clear recommendation is to setup the VMs partnering in a DBMS HA or DR configuration within Azure in a way that there is a functioning name resolution between the different VMs involved.

Deployment of Host Monitoring

For productive usage of SAP Applications in Azure Virtual Machines, SAP requires the ability to get host monitoring data from the physical hosts running the Azure Virtual Machines. A specific SAP HostAgent patch level will be required that enables this capability in SAPOS COL and SAP HostAgent. The exact patch level is documented in SAP Note [1409604](#).

For the details regarding deployment of components that deliver host data to SAPOS COL and SAPHostAgent and the lifecycle management of those components, please refer to the [Deployment Guide](#)

Specifics to Microsoft SQL Server

SQL Server IaaS

Starting with Microsoft Azure, you can easily migrate your existing SQL Server applications built on Windows Server platform to Azure Virtual Machines. SQL Server in a Virtual Machine enables you to reduce the total cost of ownership of deployment, management and maintenance of enterprise breadth applications by easily migrating these applications to Microsoft Azure. With SQL Server in an Azure Virtual Machine, administrators and developers can still use the same development and administration tools that are available on-premises.

IMPORTANT

Please note we are not discussing Microsoft Azure SQL Database which is a Platform as a Service offer of the Microsoft Azure Platform. The discussion in this paper is about running the SQL Server product as it is known for on-premises deployments in Azure Virtual Machines, leveraging the Infrastructure as a Service capability of Azure. Database capabilities and functionalities between these two offers are different and should not be mixed up with each other. See also: <https://azure.microsoft.com/services/sql-database/>

It is strongly recommended to review [this](#) documentation before continuing.

In the following sections pieces of parts of the documentation under the link above will be aggregated and mentioned. Specifics around SAP are mentioned as well and some concepts are described in more detail. However, it is highly recommended to work through the documentation above first before reading the SQL Server specific documentation.

There is some SQL Server in IaaS specific information you should know before continuing:

- **Virtual Machine SLA:** There is an SLA for Virtual Machines running in Azure which can be found here: <https://azure.microsoft.com/support/legal/sla/>
- **SQL Version Support:** For SAP customers, we support SQL Server 2008 R2 and higher on Microsoft Azure Virtual Machine. Earlier editions are not supported. Review this general [Support Statement](#) for more details. Please note that in general SQL Server 2008 is supported by Microsoft as well. However due to significant functionality for SAP which was introduced with SQL Server 2008 R2, SQL Server 2008 R2 is the minimum release for SAP. Keep in mind that SQL Server 2012 and 2014 got extended with deeper integration into the IaaS scenario (like backing up directly against Azure Storage). Therefore, we restrict this paper to SQL Server 2012 and 2014 with its latest patch level for Azure.
- **SQL Feature Support:** Most SQL Server features are supported on Microsoft Azure Virtual Machines with some exceptions. **SQL Server Failover Clustering using Shared Disks is not supported.** Distributed technologies like Database Mirroring, AlwaysOn Availability Groups, Replication, Log Shipping and Service Broker are supported within a single Azure Region. SQL Server AlwaysOn also is supported between different Azure Regions as documented here: <https://blogs.technet.com/b/dataplatforminsider/archive/2014/06/19/sql-server-alwayson-availability-groups-supported-between-microsoft-azure-regions.aspx>. Review the [Support Statement](#) for more details. An example on how to deploy an AlwaysOn configuration is shown in [this](#) article. Also, check out the Best Practices documented [here](#)
- **SQL Performance:** We are confident that Microsoft Azure hosted Virtual Machines will perform very well in comparison to other public cloud virtualization offerings, but individual results may vary. Check out [this](#) article.
- **Using Images from Azure Marketplace:** The fastest way to deploy a new Microsoft Azure VM is to use an image from the Azure Marketplace. There are images in the Azure Marketplace which contain SQL Server. The images where SQL Server already is installed can't be immediately used for SAP NetWeaver applications. The reason is the default SQL Server collation is installed within those images and not the collation required by SAP NetWeaver systems. In order to use such images, please check the steps documented in chapter [Using a SQL Server images out of the Microsoft Azure Marketplace](#).
- Check out [Pricing Details](#) for more information. The [SQL Server 2012 Licensing Guide](#) and [SQL Server 2014 Licensing Guide](#) are also an important resource.

SQL Server configuration guidelines for SAP related SQL Server installations in Azure VMs

Recommendations on VM/VHD structure for SAP related SQL Server deployments

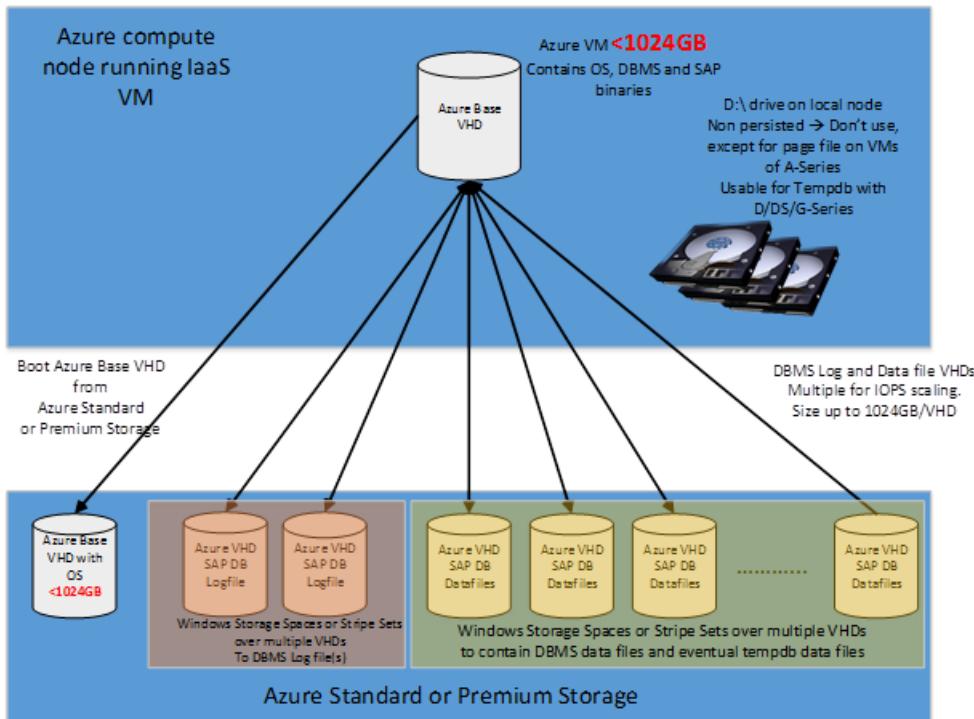
In accordance with the general description, SQL Server executables should be located or installed into the system drive of the VM's base VHD (drive C:). Typically, most of the SQL Server system databases are not utilized at a high level by SAP NetWeaver workload. Hence the system databases of SQL Server (master, msdb, and model) can remain on the C:\ drive as well. An exception could be tempdb, which in the case of some SAP ERP and all BW workloads, might require either higher data volume or I/O operations volume which can't fit into the original VM. For such systems, the following steps should be performed:

- Move the primary tempdb data file(s) to the same logical drive as the primary data file(s) of the SAP database.
- Add any additional tempdb data files to each of the other logical drives containing a data file of the SAP user database.
- Add the tempdb logfile to the logical drive which contains the user database's log file.
- **Exclusively for VM types that use local SSDs** on the compute node tempdb data and log files might be placed on the D:\ drive. Nevertheless, it might be recommended to use multiple tempdb data files. Be aware D:\ drive volumes are different based on the VM type.

These configurations enable tempdb to consume more space than the system drive is able to provide. In order to determine the proper tempdb size, one can check the tempdb sizes on existing systems which run on-premises. In addition, such a configuration would enable IOPS numbers against tempdb which cannot be provided with the system drive. Again, systems that are running on-premises can be used to monitor I/O

workload against tempdb so that you can derive the IOPS numbers you expect to see on your tempdb.

A VM configuration which runs SQL Server with a SAP database and where tempdb data and tempdb logfile are placed on the D:\ drive would look like:



Be aware that the D:\ drive has different sizes dependent on the VM type. Dependent on the size requirement of tempdb you might be forced to pair tempdb data and log files with the SAP database data and log files in cases where D:\ drive is too small.

Formatting the VHDs

For SQL Server the NTFS block size for VHDs containing SQL Server data and log files should be 64K. There is no need to format the D:\ drive. This drive comes pre-formatted.

In order to make sure that the restore or creation of databases is not initializing the data files by zeroing the content of the files, one should make sure that the user context the SQL Server service is running in has a certain permission. Usually users in the Windows Administrator group have these permissions. If the SQL Server service is run in the user context of non-Windows Administrator user, you need to assign that user the User Right 'Perform volume maintenance tasks'. See the details in this Microsoft Knowledge Base Article:
<https://support.microsoft.com/kb/2574695>

Impact of database compression

In configurations where I/O bandwidth can become a limiting factor, every measure which reduces IOPS might help to stretch the workload one can run in an IaaS scenario like Azure. Therefore, if not yet done, applying SQL Server PAGE compression is strongly recommended by both SAP and Microsoft before uploading an existing SAP databases to Azure.

The recommendation to perform Database Compression before uploading to Azure is given out of two reasons:

- The amount of data to be uploaded is lower.
- The duration of the compression execution is shorter assuming that one can use stronger hardware with more CPUs or higher I/O bandwidth or less I/O latency on-premises.
- Smaller database sizes might lead to less costs for disk allocation

Database compression works as well in an Azure Virtual Machines as it does on-premises. For more details on how to compress an existing SAP SQL Server database please check here:

<https://blogs.msdn.com/b/saponsqlserver/archive/2010/10/08/compressing-an-sap-database-using-report-msscompress.aspx>

SQL Server 2014 – Storing Database Files directly on Azure Blob Storage

SQL Server 2014 opens the possibility to store database files directly on Azure Blob Store without the ‘wrapper’ of a VHD around them. Especially with using Standard Azure Storage or smaller VM types this enables scenarios where you can overcome the limits of IOPS that would be enforced by a limited number of VHDs that can be mounted to some smaller VM types. This works for user databases however not for system databases of SQL Server. It also works for data and log files of SQL Server. If you’d like to deploy a SAP SQL Server database this way instead of ‘wrapping’ it into VHDs, please keep the following in mind:

- The Storage Account used needs to be in the same Azure Region as the one that is used to deploy the VM SQL Server is running in.
- Considerations listed earlier in regards to distribute VHDs over different Azure Storage Accounts apply for this method of deployments as well. Means the I/O operations count against the limits of the Azure Storage Account.

Details about this type of deployment are listed here: <https://msdn.microsoft.com/library/dn385720.aspx>

In order to store SQL Server data files directly on Azure Premium Storage, you need to have a minimum SQL Server 2014 patch release which is documented here: <https://support.microsoft.com/kb/3063054>. Storing SQL Server data files on Azure Standard Storage does work with the released version of SQL Server 2014. However, the very same patches contain another series of fixes which make the direct usage of Azure Blob Storage for SQL Server data files and backups more reliable. Therefore we recommend to use these patches in general.

SQL Server 2014 Buffer Pool Extension

SQL Server 2014 introduced a new feature which is called Buffer Pool Extension. This functionality extends the buffer pool of SQL Server which is kept in memory with a second level cache that is backed by local SSDs of a server or VM. This enables to keep a larger working set of data ‘in memory’. Compared to accessing Azure Standard Storage the access into the extension of the buffer pool which is stored on local SSDs of an Azure VM is many factors faster. Therefore, leveraging the local D:\ drive of the VM types that have excellent IOPS and throughput could be a very reasonable way to reduce the IOPS load against Azure Storage and improve response times of queries dramatically. This applies especially when not using Premium Storage. In case of Premium Storage and the usage of the Premium Azure Read Cache on the compute node, as recommended for data files, no big differences are expected. Reason is that both caches (SQL Server Buffer Pool Extension and Premium Storage Read Cache) are using the local disks of the compute nodes. For more details about this functionality, please check this documentation: <https://msdn.microsoft.com/library/dn133176.aspx>

Backup/Recovery considerations for SQL Server

When deploying SQL Server into Azure your backup methodology must be reviewed. Even if the system is not a productive system, the SAP database hosted by SQL Server must be backed up periodically. Since Azure Storage keeps three images, a backup is now less important in respect to compensating a storage crash. The priority reason for maintaining a proper backup and recovery plan is more that you can compensate for logical/manual errors by providing point in time recovery capabilities. So the goal is to either use backups to restore the database back to a certain point in time or to use the backups in Azure to seed another system by copying the existing database. For example, you could transfer from a 2-Tier SAP configuration to a 3-Tier system setup of the same system by restoring a backup.

There are three different ways to backup SQL Server to Azure Storage:

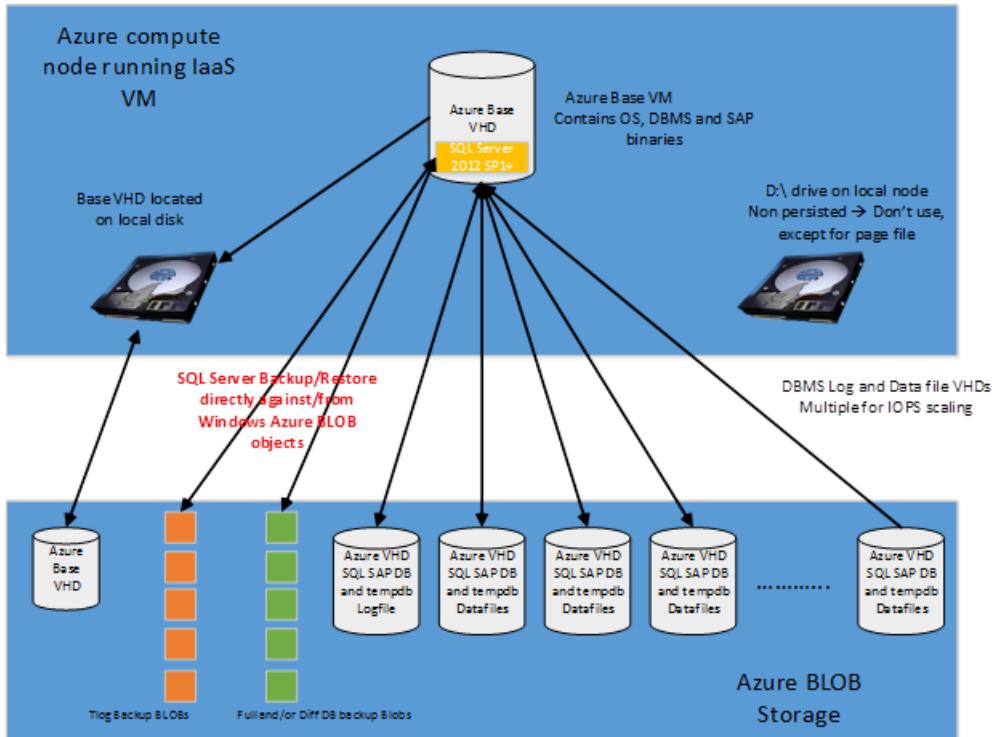
1. SQL Server 2012 CU4 and higher can natively backup databases to a URL. This is detailed in the blog [New functionality in SQL Server 2014 – Part 5 – Backup/Restore Enhancements](#). See chapter [SQL Server 2012 SP1 CU4 and later](#).
2. SQL Server releases prior to SQL 2012 CU4 can use a redirection functionality to backup to a VHD and basically move the write stream towards an Azure Storage location that has been configured. See chapter

SQL Server 2012 SP1 CU3 and earlier releases.

3. The final method is to perform a conventional SQL Server backup to disk command onto a VHD disk device.
This is identical to the on-premises deployment pattern and is not discussed in detail in this document.

SQL Server 2012 SP1 CU4 and later

This functionality allows you to directly backup to Azure BLOB storage. Without this method, you must backup to other Azure VHDs which would consume VHD and IOPS capacity. The idea is basically this:



The advantage in this case is that one doesn't need to spend VHDs to store SQL Server backups on. So you have fewer VHDs allocated and the whole bandwidth of VHD IOPS can be used for data and log files. Please note that the maximum size of a backup is limited to a maximum of 1 TB as documented in the section 'Limitations' in this article: <https://msdn.microsoft.com/library/dn435916.aspx#limitations>. If the backup size, despite using SQL Server Backup compression would exceed 1 TB in size, the functionality described in chapter [SQL Server 2012 SP1 CU3 and earlier releases](#) in this document needs to be used.

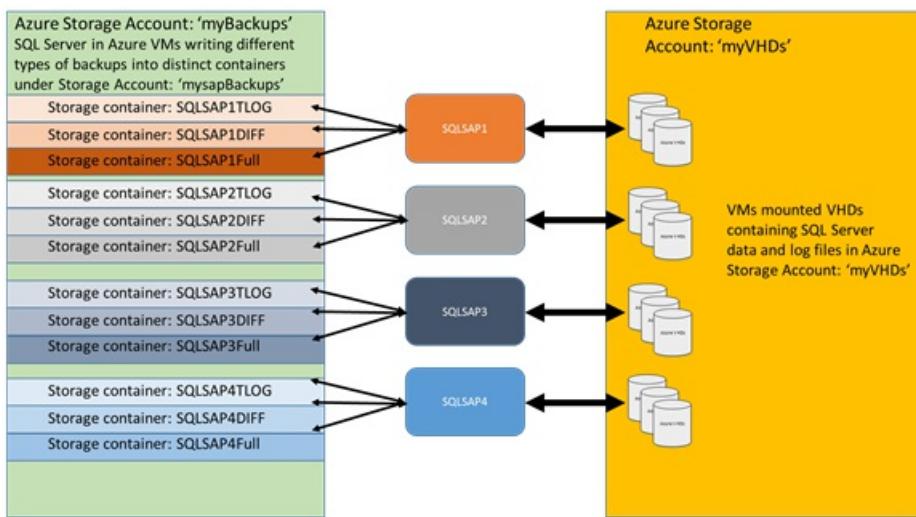
Related documentation describing the restore of databases from backups against Azure Blob Store recommend not to restore directly from Azure BLOB store if the backups is >25GB. The recommendation in this article is simply based on performance considerations and not due to functional restrictions. Therefore, different conditions may apply on a case by case basis.

Documentation on how this type of backup is set up and leveraged can be found in [this](#) tutorial

An example of the sequence of steps can be read [here](#).

Automating backups, it is of highest importance to make sure that the BLOBS for each backup are named differently. Otherwise they will be overwritten and the restore chain is broken.

In order not to mix up things between the 3 different types of backups, it is advisable to create different containers underneath the storage account used for backups. The containers could be by VM only or by VM and Backup type. The schema could look like:



In the example above, the backups would not be performed into the same storage account where the VMs are deployed. There would be a new storage account specifically for the backups. Within the storage accounts, there would be different containers created with a matrix of the type of backup and the VM name. Such segmentation will make it easier to administrate the backups of the different VMs.

The BLOBS one directly writes the backups to, are not adding to the count of the VHDs of a VM. Hence one could maximize the maximum of VHDs mounted of the specific VM SKU for the data and transaction log file and still execute a backup against a storage container.

SQL Server 2012 SP1 CU3 and earlier releases

The first step you must perform in order to achieve a backup directly against Azure Storage would be to download the msi which is linked to [this KBA article](#).

Download the x64 installation file and the documentation. The file will install a program called: 'Microsoft SQL Server Backup to Microsoft Azure Tool'. Read the documentation of the product thoroughly. The tool basically works in the following way:

- From the SQL Server side, a disk location for the SQL Server backup is defined (don't use the D:\ drive for this).
- The tool will allow you to define rules which can be used to direct different types of backups to different Azure Storage containers.
- Once the rules are in place, the tool will redirect the write stream of the backup to one of the VHDs/disks to the Azure Storage location which was defined earlier.
- The tool will leave a small stub file of a few KB size on the VHD/Disk which was defined for the SQL Server backup. **This file should be left on the storage location since it is required to restore again from Azure Storage.**
 - If you have lost the stub file (e.g. through loss of the storage media that contained the stub file) and you have chosen the option of backing up to a Microsoft Azure Storage account, you may recover the stub file through Microsoft Azure Storage by downloading it from the storage container in which it was placed. You should then place the stub file into a folder on the local machine where the Tool is configured to detect and upload to the same container with the same encryption password if encryption was used with the original rule.

This means the schema as described above for more recent releases of SQL Server can be put in place as well for SQL Server releases which are not allowing direct address an Azure Storage location.

This method should not be used with more recent SQL Server releases which support backing up natively against Azure Storage. Exceptions are where limitations of the native backup into Azure are blocking native backup execution into Azure.

Other possibilities to backup SQL Server databases

Other possibilities to backup databases is to attach additional VHDs to a VM that you use to store backups on. In such a case you would need to make sure that the VHDs are not running full. If that is the case, you would need to unmount the VHD and so to speak 'archive' it and replace it with a new empty VHD. If you go down that path, you want to keep these VHDs in separate Azure Storage Accounts from the ones that the VHDs with the database files.

A second possibility is to use a large VM that can have many VHDs attached. E.g. D14 with 32VHDs. Use Storage Spaces to build a flexible environment where you could build shares that are used then as backup targets for the different DBMS servers.

Some best practices got documented [here](#) as well.

Performance considerations for backups/restores

As in bare-metal deployments, backup/restore performance is dependent on how many volumes can be read in parallel and what the throughput of those volumes might be. In addition, the CPU consumption used by backup compression may play a significant role on VMs with just up to 8 CPU threads. Therefore, one can assume:

- The fewer the number of VHDs used to store the data files, the smaller the overall throughput in reading.
- The smaller the number of CPU threads in the VM, the more severe the impact of backup compression.
- The fewer targets (BLOBs or VHDs) to write the backup to, the lesser the throughput.
- The smaller the VM size, the smaller the storage throughput quota writing and reading from Azure Storage. Independent of whether the backups are directly stored on Azure Blob or whether they are stored in VHDs that again are stored in Azure Blobs.

When using a Microsoft Azure Storage BLOB as the backup target in more recent releases, you are restricted to designating only one URL target for each specific backup.

But when using the 'Microsoft SQL Server Backup to Microsoft Azure Tool' in older releases, you can define more than one file target. With more than one target, the backup can scale and the throughput of the backup is higher. This would result then in multiple files as well in the Azure Storage account. In our testing, using multiple file destinations one can definitely achieve the throughput which one could achieve with the backup extensions implemented in from SQL Server 2012 SP1 CU4 on. You also are not blocked by the 1TB limit as in the native backup into Azure.

However, keep in mind, the throughput also is dependent on the location of the Azure Storage Account you use for the backup. An idea might be to locate the storage account in a different region than the VMs are running in. E.g. you would run the VM configuration in West-Europe, but put the Storage Account that you use to back up against in North-Europe. That certainly will have impact on the backup throughput and is not likely to generate a throughput of 150MB/sec as it seems to be possible in cases where the target storage and the VMs are running in the same regional datacenter.

Managing Backup BLOBS

There is a requirement to manage the backups on your own. Since the expectation is that many blobs will be created by executing frequent transaction log backups, administration of those blobs easily can overburden the Azure Portal. Therefore, it is recommendable to leverage a Azure Storage Explorer. There are several good ones available which can help to manage an Azure storage account

- Microsoft Visual Studio with Azure SDK installed (<https://azure.microsoft.com/downloads/>)
- Microsoft Azure Storage Explorer (<https://azure.microsoft.com/downloads/>)
- 3rd party tools

Using a SQL Server images out of the Microsoft Azure Marketplace

Microsoft offers VMs in the Azure Marketplace which already contain versions of SQL Server. For SAP customers who require licenses for SQL Server and Windows, this might be an opportunity to basically cover the need for licenses by spinning up VMs with SQL Server already installed. In order to use such images for

SAP, the following considerations need to be made:

- The SQL Server non-Evaluation versions acquire higher costs than just a 'Windows-only' VM deployed from Azure Marketplace. Please see these articles to compare prices:
<https://azure.microsoft.com/pricing/details/virtual-machines/> and
<https://azure.microsoft.com/pricing/details/virtual-machines/#Sql>.
- You only can use SQL Server releases which are supported by SAP, like SQL Server 2012.
- The collation of the SQL Server instance which is installed in the VMs offered in the Azure Marketplace is not the collation SAP NetWeaver requires the SQL Server instance to run. You can change the collation though with the directions in the following section.

Changing the SQL Server Collation of a Microsoft Windows/SQL Server VM

Since the SQL Server images in the Azure Marketplace are not set up to use the collation which is required by SAP NetWeaver applications, it needs to be changed immediately after the deployment. For SQL Server 2012, this can be done with the following steps as soon as the VM has been deployed and an administrator is able to log into the deployed VM:

- Open a Windows Command Window 'as administrator'.
- Change the directory to C:\Program Files\Microsoft SQL Server\110\Setup Bootstrap\SQLServer2012.
- Execute the command: Setup.exe /QUIET /ACTION=REBUILDDATABASE /INSTANCENAME=MSSQLSERVER /SQLSYSADMINACCOUNTS=<local_admin_account_name>
/SQLCOLLATION=SQL_Latin1_General_Cp850_BIN2
 - <local_admin_account_name> is the account which was defined as the administrator account when deploying the VM for the first time through the gallery.

The process should only take a few minutes. In order to make sure whether the step ended up with the correct result, please perform the following steps:

- Open SQL Server Management Studio.
- Open a Query Window.
- Execute the command sp_helpsort in the SQL Server master database.

The desired result should look like:

```
Latin1-General, binary code point comparison sort for Unicode Data, SQL Server Sort Order 40 on Code Page 850
for non-Unicode Data
```

If this is not the result, STOP deploying SAP and investigate why the setup command did not work as expected. Deployment of SAP NetWeaver applications onto SQL Server instance with different SQL Server codepages than the one mentioned above is **NOT** supported.

SQL Server High-Availability for SAP in Azure

As mentioned earlier in this paper, there is no possibility to create shared storage which is necessary for the usage of the oldest SQL Server high availability functionality. This functionality would install two or more SQL Server instances in a Windows Server Failover Cluster (WSFC) using a shared disk for the user databases (and eventually tempdb). This is the long time standard high availability method which also is supported by SAP. Because Azure doesn't support shared storage, SQL Server high availability configurations with a shared disk cluster configuration cannot be realized. However, many other high availability methods are still possible and are described in the following sections.

SQL Server Log Shipping

One of the methods of high availability (HA) is SQL Server Log Shipping. If the VMs participating in the HA configuration have working name resolution, there is no problem and the setup in Azure will not differ from any setup that is done on-premises. It is not recommended to rely on IP resolution only. In regards to setting up Log Shipping and the principles around Log Shipping please check this documentation:

<https://technet.microsoft.com/library/ms187103.aspx>

In order to really achieve any high availability, one needs to deploy the VMs which are within such a Log Shipping configuration to be within the same Azure Availability Set.

Database Mirroring

Database Mirroring as supported by SAP (see SAP Note [965908](#)) relies on defining a failover partner in the SAP connection string. For the Cross-Premises cases, we assume that the two VMs are in the same domain and that the user context the two SQL Server instances are running under are domain users as well and have sufficient privileges in the two SQL Server instances involved. Therefore, the setup of Database Mirroring in Azure does not differ between a typical on-premises setup/configuration.

As of Cloud-Only deployments, the easiest method is to have another domain setup in Azure to have those DBMS VMs (and ideally dedicated SAP VMs) within one domain.

If a domain is not possible, one can also use certificates for the database mirroring endpoints as described here:

<https://technet.microsoft.com/library/ms191477.aspx>

A tutorial to set-up Database Mirroring in Azure can be found here:

<https://technet.microsoft.com/library/ms189852.aspx>

AlwaysOn

As AlwaysOn is supported for SAP on-premises (see SAP Note [1772688](#)), it is supported to be used in combination with SAP in Azure. The fact that you are not able to create shared disks in Azure doesn't mean that one can't create an AlwaysOn Windows Server Failover Cluster (WSFC) configuration between different VMs. It only means that you do not have the possibility to use a shared disk as a quorum in the cluster configuration. Hence you can build an AlwaysOn WSFC configuration in Azure and simply not select the quorum type that utilizes shared disk. The Azure environment those VMs are deployed in should resolve the VMs by name and the VMs should be in the same domain. This is true for Azure only and Cross-Premises deployments. There are some special considerations around deploying the SQL Server Availability Group Listener (not to be confused with the Azure Availability Set) since Azure at this point in time does not allow to simply create an AD/DNS object as it is possible on-premises. Therefore, some different installation steps are necessary to overcome the specific behavior of Azure.

Some considerations using an Availability Group Listener are:

- Using an Availability Group Listener is only possible with Windows Server 2012 or Windows Server 2012 R2 as guest OS of the VM. For Windows Server 2012 you need to make sure that this patch is applied:
<https://support.microsoft.com/kb/2854082>
- For Windows Server 2008 R2 this patch does not exist and AlwaysOn would need to be used in the same manner as Database Mirroring by specifying a failover partner in the connections string (done through the SAP default.pfl parameter dbs/mss/server – see SAP Note [965908](#)).
- When using an Availability Group Listener, the Database VMs need to be connected to a dedicated Load Balancer. Name resolution in Cloud-Only deployments would either require all VMs of an SAP system (application servers, DBMS server and (A)SCS server) are in the same virtual network or would require from an SAP application layer the maintenance of the etc\host file in order to get the VM names of the SQL Server VMs resolved. In order to avoid that Azure is assigning new IP addresses in cases where both VMs incidentally are shutdown, one should assign static IP addresses to the network interfaces of those VMs in the AlwaysOn configuration (defining a static IP address is described in [this article](#))
- There are special steps required when building the WSFC cluster configuration where the cluster needs a special IP address assigned, because Azure with its current functionality would assign the cluster name the same IP address as the node the cluster is created on. This means a manual step must be performed to assign a different IP address to the cluster.
- The Availability Group Listener is going to be created in Azure with TCP/IP endpoints which are assigned to the VMs running the primary and secondary replicas of the Availability group.

- There might be a need to secure these endpoints with ACLs.

It is possible to deploy a SQL Server AlwaysOn Availability Group over different Azure Regions as well. This functionality will leverage the Azure VNet-to-Vnet connectivity ([more details](#)).

Summary on SQL Server High Availability in Azure

Given the fact that Azure Storage is protecting the content, there is one less reason to insist on a hot-standby image. This means your High Availability scenario needs to only protect against the following cases:

- Unavailability of the VM as a whole due to maintenance on the server cluster in Azure or other reasons
- Software issues in the SQL Server instance
- Protecting from manual error where data gets deleted and point-in-time recovery is needed

Looking at matching technologies one can argue that the first two cases can be covered by Database Mirroring or AlwaysOn, whereas the third case only can be covered by Log-Shipping.

You will need to balance the more complex setup of AlwaysOn, compared to Database Mirroring, with the advantages of AlwaysOn. Those advantages can be listed like:

- Readable secondary replicas.
- Backups from secondary replicas.
- Better scalability.
- More than one secondary replicas.

General SQL Server for SAP on Azure Summary

There are many recommendations in this guide and we recommend you read it more than once before planning your Azure deployment. In general, though, be sure to follow the top ten general DBMS on Azure specific points:

1. Use the latest DBMS release, like SQL Server 2014, that has the most advantages in Azure. For SQL Server, this is SQL Server 2012 SP1 CU4 which would include the feature of backing up against Azure Storage. However, in conjunction with SAP we would recommend at least SQL Server 2014 SP1 CU1 or SQL Server 2012 SP2 and the latest CU.
2. Carefully plan your SAP system landscape in Azure to balance the data file layout and Azure restrictions:
 - Don't have too many VHDs, but have enough to ensure you can reach your required IOPS.
 - Remember that IOPS are also limited per Azure Storage Account and that Storage Accounts are limited within each Azure subscription ([more details](#)).
 - Only stripe across VHDs if you need to achieve a higher throughput.
3. Never install software or put any files that require persistence on the D:\ drive as it is non-permanent and anything on this drive will be lost at a Windows reboot.
4. Don't use Azure VHD caching for Azure Standard Storage.
5. Don't use Azure geo-replicated Storage Accounts. Use Locally Redundant for DBMS workloads.
6. Use your DBMS vendor's HA/DR solution to replicate database data.
7. Always use Name Resolution, don't rely on IP addresses.
8. Use the highest database compression possible. For SQL Server this is page compression.
9. Be careful using SQL Server images from the Azure Marketplace. If you use the SQL Server one, you must change the instance collation before installing any SAP NetWeaver system on it.
10. Install and configure the SAP Host Monitoring for Azure as described in [Deployment Guide](#).

Specifics to SAP ASE on Windows

Starting with Microsoft Azure, you can easily migrate your existing SAP ASE applications to Azure Virtual Machines. SAP ASE in a Virtual Machine enables you to reduce the total cost of ownership of deployment,

management and maintenance of enterprise breadth applications by easily migrating these applications to Microsoft Azure. With SAP ASE in an Azure Virtual Machine, administrators and developers can still use the same development and administration tools that are available on-premises.

There is an SLA for the Azure Virtual Machines which can be found here:

<https://azure.microsoft.com/support/legal/sla>

We are confident that Microsoft Azure hosted Virtual Machines will perform very well in comparison to other public cloud virtualization offerings, but individual results may vary. SAP sizing SAPS numbers of the different SAP certified VM SKUs will be provided in a separate SAP Note [1928533](#).

Statements and recommendations in regard to the usage of Azure Storage, Deployment of SAP VMs or SAP Monitoring apply to deployments of SAP ASE in conjunction with SAP applications as stated throughout the first four chapters of this document.

SAP ASE Version Support

SAP currently supports SAP ASE version 16.0 for use with SAP Business Suite products. All updates for SAP ASE server, or JDBC and ODBC drivers to be used with SAP Business Suite products are provided solely through the SAP Service Marketplace at: <https://support.sap.com/swdc>.

As for installations on-premises, do not download updates for the SAP ASE server, or for the JDBC and ODBC drivers directly from Sybase websites. For detailed information on patches which are supported for use with SAP Business Suite products on-premises and in Azure Virtual Machines see the following SAP Notes:

- [1590719](#)
- [1973241](#)

General information on running SAP Business Suite on SAP ASE can be found in the [SCN](#)

SAP ASE Configuration Guidelines for SAP related SAP ASE Installations in Azure VMs

Structure of the SAP ASE Deployment

In accordance with the general description, SAP ASE executables should be located or installed into the system drive of the VM's base VHD (drive c:). Typically, most of the SAP ASE system and tools databases are not really leveraged hard by SAP NetWeaver workload. Hence the system and tools databases (master, model, saptools, sybmgmtdb, sysystemdb) can remain on the C:\drive as well.

An exception could be the temporary database containing all work tables and temporary tables created by SAP ASE, which in case of some SAP ERP and all BW workloads might require either higher data volume or I/O operations volume which can't fit into the original VM's base VHD (drive c:).

Depending on the version of SAPInst/SWPM used to install the system, the database might contain:

- A single SAP ASE tempdb which is created when installing SAP ASE
- An SAP ASE tempdb created by installing SAP ASE and an additional saptempdb created by the SAP installation routine
- An SAP ASE tempdb created by installing SAP ASE and an additional tempdb that has been created manually (e.g. following SAP Note [1752266](#)) to meet ERP/BW specific tempdb requirements

In case of specific ERP or all BW workloads it makes sense, in regard to performance, to keep the tempdb devices of the additionally created tempdb (by SWPM or manually) on a drive other than C:. If no additional tempdb exists it is recommended to create one (SAP Note [1752266](#)).

For such systems the following steps should be performed for the additionally created tempdb:

- Move the first tempdb device to the first device of the SAP database
- Add tempdb devices to each of the VHDs containing a devices of the SAP database

This configuration enables tempdb to either consume more space than the system drive is able to provide. As a

reference one can check the tempdb device sizes on existing systems which run on-premises. Or such a configuration would enable IOPS numbers against tempdb which cannot be provided with the system drive. Again systems that are running on-premises can be used to monitor I/O workload against tempdb.

Never put any SAP ASE devices onto the D:\ drive of the VM. This also applies to the tempdb, even if the objects kept in the tempdb are only temporary.

Impact of Database Compression

In configurations where I/O bandwidth can become a limiting factor, every measure which reduces IOPS might help to stretch the workload one can run in an IaaS scenario like Azure. Therefore, it is strongly recommended to make sure that SAP ASE compression is used before uploading an existing SAP database to Azure.

The recommendation to perform compression before uploading to Azure if it is not already implemented is given out of several reasons:

- The amount of data to be uploaded to Azure is lower
- The duration of the compression execution is shorter assuming that one can use stronger hardware with more CPUs or higher I/O bandwidth or less I/O latency on-premises
- Smaller database sizes might lead to less costs for disk allocation

Data- and LOB-Compression work in a VM hosted in Azure Virtual Machines as it does on-premises. For more details on how to check if compression is already in use in an existing SAP ASE database please check SAP Note [1750510](#).

Using DBACockpit to monitor Database Instances

For SAP systems which are using SAP ASE as database platform, the DBACockpit is accessible as embedded browser windows in transaction DBACockpit or as Webdynpro. However the full functionality for monitoring and administering the database is available in the Webdynpro implementation of the DBACockpit only.

As with on-premises systems several steps are required to enable all SAP NetWeaver functionality used by the Webdynpro implementation of the DBACockpit. Please follow SAP Note [1245200](#) to enable the usage of webdynpros and generate the required ones. When following the instructions in the above notes you will also configure the Internet Communication Manager (icm) along with the ports to be used for http and https connections. The default setting for http looks like this:

```
icm/server_port_0 = PROT=HTTP,PORT=8000,PROCTIMEOUT=600,TIMEOUT=600  
icm/server_port_1 = PROT=HTTPS,PORT=443$$,PROCTIMEOUT=600,TIMEOUT=600
```

and the links generated in transaction DBACockpit will look similar to this:

```
https://<fullyqualifiedhostname>:44300/sap/bc/webdynpro/sap/dba_cockpit  
http://<fullyqualifiedhostname>:8000/sap/bc/webdynpro/sap/dba_cockpit
```

Depending on if and how the Azure Virtual Machine hosting the SAP system is connected via site-to-site, multi-site or ExpressRoute (Cross-Premises deployment) you need to make sure that ICM is using a fully qualified hostname that can be resolved on the machine where you are trying to open the DBACockpit from. Please see SAP Note [773830](#) to understand how ICM determines the fully qualified host name depending on profile parameters and set parameter icm/host_name_full explicitly if required.

If you deployed the VM in a Cloud-Only scenario without cross-premises connectivity between on-premises and Azure, you need to define a public IP address and a domainlabel. The format of the public DNS name of the VM will then look like this :

```
<custom domainlabel>.<azure region>.cloudapp.azure.com
```

More details related to the DNS name can be found [here](#).

Setting the SAP profile parameter icm/host_name_full to the DNS name of the Azure VM the link might look similar to:

https://mydomainlabel.westeurope.cloudapp.net:44300/sap/bc/webdynpro/sap/dba_cockpit

http://mydomainlabel.westeurope.cloudapp.net:8000/sap/bc/webdynpro/sap/dba_cockpit

In this case you need to make sure to:

- Add Inbound rules to the Network Security Group in the Azure Portal for the TCP/IP ports used to communicate with ICM
- Add Inbound rules to the Windows Firewall configuration for the TCP/IP ports used to communicate with the ICM

For an automated imported of all corrections available, it is recommended to periodically apply the correction collection SAP Note applicable to your SAP version:

- [1558958](#)
- [1619967](#)
- [1882376](#)

Further information about DBA Cockpit for SAP ASE can be found in the following SAP Notes:

- [1605680](#)
- [1757924](#)
- [1757928](#)
- [1758182](#)
- [1758496](#)
- [1814258](#)
- [1922555](#)
- [1956005](#)

Backup/Recovery Considerations for SAP ASE

When deploying SAP ASE into Azure your backup methodology must be reviewed. Even if the system is not a productive system, the SAP database hosted by SAP ASE must be backed up periodically. Since Azure Storage keeps three images, a backup is now less important in respect to compensating a storage crash. The primary reason for maintaining a proper backup and restore plan is more that you can compensate for logical/manual errors by providing point in time recovery capabilities. So the goal is to either use backups to restore the database back to a certain point in time or to use the backups in Azure to seed another system by copying the existing database. For example, you could transfer from a 2-Tier SAP configuration to a 3-Tier system setup of the same system by restoring a backup.

Backing up and restoring a database in Azure works the same way as it does on-premises. Please see SAP Notes:

- [1588316](#)
- [1585981](#)

for details on creating dump configurations and scheduling backups. Depending on your strategy and needs you can configure database and log dumps to disk onto one of the existing VHDs or add an additional VHD for the backup. To reduce the danger of data loss in case of an error it is recommended to use a VHD where no database device is located.

Besides data- and LOB compression SAP ASE also offers backup compression. To occupy less space with the

database and log dumps it is recommended to use backup compression. Please see SAP Note [1588316](#) for more information. Compressing the backup is also crucial to reduce the amount of data to be transferred if you plan to download backups or VHDs containing backup dumps from the Azure Virtual Machine to on-premises.

Do not use drive D:\ as database or log dump destination.

Performance Considerations for Backups/Restores

As in bare-metal deployments, backup/restore performance is dependent on how many volumes can be read in parallel and what the throughput of those volumes might be. In addition, the CPU consumption used by backup compression may play a significant role on VMs with just up to 8 CPU threads. Therefore, one can assume:

- The fewer the number of VHDs used to store the database devices, the smaller the overall throughput in reading
- The smaller the number of CPU threads in the VM, the more severe the impact of backup compression
- The fewer targets (Stripe Directories, VHDs) to write the backup to, the lesser the throughput

To increase the number of targets to write to there are two options which can be used/combined depending on your needs:

- Striping the backup target volume over multiple mounted VHDs in order to improve the IOPS throughput on that striped volume
- Creating a dump configuration on SAP ASE level which uses more than one target directory to write the dump to

Striping a volume over multiple mounted VHDs has been discussed earlier in this guide. For more information on using multiple directories in the SAP ASE dump configuration please refer to the documentation on Stored Procedure `sp_config_dump` which is used to create the dump configuration on the [Sybase Infocenter](#).

Disaster Recovery with Azure VMs

Data Replication with SAP Sybase Replication Server

With the SAP Sybase Replication Server (SRS) SAP ASE provides a warm standby solution to transfer database transactions to a distant location asynchronously.

The installation and operation of SRS works as well functionally in a VM hosted in Azure Virtual Machine Services as it does on-premises.

ASE HADR via SAP Replication Server is planned with a future release. It will be tested with and released for Microsoft Azure platforms as soon as it is available.

Specifics to SAP ASE on Linux

Starting with Microsoft Azure, you can easily migrate your existing SAP ASE applications to Azure Virtual Machines. SAP ASE in a Virtual Machine enables you to reduce the total cost of ownership of deployment, management and maintenance of enterprise breadth applications by easily migrating these applications to Microsoft Azure. With SAP ASE in an Azure Virtual Machine, administrators and developers can still use the same development and administration tools that are available on-premises.

For deploying Azure VMs it's important to know the official SLAs which can be found here :

<https://azure.microsoft.com/support/legal/sla>

SAP sizing information and a list of SAP certified VM SKUs will be provided in SAP Note [1928533](#). Additional SAP sizing documents for Azure Virtual machines can be found here <http://blogs.msdn.com/b/saponsqlserver/archive/2015/06/19/how-to-size-sap-systems-running-on-azure-vms.aspx> and here <http://blogs.msdn.com/b/saponsqlserver/archive/2015/12/01/new-white-paper-on-sizing-sap-solutions-on-azure-public-cloud.aspx>

Statements and recommendations in regard to the usage of Azure Storage, Deployment of SAP VMs or SAP Monitoring apply to deployments of SAP ASE in conjunction with SAP applications as stated throughout the first four chapters of this document.

The following two SAP Notes include general information about ASE on Linux and ASE in the Cloud :

- [2134316](#)
- [1941500](#)

SAP ASE Version Support

SAP currently supports SAP ASE version 16.0 for use with SAP Business Suite products. All updates for SAP ASE server, or JDBC and ODBC drivers to be used with SAP Business Suite products are provided solely through the SAP Service Marketplace at: <https://support.sap.com/swdc>.

As for installations on-premises, do not download updates for the SAP ASE server, or for the JDBC and ODBC drivers directly from Sybase websites. For detailed information on patches which are supported for use with SAP Business Suite products on-premises and in Azure Virtual Machines see the following SAP Notes:

- [1590719](#)
- [1973241](#)

General information on running SAP Business Suite on SAP ASE can be found in the [SCN](#)

SAP ASE Configuration Guidelines for SAP related SAP ASE Installations in Azure VMs

Structure of the SAP ASE Deployment

In accordance with the general description, SAP ASE executables should be located or installed into the root file system of the VM (/sybase). Typically, most of the SAP ASE system and tools databases are not really leveraged hard by SAP NetWeaver workload. Hence the system and tools databases (master, model, saptools, sybmgmtdb, sysystemdb) can remain on the root file system as well.

An exception could be the temporary database containing all work tables and temporary tables created by SAP ASE, which in case of some SAP ERP and all BW workloads might require either higher data volume or I/O operations volume which can't fit into the original VM's OS disk.

Depending on the version of SAPInst/SWPM used to install the system, the database might contain:

- A single SAP ASE tempdb which is created when installing SAP ASE
- An SAP ASE tempdb created by installing SAP ASE and an additional saptempdb created by the SAP installation routine
- An SAP ASE tempdb created by installing SAP ASE and an additional tempdb that has been created manually (e.g. following SAP Note [1752266](#)) to meet ERP/BW specific tempdb requirements

In case of specific ERP or all BW workloads it makes sense, in regard to performance, to keep the tempdb devices of the additionally created tempdb (by SWPM or manually) on a separate file system which can be represented by a single Azure data disk or a Linux RAID spanning multiple Azure data disks. If no additional tempdb exists it is recommended to create one (SAP Note [1752266](#)).

For such systems the following steps should be performed for the additionally created tempdb:

- Move the first tempdb directory to the first file system of the SAP database
- Add tempdb directories to each of the VHDs containing a filesystem of the SAP database

This configuration enables tempdb to either consume more space than the system drive is able to provide. As a reference one can check the tempdb directory sizes on existing systems which run on-premises. Or such a configuration would enable IOPS numbers against tempdb which cannot be provided with the system drive. Again systems that are running on-premises can be used to monitor I/O workload against tempdb.

Never put any SAP ASE directories onto /mnt or /mnt/resource of the VM. This also applies to the tempdb, even if the objects kept in the tempdb are only temporary because /mnt or /mnt/resource is a default Azure VM temp space which is not persistent. More details about the Azure VM temp space can be found in [this article](#)

Impact of Database Compression

In configurations where I/O bandwidth can become a limiting factor, every measure which reduces IOPS might help to stretch the workload one can run in an IaaS scenario like Azure. Therefore, it is strongly recommended to make sure that SAP ASE compression is used before uploading an existing SAP database to Azure.

The recommendation to perform compression before uploading to Azure if it is not already implemented is given out of several reasons:

- The amount of data to be uploaded to Azure is lower
- The duration of the compression execution is shorter assuming that one can use stronger hardware with more CPUs or higher I/O bandwidth or less I/O latency on-premises
- Smaller database sizes might lead to less costs for disk allocation

Data- and LOB-Compression work in a VM hosted in Azure Virtual Machines as it does on-premises. For more details on how to check if compression is already in use in an existing SAP ASE database please check SAP Note [1750510](#). Also see SAP Note [2121797](#) for additional information regarding database compression.

Using DBACockpit to monitor Database Instances

For SAP systems which are using SAP ASE as database platform, the DBACockpit is accessible as embedded browser windows in transaction DBACockpit or as Webdynpro. However the full functionality for monitoring and administering the database is available in the Webdynpro implementation of the DBACockpit only.

As with on-premises systems several steps are required to enable all SAP NetWeaver functionality used by the Webdynpro implementation of the DBACockpit. Please follow SAP Note [1245200](#) to enable the usage of webdynpros and generate the required ones. When following the instructions in the above notes you will also configure the Internet Communication Manager (icm) along with the ports to be used for http and https connections. The default setting for http looks like this:

```
icm/server_port_0 = PROT=HTTP,PORT=8000,PROCTIMEOUT=600,TIMEOUT=600  
icm/server_port_1 = PROT=HTTPS,PORT=443$$,PROCTIMEOUT=600,TIMEOUT=600
```

and the links generated in transaction DBACockpit will look similar to this:

```
https://<fullyqualifiedhostname>:44300/sap/bc/webdynpro/sap/dba_cockpit  
http://<fullyqualifiedhostname>:8000/sap/bc/webdynpro/sap/dba_cockpit
```

Depending on if and how the Azure Virtual Machine hosting the SAP system is connected via site-to-site, multi-site or ExpressRoute (Cross-Premises deployment) you need to make sure that ICM is using a fully qualified hostname that can be resolved on the machine where you are trying to open the DBACockpit from. Please see SAP Note [773830](#) to understand how ICM determines the fully qualified host name depending on profile parameters and set parameter icm/host_name_full explicitly if required.

If you deployed the VM in a Cloud-Only scenario without cross-premises connectivity between on-premises and Azure, you need to define a public IP address and a domainlabel. The format of the public DNS name of the VM will then look like this :

```
<custom domainlabel>.<azure region>.cloudapp.azure.com
```

More details related to the DNS name can be found [here](#).

Setting the SAP profile parameter `icm/host_name_full` to the DNS name of the Azure VM the link might look similar to:

https://mydomainlabel.westeurope.cloudapp.net:44300/sap/bc/webdynpro/sap/dba_cockpit

http://mydomainlabel.westeurope.cloudapp.net:8000/sap/bc/webdynpro/sap/dba_cockpit

In this case you need to make sure to:

- Add Inbound rules to the Network Security Group in the Azure Portal for the TCP/IP ports used to communicate with ICM
- Add Inbound rules to the Windows Firewall configuration for the TCP/IP ports used to communicate with the ICM

For an automated imported of all corrections available, it is recommended to periodically apply the correction collection SAP Note applicable to your SAP version:

- [1558958](#)
- [1619967](#)
- [1882376](#)

Further information about DBA Cockpit for SAP ASE can be found in the following SAP Notes:

- [1605680](#)
- [1757924](#)
- [1757928](#)
- [1758182](#)
- [1758496](#)
- [1814258](#)
- [1922555](#)
- [1956005](#)

Backup/Recovery Considerations for SAP ASE

When deploying SAP ASE into Azure your backup methodology must be reviewed. Even if the system is not a productive system, the SAP database hosted by SAP ASE must be backed up periodically. Since Azure Storage keeps three images, a backup is now less important in respect to compensating a storage crash. The primary reason for maintaining a proper backup and restore plan is more that you can compensate for logical/manual errors by providing point in time recovery capabilities. So the goal is to either use backups to restore the database back to a certain point in time or to use the backups in Azure to seed another system by copying the existing database. For example, you could transfer from a 2-Tier SAP configuration to a 3-Tier system setup of the same system by restoring a backup.

Backing up and restoring a database in Azure works the same way as it does on-premises. Please see SAP Notes:

- [1588316](#)
- [1585981](#)

for details on creating dump configurations and scheduling backups. Depending on your strategy and needs you can configure database and log dumps to disk onto one of the existing VHDs or add an additional VHD for the backup. To reduce the danger of data loss in case of an error it is recommended to use a VHD where no database directory/file is located.

Besides data- and LOB compression SAP ASE also offers backup compression. To occupy less space with the database and log dumps it is recommended to use backup compression. Please see SAP Note [1588316](#) for

more information. Compressing the backup is also crucial to reduce the amount of data to be transferred if you plan to download backups or VHDs containing backup dumps from the Azure Virtual Machine to on-premises.

Do not use the Azure VM temp space /mnt or /mnt/resource as database or log dump destination.

Performance Considerations for Backups/Restores

As in bare-metal deployments, backup/restore performance is dependent on how many volumes can be read in parallel and what the throughput of those volumes might be. In addition, the CPU consumption used by backup compression may play a significant role on VMs with just up to 8 CPU threads. Therefore, one can assume:

- The fewer the number of VHDs used to store the database devices, the smaller the overall throughput in reading
- The smaller the number of CPU threads in the VM, the more severe the impact of backup compression
- The fewer targets (Linux software RAID, VHDs) to write the backup to, the lesser the throughput

To increase the number of targets to write to there are two options which can be used/combined depending on your needs:

- Striping the backup target volume over multiple mounted VHDs in order to improve the IOPS throughput on that striped volume
- Creating a dump configuration on SAP ASE level which uses more than one target directory to write the dump to

Striping a volume over multiple mounted VHDs has been discussed earlier in this guide. For more information on using multiple directories in the SAP ASE dump configuration please refer to the documentation on Stored Procedure `sp_config_dump` which is used to create the dump configuration on the [Sybase Infocenter](#).

Disaster Recovery with Azure VMs

Data Replication with SAP Sybase Replication Server

With the SAP Sybase Replication Server (SRS) SAP ASE provides a warm standby solution to transfer database transactions to a distant location asynchronously.

The installation and operation of SRS works as well functionally in a VM hosted in Azure Virtual Machine Services as it does on-premises.

ASE HADR via SAP Replication Server is NOT supported at this point in time. It might be tested with and released for Microsoft Azure platforms in the future.

Specifics to Oracle Database on Windows

Since midyear 2013, Oracle software is supported by Oracle to run on Microsoft Windows Hyper-V and Azure. Please read this article to get more details on the general support of Windows Hyper-V and Azure by Oracle:
https://blogs.oracle.com/cloud/entry/oracle_and_microsoft_join_forces

Following the general support, the specific scenario of SAP applications leveraging Oracle Databases is supported as well. Details are named in this part of the document.

Oracle Version Support

All details about Oracle versions and corresponding OS versions which are supported for running SAP on Oracle on Azure Virtual Machines can be found in the following SAP Note [2039619](https://service.sap.com/note/2039619)

General information about running SAP Business Suite on Oracle can be found on SCN:

<https://scn.sap.com/community/oracle>

Oracle Configuration Guidelines for SAP Installations in Azure VMs

Storage configuration

Only single instance Oracle using NTFS formatted disks is supported. All database files must be stored on the NTFS file system based on VHD disks. These VHDs are mounted to the Azure VM and are based on Azure Page BLOB Storage (<https://msdn.microsoft.com/library/azure/ee691964.aspx>). Any kind of network drives or remote shares like Azure file services:

- <https://blogs.msdn.com/b/windowsazurerestorage/archive/2014/05/12/introducing-microsoft-azure-file-service.aspx>
- <https://blogs.msdn.com/b/windowsazurerestorage/archive/2014/05/27/persisting-connections-to-microsoft-azure-files.aspx>

are **NOT** supported for Oracle database files!

Using Azure VHDs based on Azure Page BLOB Storage, the statements made in this document in chapter [Caching for VMs and VHDs](#) and [Microsoft Azure Storage](#) apply to deployments with the Oracle Database as well.

As explained earlier in the general part of the document, quotas on IOPS throughput for Azure VHDs exist. The exact quotas are depending on the VM type used. A list of VM types with their quotas can be found [here](#)

To identify the supported Azure VM types, please refer to SAP note [1928533](#)

As long as the current IOPS quota per disk satisfies the requirements, it is possible to store all the DB files on one single mounted Azure VHD.

If more IOPS are required, it is strongly recommended to use Window Storage Pools (only available in Windows Server 2012 and higher) or Windows striping for Windows 2008 R2 to create one big logical device over multiple mounted VHD disks. See also chapter [Software RAID](#) of this document. This approach simplifies the administration overhead to manage the disk space and avoids the effort to manually distribute files across multiple mounted VHDs.

Backup / Restore

For backup / restore functionality, the SAP BR*Tools for Oracle are supported in the same way as on standard Windows Server Operating Systems and Hyper-V. Oracle Recovery Manager (RMAN) is also supported for backups to disk and restore from disk.

High Availability

Oracle Data Guard is supported for high availability and disaster recovery purposes. Details can be found in [this documentation](#).

Other

All other general topics like Azure Availability Sets or SAP monitoring apply as described in the first three chapters of this document for deployments of VMs with the Oracle Database as well.

Specifics for the SAP MaxDB Database on Windows

SAP MaxDB Version Support

SAP currently supports SAP MaxDB version 7.9 for use with SAP NetWeaver-based products in Azure. All updates for SAP MaxDB server, or JDBC and ODBC drivers to be used with SAP NetWeaver-based products are provided solely through the SAP Service Marketplace at <https://support.sap.com/swdc>. General information on running SAP NetWeaver on SAP MaxDB can be found at <https://scn.sap.com/community/maxdb>.

Supported Microsoft Windows Versions and Azure VM types for SAP MaxDB DBMS

To find the supported Microsoft Windows version for SAP MaxDB DBMS on Azure, see:

- [SAP Product Availability Matrix \(PAM\)](#)
- SAP Note 1928533

It is highly recommended to use the newest version of the operating system Microsoft Windows, which is Microsoft Windows 2012 R2.

Available SAP MaxDB Documentation

You can find the updated list of SAP MaxDB documentation in the following SAP Note [767598](#)

SAP MaxDB Configuration Guidelines for SAP Installations in Azure VMs

Storage configuration

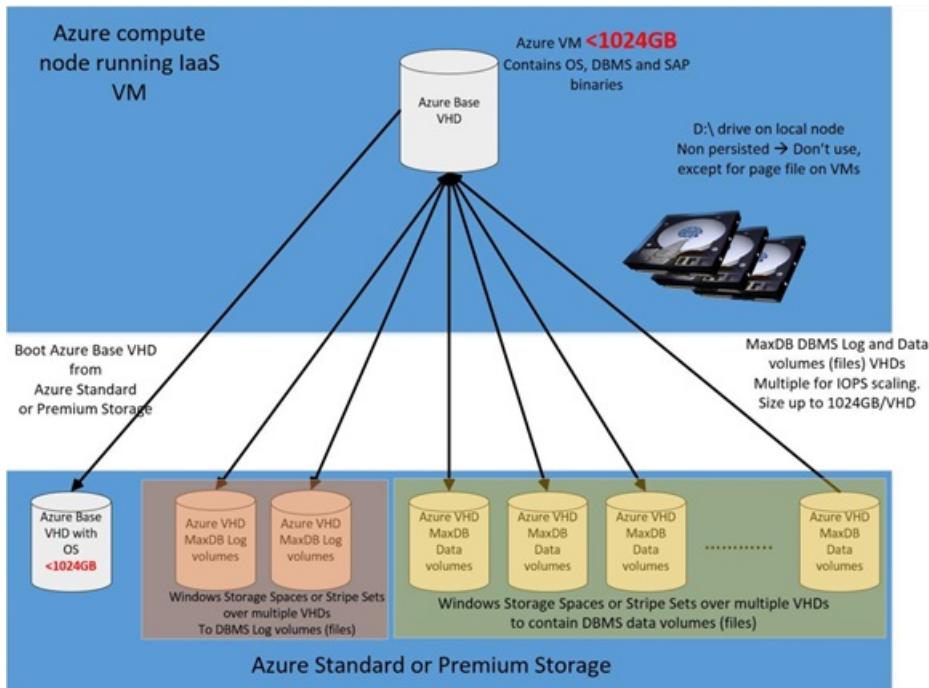
Azure storage best practices for SAP MaxDB follow the general recommendations mentioned in chapter [Structure of a RDBMS Deployment](#).

IMPORTANT

Like other databases, SAP MaxDB also has data and log files. However, in SAP MaxDB terminology the correct term is "volume" (not "file"). For example, there are SAP MaxDB data volumes and log volumes. Do not confuse these with OS disk volumes.

In short you have to:

- Set the Azure storage account that holds the SAP MaxDB data and log volumes (i.e. files) to **Local Redundant Storage (LRS)** as specified in chapter [Microsoft Azure Storage](#).
- Separate the IO path for SAP MaxDB data volumes (i.e. files) from the IO path for log volumes (i.e. files). This means that SAP MaxDB data volumes (i.e. files) have to be installed on one logical drive and SAP MaxDB log volumes (i.e. files) have to be installed on another logical drive.
- Set the proper file caching for each Azure blob, depending on whether you use it for SAP MaxDB data or log volumes (i.e. files), and whether you use Azure Standard or Azure Premium Storage, as described in chapter [Caching for VMs](#).
- As long as the current IOPS quota per disk satisfies the requirements, it is possible to store all the data volumes on a single mounted Azure VHD, and also store all database log volumes on another single mounted Azure VHD.
- If more IOPS and/or space are required, it is strongly recommended to use Microsoft Window Storage Pools (only available in Microsoft Windows Server 2012 and higher) or Microsoft Windows striping for Microsoft Windows 2008 R2 to create one large logical device over multiple mounted VHD disks. See also chapter [Software RAID](#) of this document. This approach simplifies the administration overhead to manage the disk space and avoids the effort of manually distributing files across multiple mounted VHDs.
- For the highest IOPS requirements, you can use Azure Premium Storage, which is available on DS-series and GS-series VMs.



Backup and Restore

When deploying SAP MaxDB into Azure, you must review your backup methodology. Even if the system is not a productive system, the SAP database hosted by SAP MaxDB must be backed up periodically. Since Azure Storage keeps three images, a backup is now less important in terms of protecting your system against storage failure and more important operational or administrative failures. The primary reason for maintaining a proper backup and restore plan is so that you can compensate for logical or manual errors by providing point-in-time recovery capabilities. So the goal is to either use backups to restore the database to a certain point in time or to use the backups in Azure to seed another system by copying the existing database. For example, you could transfer from a 2-tier SAP configuration to a 3-tier system setup of the same system by restoring a backup.

Backing up and restoring a database in Azure works the same way as it does for on-premises systems, so you can use standard SAP MaxDB backup/restore tools, which are described in one of the SAP MaxDB documentation documents listed in SAP Note [767598](#).

Performance Considerations for Backup and Restore

As in bare-metal deployments, backup and restore performance is dependent on how many volumes can be read in parallel and the throughput of those volumes. In addition, the CPU consumption used by backup compression can play a significant role on VMs with up to 8 CPU threads. Therefore, one can assume:

- The fewer the number of VHDs used to store the database devices, the lower the overall read throughput
- The smaller the number of CPU threads in the VM, the more severe the impact of backup compression
- The fewer targets (Stripe Directories, VHDs) to write the backup to, the lower the throughput

To increase the number of targets to write to, there are two options that you can use, possibly in combination, depending on your needs:

- Dedicating separate volumes for backup
- Striping the backup target volume over multiple mounted VHDs in order to improve the IOPS throughput on that striped disk volume
- Having separate dedicated logical disk devices for:
 - SAP MaxDB backup volumes (i.e. files)
 - SAP MaxDB data volumes (i.e. files)
 - SAP MaxDB log volumes (i.e. files)

Striping a volume over multiple mounted VHDs has been discussed earlier in chapter [Software RAID](#) of this document.

Other

All other general topics such as Azure Availability Sets or SAP monitoring also apply as described in the first three chapters of this document for deployments of VMs with the SAP MaxDB database. Other SAP MaxDB-specific settings are transparent to Azure VMs and are described in different documents listed in SAP Note [767598](#) and in these SAP notes:

- [826037](#)
- [1139904](#)
- [1173395](#)

Specifics for SAP liveCache on Windows

SAP liveCache Version Support

Minimal version of SAP liveCache supported in Azure Virtual Machines is **SAP LC/LCAPP 10.0 SP 25** including **liveCache 7.9.08.31** and **LCA-Build 25**, released for **EhP 2 for SAP SCM 7.0** and higher.

Supported Microsoft Windows Versions and Azure VM types for SAP liveCache DBMS

To find the supported Microsoft Windows version for SAP liveCache on Azure, see:

- [SAP Product Availability Matrix \(PAM\)](#)
- SAP Note [1928533](#)

It is highly recommended to use the newest version of the operating system Microsoft Windows, which is Microsoft Windows 2012 R2.

SAP liveCache Configuration Guidelines for SAP Installations in Azure VMs

Recommended Azure VM Types

As SAP liveCache is an application that performs huge calculations, the amount and speed of RAM and CPU has a major influence on SAP liveCache performance.

For the Azure VM types supported by SAP (SAP Note [1928533](#)), all virtual CPU resources allocated to the VM are backed by dedicated physical CPU resources of the hypervisor. No overprovisioning (and therefore no competition for CPU resources) takes place.

Similarly, for all Azure VM instance types supported by SAP, the VM memory is 100 % mapped to the physical memory – overprovisioning (over-commitment), for example, is not used.

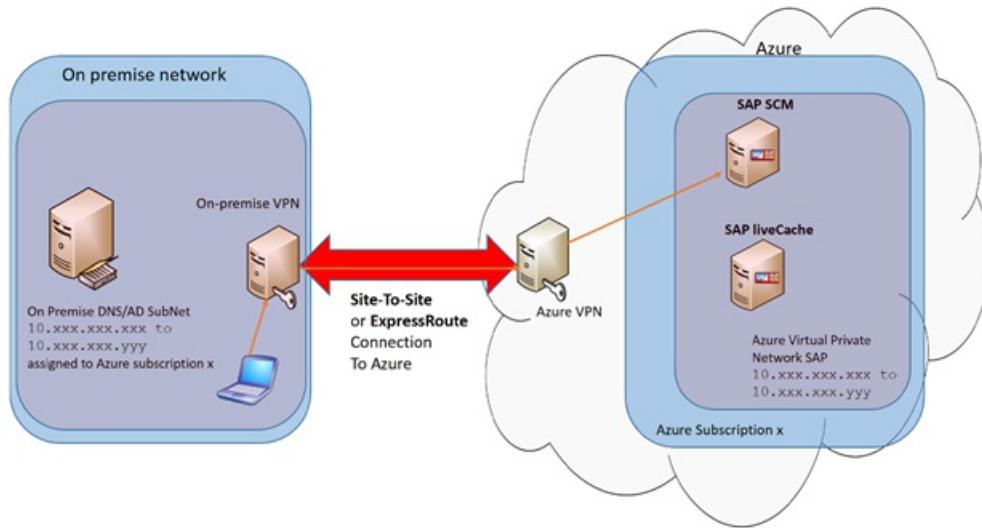
From this perspective it is highly recommended to use the new D- series or DS-series (in combination with Azure Premium Storage) Azure VM type, as they have 60 % faster processors than the A-series. For the highest RAM and CPU load, you can use G-series and GS-series (in combination with Azure Premium Storage) VMs with the latest Intel® Xeon® processor E5 v3 family, which have twice the memory and four times the solid state drive storage (SSDs) of the D/DS-series.

Storage Configuration

As SAP liveCache is based on SAP MaxDB technology, all the Azure storage best practice recommendations mentioned for SAP MaxDB in chapter [Storage configuration](#) are also valid for SAP liveCache.

Dedicated Azure VM for liveCache

As SAP liveCache intensively uses computational power, for productive usage it is highly recommended to deploy on a dedicated Azure Virtual Machine.



Backup and Restore

Backup and restore, including performance considerations, are already described in the relevant SAP MaxDB chapters [Backup and Restore](#) and [Performance Considerations for Backup and Restore](#).

Other

All other general topics are already described in the relevant SAP MaxDB [this](#) chapter.

Specifics for the SAP Content Server on Windows

The SAP Content Server is a separate, server-based component to store content such as electronic documents in different formats. The SAP Content Server is provided by development of technology and is to be used cross-application for any SAP applications. It is installed on a separate system. Typical content is training material and documentation from Knowledge Warehouse or technical drawings originating from the mySAP PLM Document Management System.

SAP Content Server Version Support

SAP currently supports:

- **SAP Content Server** with version **6.50 (and higher)**
- **SAP MaxDB version 7.9**
- **Microsoft IIS (Internet Information Server) version 8.0 (and higher)**

It is highly recommended to use the newest version of SAP Content Server, which at the time of writing this document is **6.50 SP4**, and the newest version of **Microsoft IIS 8.5**.

Check the latest supported versions of SAP Content Server and Microsoft IIS in the [SAP Product Availability Matrix \(PAM\)](#).

Supported Microsoft Windows and Azure VM types for SAP Content Server

To find out supported Windows version for SAP Content Server on Azure, see:

- [SAP Product Availability Matrix \(PAM\)](#)
- [SAP Note 1928533](#)

It is highly recommended to use the newest version of Microsoft Windows, which at the time of writing this document is **Windows Server 2012 R2**.

SAP Content Server Configuration Guidelines for SAP Installations in Azure VMs

Storage Configuration

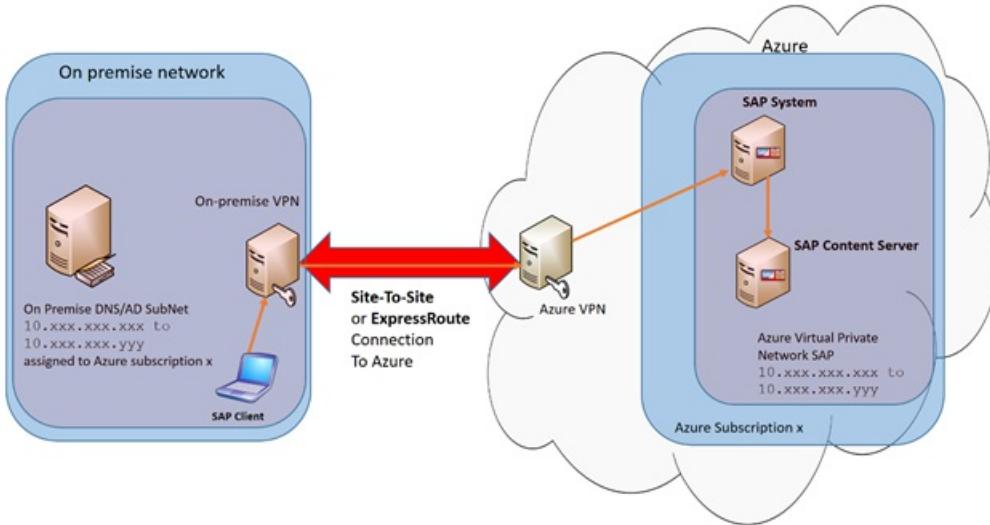
If you configure SAP Content Server to store files in the SAP MaxDB database, all Azure storage best practices recommendation mentioned for SAP MaxDB in chapter [Storage Configuration](#) are also valid for the SAP

Content Server scenario.

If you configure SAP Content Server to store files in the file system, it is recommended to use a dedicated logical drive. Using storage spaces enables you to also increase logical disk size and IOPS throughput, as described in chapter [Software RAID](#).

SAP Content Server Location

SAP Content Server has to be deployed in the same Azure region and Azure VNET where the SAP system is deployed. You are free to decide whether you want to deploy SAP Content Server components on a dedicated Azure VM or on the same VM where the SAP system is running.

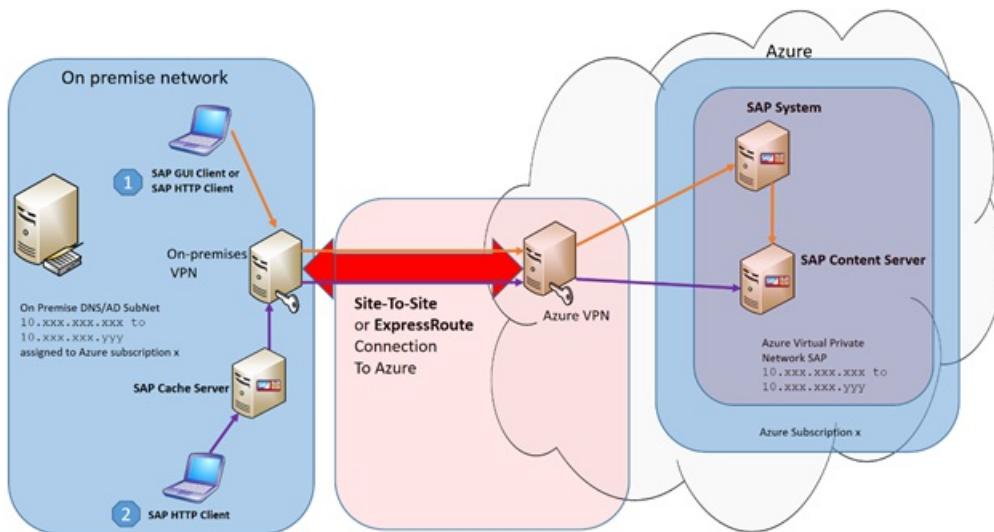


SAP Cache Server Location

The SAP Cache Server is an additional server-based component to provide access to (cached) documents locally. The SAP Cache Server caches the documents of a SAP Content Server. This is to optimize network traffic if documents have to be retrieved more than once from different locations. The general rule is that the SAP Cache Server has to be physically close to the client that accesses the SAP Cache Server.

Here you have two options:

1. **Client is a backend SAP system** If a backend SAP system is configured to access SAP Content Server, that SAP system is a client. As both SAP system and SAP Content Server are deployed in the same Azure region – in the same Azure datacenter – they are physically close to each other. Therefore, there is no need to have a dedicated SAP Cache Server. SAP UI clients (SAP GUI or web browser) access the SAP system directly, and the SAP system retrieves documents from the SAP Content Server.
2. **Client is an on-premises web browser** The SAP Content Server can be configured to be accessed directly by the web browser. In this case, a web browser running on-premises is a client of the SAP Content Server. On-premises datacenter and Azure datacenter are placed in different physical locations (ideally close to each other). Your on-premises datacenter is connected to Azure via Azure Site-to-Site VPN or ExpressRoute. Although both options offer secure VPN network connection to Azure, site-to-site network connection does not offer a network bandwidth and latency SLA between the on-premises datacenter and the Azure datacenter. To speed up access to documents, you can do one of the following:
 - a. Install SAP Cache Server on-premises, close to the on-premises web browser (option on [this Figure](#))
 - b. Configure Azure ExpressRoute, which offers a high-speed and low-latency dedicated network connection between on-premises datacenter and Azure datacenter.



Backup / Restore

If you configure the SAP Content Server to store files in the SAP MaxDB database, the backup/restore procedure and performance considerations are already described in SAP MaxDB chapter [Backup and Restore](#) and chapter [Performance Considerations for Backup and Restore](#).

If you configure the SAP Content Server to store files in the file system, one option is to execute manual backup/restore of the whole file structure where the documents are located. Similar to SAP MaxDB backup/restore, it is recommended to have a dedicated disk volume for backup purpose.

Other

Other SAP Content Server specific settings are transparent to Azure VMs and are described in various documents and SAP Notes:

- <https://service.sap.com/contentserver>
- SAP Note 1619726

Specifics to IBM DB2 for LUW on Windows

With Microsoft Azure, you can easily migrate your existing SAP application running on IBM DB2 for Linux, UNIX, and Windows (LUW) to Azure virtual machines. With SAP on IBM DB2 for LUW, administrators and developers can still use the same development and administration tools which are available on-premises. General information about running SAP Business Suite on IBM DB2 for LUW can be found in the SAP Community Network (SCN) at <https://scn.sap.com/community/db2-for-linux-unix-windows>.

For additional information and updates about SAP on DB2 for LUW on Azure, see SAP Note [2233094](#).

IBM DB2 for Linux, UNIX, and Windows Version Support

SAP on IBM DB2 for LUW on Microsoft Azure Virtual Machine Services is supported as of DB2 version 10.5.

For information about supported SAP products and Azure VM types, please refer to SAP Note [1928533](#).

IBM DB2 for Linux, UNIX, and Windows Configuration Guidelines for SAP Installations in Azure VMs

Storage Configuration

All database files must be stored on the NTFS file system based on VHD disks. These VHDs are mounted to the Azure VM and are based in Azure Page BLOB Storage

(<https://msdn.microsoft.com/library/azure/ee691964.aspx>). Any kind of network drives or remote shares like the following Azure file services are **NOT** supported for database files:

- <https://blogs.msdn.com/b/windowsazurestorage/archive/2014/05/12/introducing-microsoft-azure-file-service.aspx>
- <https://blogs.msdn.com/b/windowsazurestorage/archive/2014/05/27/persisting-connections-to-microsoft->

[azure-files.aspx](#)

If you are using Azure VHDs based on Azure Page BLOB Storage, the statements made in this document in chapter [Structure of a RDBMS Deployment](#) also apply to deployments with the IBM DB2 for LUW Database.

As explained earlier in the general part of the document, quotas on IOPS throughput for Azure VHDs exist. The exact quotas depend on the VM type used. A list of VM types with their quotas can be found [here](#)

As long as the current IOPS quota per disk is sufficient, it is possible to store all the database files on one single mounted Azure VHD.

For performance considerations also refer to chapter "Data Safety and Performance Considerations for Database Directories" in SAP installation guides.

Alternatively, you can use Windows Storage Pools (only available in Windows Server 2012 and higher) or Windows striping for Windows 2008 R2 as described in chapter [Software RAID](#) of this document to create one big logical device over multiple mounted VHD disks. For the disks containing the DB2 storage paths for your sapdata and saptmp directories, you must specify a physical disk sector size of 512 KB. When using Windows Storage Pools, you must create the storage pools manually via command line interface using the parameter „-LogicalSectorSizeDefault". For details see <https://technet.microsoft.com/library/hh848689.aspx>.

Backup/Restore

The backup/restore functionality for IBM DB2 for LUW is supported in the same way as on standard Windows Server Operating Systems and Hyper-V.

You must make sure that you have a valid database backup strategy in place.

As in bare-metal deployments, backup/restore performance depends on how many volumes can be read in parallel and what the throughput of those volumes might be. In addition, the CPU consumption used by backup compression may play a significant role on VMs with just up to 8 CPU threads. Therefore, one can assume:

- The fewer the number of VHDs used to store the database devices, the smaller the overall throughput in reading
- The smaller the number of CPU threads in the VM, the more severe the impact of backup compression
- The fewer targets (Stripe Directories, VHDs) to write the backup to, the lower the throughput

To increase the number of targets to write to, two options can be used/combined depending on your needs:

- Striping the backup target volume over multiple mounted VHDs in order to improve the IOPS throughput on that striped volume
- Using more than one target directory to write the backup to

High Availability and Disaster Recovery

Microsoft Cluster Server (MSCS) is not supported.

DB2 high availability disaster recovery (HADR) is supported. If the virtual machines of the HA configuration have working name resolution, the setup in Azure will not differ from any setup that is done on-premises. It is not recommended to rely on IP resolution only.

Do not use Azure Store Geo-Replication. For further information, refer to chapter [Microsoft Azure Storage](#) and chapter [High Availability and Disaster Recovery with Azure VMs](#).

Other

All other general topics like Azure Availability Sets or SAP monitoring apply as described in the first three chapters of this document for deployments of VMs with IBM DB2 for LUW as well.

Also refer to chapter [General SQL Server for SAP on Azure Summary](#).

SAP NetWeaver on Azure Virtual Machines (VMs) – Deployment Guide

1/17/2017 • 41 min to read • [Edit on GitHub](#)

Microsoft Azure enables companies to acquire compute and storage resources in minimal time without lengthy procurement cycles. Azure Virtual Machines allows companies to deploy classical applications, like SAP NetWeaver based applications into Azure and extend their reliability and availability without having further resources available on-premises. Microsoft Azure also supports cross-premises connectivity, which enables companies to actively integrate Azure Virtual Machines into their on-premises domains, their Private Clouds and their SAP System Landscape.

This white paper describes step by step how an Azure Virtual Machine is prepared for the deployment of SAP NetWeaver based applications. It assumes that the information contained in the [Planning and Implementation Guide](#) is known. If not, the respective document should be read first.

The paper complements the SAP Installation Documentation and SAP Notes which represent the primary resources for installations and deployments of SAP software on given platforms.

Introduction

A large number of companies worldwide use SAP NetWeaver based applications – most prominently the SAP Business Suite – to run their mission critical business processes. System health is therefore a crucial asset, and the ability to provide enterprise support in case of a malfunction, including performance incidents, becomes a vital requirement. Microsoft Azure provides superior platform instrumentation to accommodate the supportability requirements of all business critical applications. This guide makes sure that a Microsoft Azure Virtual Machine targeted for deployment of SAP Software is configured such that enterprise support can be offered, regardless which way the virtual machine gets created, be it taken out of the Azure Marketplace or using a customer specific image. In the following, all necessary setup steps are described in detail.

Prerequisites and Resources

Prerequisites

Before you start, please make sure that the prerequisites that are described in the following chapters are met.

Local Personal Computer

The setup of an Azure Virtual Machine for SAP Software deployment comprises of several steps. To manage Windows VMs or Linux VMs, you need to use a PowerShell script and the Microsoft Azure Portal. For that, a local Personal Computer running Windows 7 or higher is necessary. If you only want to manage Linux VMs and want to use a Linux machine for this task, you can also use the Azure Command Line Interface (Azure CLI).

Internet Connection

To download and execute the required tools and scripts, an Internet connection is required. Furthermore, the Microsoft Azure Virtual Machine running the Azure Enhanced Monitoring Extension for SAP needs access to the Internet. In case this Azure VM is part of an Azure Virtual Network or on-premises domain, make sure that the relevant proxy settings are set as described in chapter [Configure Proxy](#) in this document.

Microsoft Azure Subscription

An Azure account already exists and according logon credentials are known.

Topology Consideration and Networking

The topology and architecture of the SAP deployment in Azure needs to be defined. Architecture in regards to:

- Microsoft Azure Storage account(s) to be used
- Virtual Network to deploy the SAP system into
- Resource Group to deploy the SAP system into
- Azure Region to deploy the SAP system
- SAP configuration (2-tier or 3-tier)
- VM size(s) and number of additional VHDs to be mounted to the VM(s)
- SAP Transport and Correction System configuration

Azure Storage Accounts or Azure Virtual Networks as such should have been created and configured already. How to create and configure them is covered in the [Planning and Implementation Guide](#).

SAP Sizing

- The projected SAP workload has been determined, e.g. by using the SAP Quicksizer, and the according SAPS number is known
- The required CPU resource and memory consumption of the SAP system should be known
- The required I/O operations per second should be known
- The required network bandwidth in eventual communication between different VMs in Azure is known
- The required network bandwidth between the on-premises assets and the Azure deployed SAP systems is known

Resource Groups

Resource groups are part of a new deployment concept. All resources of a resource group have the same lifecycle, e.g. they are created and deleted at the same time. Read [this article](#) article for more information about resource groups.

SAP Resources

During the configuration work, the following resources are needed:

- SAP Note [1928533](#) contains:
 - the list of Azure Virtual Machine sizes, which are supported for the deployment of SAP software
 - important capacity information per Azure Virtual Machine size
 - supported SAP software and OS and DB combinations
 - the required SAP kernel version for Windows and Linux on Microsoft Azure
- SAP Note [2015553](#) lists prerequisites to be supported by SAP when deploying SAP software into Microsoft Azure.
- SAP Note [2178632](#) contains detail information on all monitoring metrics reported for SAP on Microsoft Azure.
- SAP Note [1409604](#) contains the required SAP Host Agent version for Windows on Microsoft Azure.
- SAP Note [2191498](#) contains the required SAP Host Agent version for Linux on Microsoft Azure.
- SAP Note [2243692](#) contains information about licensing for SAP on Linux on Azure
- SAP Note [1984787](#) contains general information about SUSE LINUX Enterprise Server 12
- SAP Note [2002167](#) contains general information about Red Hat Enterprise Linux 7.x
- SAP Note [1999351](#) contains additional troubleshooting information for the Enhanced Azure Monitoring for SAP.
- [SAP WIKI](#) that contains all required SAP Notes for Linux
- SAP specific PowerShell cmdlets that are part of the [Azure PowerShell](#)
- SAP specific Azure CLI that are part of the [Azure CLI](#)
- [Microsoft Azure Portal](#)

The following guides cover the topic of SAP on Microsoft Azure as well:

- SAP NetWeaver on Azure Virtual Machines (VMs) – Planning and Implementation Guide
- SAP NetWeaver on Azure Virtual Machines (VMs) – Deployment Guide (this document)
- SAP NetWeaver on Azure Virtual Machines (VMs) – DBMS Deployment Guide
- SAP NetWeaver on Azure Virtual Machines (VMs) – High Availability Deployment Guide

Deployment Scenarios of VMs for SAP on Microsoft Azure

In this chapter, you learn the different ways of deployment and the single steps for each deployment type.

Deployment of VMs for SAP

Microsoft Azure offers multiple ways to deploy VMs and associated disks. Thereby it is very important to understand the differences since preparations of the VMs might differ dependent on the way of deployment. In general, we look into the following scenarios:

Deploying a VM out of the Azure Marketplace

You like to take a Microsoft or 3rd party provided image out of the Azure Marketplace to deploy your VM. After you deployed your VM on Microsoft Azure, you follow the same guidelines and tools to install the SAP software inside your VM as you would do in an on-premises environment. For installing the SAP software inside the Azure VM, SAP and Microsoft recommend to upload and store the SAP installation media in Azure VHDs or to create an Azure VM working as a 'File server' which contains all the necessary SAP installation media.

For more details see chapter [Scenario 1: Deploying a VM out of the Azure Marketplace for SAP](#).

Deploying a VM with a Custom Image

Due to specific patch requirements in regards to your OS or DBMS version, the provided images out of the Azure Marketplace might not fit your needs. Therefore, you might need to create a VM using your own 'private' OS/DB VM image which can be deployed several times afterwards. The steps to create a private image differ between a Windows and a Linux image.



To prepare a Windows image that can be used to deploy multiple virtual machines, the Windows settings (like Windows SID and hostname) must be abstracted/generalized on the on-premises VM. This can be done using sysprep as described on <https://msdn.microsoft.com/library/hh825084.aspx>.



To prepare a Linux image that can be used to deploy multiple virtual machines, some Linux settings must be abstracted/generalized on the on-premises VM. This can be done using waagent -deprovision as described in [this article](#) or in [this article](#).

You can set up your database content by either using the SAP Software Provision Manager to install a new SAP system, restore a database backup from a VHD that is attached to the virtual machine or directly restore a database backup from Azure Storage if the DBMS supports it. (see the [DBMS Deployment Guide](#)). If you have already installed an SAP system in your on-premises VM (especially for 2-tier systems), you can adapt the SAP system settings after the deployment of the Azure VM through the System Rename procedure supported by the SAP Software Provisioning Manager (SAP Note [1619720](#)). Otherwise you can install the SAP software later after the deployment of the Azure VM.

For more details see chapter [Scenario 2: Deploying a VM with a custom image for SAP](#).

Moving a VM from On-Premises to Microsoft Azure with a Non-Generalized Disk

You plan to move a specific SAP system from on-premises to Microsoft Azure. This can be done by uploading the VHD which contains the OS, the SAP binaries and eventual DBMS binaries plus the VHDs with the data and log files of the DBMS to Microsoft Azure. In opposite to the scenario described in chapter [Deploying a VM with](#)

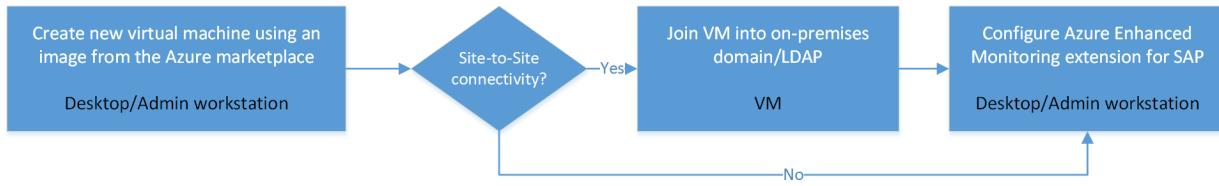
a custom image above, you keep the hostname, SAP SID and SAP user accounts in the Azure VM as they were configured in the on-premises environment. Therefore, generalizing the operating system is not necessary. This case will mostly apply for cross-premises scenarios where a part of the SAP landscape is run on-premises and parts on Microsoft Azure.

For more details see chapter [Scenario 3: Moving a VM from On-Premises Using a Non-Generalized Azure VHD with SAP](#).

Scenario 1: Deploying a VM out of the Azure Marketplace for SAP

Microsoft Azure offers the possibility to deploy a VM instance out of the Azure Marketplace, which offers some standard OS images of Windows Server and different Linux distributions. It is also possible to deploy an image that includes DBMS SKUs e.g. MS SQL Server. For details using those images with DBMS SKUs please refer to the [DBMS Deployment Guide](#).

The SAP specific sequence of steps deploying a VM out of the Azure Marketplace would look like:



Following the flowchart the following steps need to be executed:

Create Virtual Machine Using the Azure Portal

The easiest way to create a new virtual machine using an image from the Azure Marketplace is through the Azure Portal. Navigate to <https://portal.azure.com/#create> or click on "+" on the left side of the Azure Portal. Enter the type of operating system you want to deploy in the search field, e.g. Windows, SLES or RHEL and select the version. Make sure to select the deployment model "Resource Manager" and click Create.

The wizard will guide you through the required parameters to create the virtual machine along with all required resources like network interfaces or storage accounts. Some of these parameters are:

1. Basics
 - a. Name: The name of the resource, i.e. virtual machine name
 - b. Username and password/SSH public key: Enter the username and password of the user that is created during the provisioning. For a Linux virtual machine, you can also enter the public SSH key that you want to use to login to the machine using SSH.
 - c. Subscription: Select the subscription that you want to use to provision the new virtual machine.
 - d. Resource Group: The name of the resource group. You can either insert the name of a new resource group or of a resource group that already exists
 - e. Location: Select the location where the new virtual machine should be deployed. If you want to connect the virtual machine to your on-premises network, make sure to select the location of the Virtual Network that connects Azure to your on-premises network. For more details, see chapter [Microsoft Azure Networking](#) in the [Planning Guide](#).

2. Size

Check SAP Note [1928533](#) for a list of supported VM types. Please also make sure to select the correct type if you want to use Premium Storage. Not all VM types support Premium Storage. See chapter [Storage: Microsoft Azure Storage and Data Disks](#) and [Azure Premium Storage](#) in the [Planning Guide](#) for more details.

3. Settings

- a. Storage Account: Select an existing storage account or create a new one. Please read chapter [Microsoft Azure Storage](#) of the [DBMS Guide](#) for more details on the different storage types. Note that

not all storage types are supported for running SAP applications.

- b. Virtual network and Subnet: Select the virtual network that is connected to your on-premises network if you want to integrate the virtual machine into your intranet.
 - c. Public IP address: Select the Public IP address that you want to use or enter the parameters to create a new Public IP address. You can use a Public IP address to access your virtual machine over the internet. Make sure to also create a Network Security Group to filter access to your virtual machine.
 - d. Network Security Group: see [What is a Network Security Group \(NSG\)](#) for more details.
 - e. Availability: Select an Availability Set or enter the parameters to create a new Availability Set. For more information see chapter [Azure Availability Sets](#).
 - f. Monitoring: You can disable the diagnostics setting. It will be enabled automatically when you run the commands to enable the Azure Enhanced Monitoring as described in chapter [Configure Monitoring](#).
4. Summary: Validate the information provided on the summary page and click OK.

After you finished the wizard, your virtual machine will be deployed in the resource group you selected.

Create Virtual Machine Using a Template

You can also create a deployment using one of the SAP templates published in the [azure-quickstart-templates github repository](#). Or you can create a virtual machine using the [Azure Portal](#), [PowerShell](#) or [Azure CLI](#) manually.

- [2-tier configuration \(only one virtual machine\) template \(sap-2-tier-marketplace-image\)](#) Use this template if you want to create a 2-tier system using only one virtual machine.
- [3-tier configuration \(multiple virtual machines\) template \(sap-3-tier-marketplace-image\)](#) Use this template if you want to create a 3-tier system using multiple virtual machines.

After you opened one of the templates above, the Azure Portal navigates to a screen on which you can enter the parameters for the template. Enter the following information:

1. Basics

- **Subscription:** The subscription you want to deploy the template to
- **Resource group:** The resource group you want to deploy the template to. You can create a new resource group or select an existing resource group in the selected subscription.
- **Location:** The location you want to deploy the template to. If you selected an existing resource group, the location of the resource group is used.

2. Settings

- **Sap System Id:** SAP System ID
- **Os Type:** Operating system you want to deploy, e.g. Windows Server 2012 R2, SLES 12 or RHEL 7.2
 - The list does not contain all supported operating systems e.g. the list does not contain Windows Server 2008 R2 although it is supported by SAP. Please read SAP Note [1928533](#) for a list of all supported operating systems.
- **Sap System Size:** the size of the SAP system
 - The amount of SAPS the new system will provide. If you are not sure how many SAPS the system will require, please ask your SAP Technology Partner or System Integrator
- **System Availability:** (3-tier template only) System Availability
 - Select HA for a configuration that is suitable for a HA installation. Two database servers and two servers for the ASCS will be created.
- **Storage Type:** (2-tier template only) Type of storage that should be used
 - For bigger systems, the usage of Premium Storage is highly recommended. For more information about the different storage types, read
 - [Use of Azure Premium SSD Storage for SAP DBMS Instance](#)
 - [Microsoft Azure Storage](#) of the DBMS Guide

- [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#)
 - [Introduction to Microsoft Azure Storage](#)
- **Admin Username** and **Admin Password**: Username and password
 - A new user is created that can be used to log on to the machine.
- **New Or Existing Subnet**: Determines whether a new virtual network and subnet should be created or an existing subnet should be used. If you already have a virtual network that is connected to your on-premises network, select existing.
- **Subnet Id**: The ID of the subnet to which the virtual machines should be connected to. Select the subnet of your VPN or Express Route virtual network to connect the virtual machine to your on-premises network. The ID usually looks like /subscriptions/ <subscription id>/resourceGroups/ <resource group name>/providers/Microsoft.Network/virtualNetworks/ <virtual network name>/subnets/ <subnet name>

3. Terms and Conditions

Review the legal terms and accept them.

After you entered all parameters, confirm the screen by clicking Purchase.

Please note that the Azure VM Agent is deployed by default when using an image from the Azure Marketplace.

Configure Proxy Settings

Depending on your on-premises network configuration, it might be required to configure the proxy on your virtual machine if it is connected to your on-premises network via VPN or Express Route. Otherwise the virtual machine might not be able to access the Internet and therefore cannot download the required extensions or collect monitoring data. Please see chapter [Configure Proxy](#) of this document.

Join Domain (Windows only)

In the case that the deployment in Azure is connected to the on-premises AD/DNS via Azure Site-to-Site or Express Route (also referenced as Cross-Premises in the [Planning and Implementation Guide](#)), it is expected that the VM is joining an on-premises domain. Considerations of this step are described in chapter [Join VM into on-premises Domain \(Windows only\)](#) of this document.

Configure Monitoring

In order to ensure an SAP supported environment, configure the Azure Enhanced Monitoring Extension for SAP as described in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#) of this document.

Check the prerequisites for SAP Monitoring for required minimum versions of SAP Kernel and SAP Host Agent in the resources listed in chapter [SAP Resources](#) of this document.

Monitoring Check

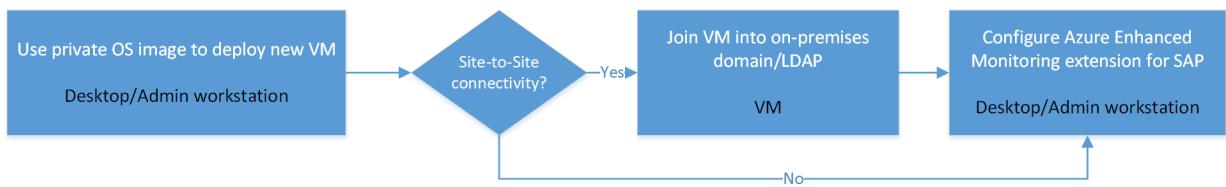
Check if the monitoring is working as described in chapter [Checks and Troubleshooting for End-to-End Monitoring Setup for SAP on Azure](#).

Post Deployment Steps

After creating the VM, it will be deployed and you need to install all the necessary software components into the VM. Hence this type of VM deployment would require either the software to be installed being already available in Microsoft Azure in some other VM or as a disk which can be attached. Or we are looking into cross-premises scenarios where connectivity to the on-premises assets (install shares) is given.

Scenario 2: Deploying a VM with a Custom Image for SAP

As described in the [Planning and Implementation Guide](#), you can prepare and create a custom image and use it to create multiple new VMs. The sequence of steps in the flow chart would look like:



Following the flowchart the following steps need to be executed:

Create the Virtual Machine

To create a deployment using a private OS image through the Azure Portal, use one of the SAP templates listed below. These templates are published at the [azure-quickstart-templates github repository](#). You can also create a virtual machine using the [PowerShell](#) manually.

- [2-tier configuration \(only one virtual machine\) template \(sap-2-tier-user-image\)](#) Use this template if you want to create a 2-tier system using only one virtual machine and your own OS image.
- [3-tier configuration \(multiple virtual machines\) template \(sap-3-tier-user-image\)](#) Use this template if you want to create a 3-tier system using multiple virtual machines and your own OS image.

After you opened one of the templates above, the Azure Portal navigates to a screen on which you can enter the parameters for the template. Enter the following information:

1. Basics

- **Subscription:** The subscription you want to deploy the template to
- **Resource group:** The resource group you want to deploy the template to. You can create a new resource group or select an existing resource group in the selected subscription.
- **Location:** The location you want to deploy the template to. If you selected an existing resource group, the location of the resource group is used.

2. Settings

- **Sap System Id:** SAP System ID
- **Os Type:** Operating system type you want to deploy, Windows or Linux
- **Sap System Size:** the size of the SAP system
 - The amount of SAPS the new system will provide. If you are not sure how many SAPS the system will require, please ask your SAP Technology Partner or System Integrator
- **System Availability:** (3-tier template only) System Availability
 - Select HA for a configuration that is suitable for a HA installation. Two database servers and two servers for the ASCS will be created.
- **Storage Type:** (2-tier template only) Type of storage that should be used
 - For bigger systems, the usage of Premium Storage is highly recommended. For more information about the different storage types, read
 - [Use of Azure Premium SSD Storage for SAP DBMS Instance](#)
 - [Microsoft Azure Storage of the DBMS Guide](#)
 - [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#)
 - [Introduction to Microsoft Azure Storage](#)
- **User Image Vhd Uri:** URI of the private OS image vhd e.g. <https://<accountname>.blob.core.windows.net/vhds/userimage.vhd>
- **User Image Storage Account:** Name of the storage account where the private OS image is stored e.g. <accountname> in the example URI above
- **Admin Username** and **Admin Password:** Username and password
 - A new user is created that can be used to log on to the machine.
- **New Or Existing Subnet:** Determines whether a new virtual network and subnet should be created or an existing subnet should be used. If you already have a virtual network that is connected to your

on-premises network, select existing.

- **Subnet Id:** The ID of the subnet to which the virtual machines should be connected to. Select the subnet of your VPN or Express Route virtual network to connect the virtual machine to your on-premises network. The ID usually looks like /subscriptions/ <subscription id>/resourceGroups/ <resource group name>/providers/Microsoft.Network/virtualNetworks/ <virtual network name>/subnets/ <subnet name>

3. Terms and Conditions

Review the legal terms and accept them.

After you entered all parameters, confirm the screen by clicking Purchase.

Install VM Agent (Linux only)

The Linux Agent must already be installed in the user image if you want to use the templates above. Otherwise the deployment will fail. Download and install the VM Agent in the user image as described in chapter [Download, Install and Enable Azure VM Agent](#) of this document. If you don't use the templates above, you can also install the VM Agent afterwards.

Join Domain (Windows only)

In the case that the deployment in Azure is connected to the on-premises AD/DNS via Azure Site-to-Site or Express Route (also referenced as Cross-Premises in the [Planning and Implementation Guide](#)), it is expected that the VM is joining an on-premises domain. Considerations of this step are described in chapter [Join VM into On-Premises Domain \(Windows only\)](#) of this document.

Configure Proxy Settings

Depending on your on-premises network configuration, it might be required to configure the proxy on your virtual machine if it is connected to your on-premises network via VPN or Express Route. Otherwise the virtual machine might not be able to access the internet and therefore cannot download the required extensions or collect monitoring data. Please see chapter [Configure Proxy](#) of this document.

Configure Monitoring

In order to ensure an SAP supported environment, configure the Azure Monitoring Extension for SAP as described in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#) of this document. Check the prerequisites for SAP Monitoring for required minimum versions of SAP Kernel and SAP Host Agent in the resources listed in chapter [SAP Resources](#) of this document.

Monitoring Check

Check if the monitoring is working as described in chapter [Checks and Troubleshooting for End-to-End Monitoring Setup for SAP on Azure](#).

Scenario 3: Moving a VM from On-Premises Using a Non-Generalized Azure VHD with SAP

If you plan to move your SAP system in its current form and shape (same hostname and SAP SID) from on-premises to Azure, the VHD is directly used as the OS disk and not referenced as an image during deployment. In this case the VM Agent will not be automatically installed during the deployment. As the VM Agent and the Azure Enhanced Monitoring Extension for SAP is a prerequisite for SAP support, you need to download, install and enable both components manually after creation of the virtual machine.

For details on the Azure VM Agent, please check this article:

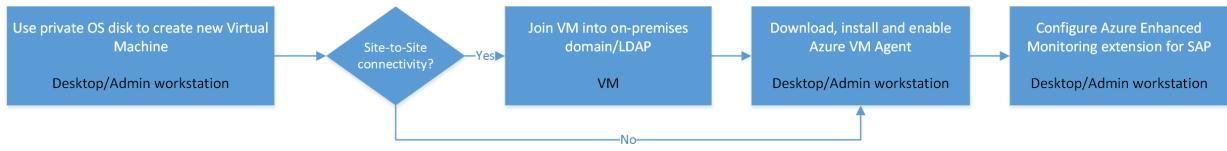


<http://blogs.msdn.com/b/wats/archive/2014/02/17/bginfo-guest-agent-extension-for-azure-vms.aspx>



[Azure Linux Agent User Guide](#)

The workflow of the different steps looks like:



Assuming that the disk is already uploaded and defined in Azure (see [Planning and Implementation Guide](#)), follow these steps:

Create Virtual Machine

To create a deployment using a private OS disk through the Azure Portal, use the SAP template published on the [azure-quickstart-templates github repository](#). You can also create a virtual machine using the [PowerShell](#) manually.

- [2-tier configuration \(only one virtual machine\) template \(sap-2-tier-user-disk\)](#)
 - Use this template if you want to create a 2-tier system using only one virtual machine.

After you opened one of the templates above, the Azure Portal navigates to a screen on which you can enter the parameters for the template. Enter the following information:

1. Basics

- **Subscription:** The subscription you want to deploy the template to
- **Resource group:** The resource group you want to deploy the template to. You can create a new resource group or select an existing resource group in the selected subscription.
- **Location:** The location you want to deploy the template to. If you selected an existing resource group, the location of the resource group is used.

2. Settings

- **Sap System Id:** SAP System ID
- **Os Type:** Operating system type you want to deploy, Windows or Linux
- **Sap System Size:** the size of the SAP system
 - The amount of SAPS the new system will provide. If you are not sure how many SAPS the system will require, please ask your SAP Technology Partner or System Integrator
- **Storage Type:** (2-tier template only) Type of storage that should be used
 - For bigger systems, the usage of Premium Storage is highly recommended. For more information about the different storage types, read
 - [Use of Azure Premium SSD Storage for SAP DBMS Instance](#)
 - [Microsoft Azure Storage](#) of the DBMS Guide
 - [Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads](#)
 - [Introduction to Microsoft Azure Storage](#)
- **Os Disk Vhd Uri:** URI of the private OS disk e.g. <https://<accountname>.blob.core.windows.net/vhds/osdisk.vhd>
- **New Or Existing Subnet:** Determines whether a new virtual network and subnet should be created or an existing subnet should be used. If you already have a virtual network that is connected to your on-premises network, select existing.
- **Subnet Id:** The ID of the subnet to which the virtual machines should be connected to. Select the subnet of your VPN or Express Route virtual network to connect the virtual machine to your on-premises network. The ID usually looks like /subscriptions/<subscription id>/resourceGroups/<resource group name>/providers/Microsoft.Network/virtualNetworks/<virtual network name>/subnets/<subnet name>

3. Terms and Conditions

Review the legal terms and accept them.

After you entered all parameters, confirm the screen by clicking Purchase.

Install VM Agent

The VM Agent has to be installed in the OS disk if you want to use the templates above. Otherwise the deployment will fail. Download and install the VM Agent in the VM as described in chapter [Download, Install and Enable Azure VM Agent](#) of this document.

If you don't use the templates above, you can also install the VM Agent afterwards.

Join Domain (Windows only)

In the case that the deployment in Azure is connected to the on-premises AD/DNS via Azure Site-to-Site or Express Route (also referenced as Cross-Premises in the [Planning and Implementation Guide](#)), it is expected that the VM is joining an on-premises domain. Considerations of this step are described in chapter [Join VM into on-premises Domain \(Windows only\)](#) of this document.

Configure Proxy Settings

Depending on your on-premises network configuration, it might be required to configure the proxy on your virtual machine if it is connected to your on-premises network via VPN or Express Route. Otherwise the virtual machine might not be able to access the internet and therefore cannot download the required extensions or collect monitoring data. Please see chapter [Configure Proxy](#) of this document.

Configure Monitoring

In order to ensure an SAP supported environment, configure the Azure Enhanced Monitoring Extension for SAP as described in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#) of this document.

Check the prerequisites for SAP Monitoring for required minimum versions of SAP kernel and SAP Host Agent in the resources listed in chapter [SAP Resources](#) of this document.

Monitoring Check

Check if the monitoring is working as described in chapter [Checks and Troubleshooting for End-to-End Monitoring Setup for SAP on Azure](#).

Scenario 4: Updating the Monitoring Configuration for SAP

In the following cases you need to update the SAP monitoring configuration:

- The joint MS/SAP team extended the monitoring capabilities and requests more or less counters.
- Microsoft introduces a new version of the underlying Azure infrastructure delivering the monitoring data, and the Azure Enhanced Monitoring Extension for SAP needs to be adapted to those changes.
- You mount additional VHDs to your Azure VM or you remove a VHD. In this case, you need to update the collection of storage related data. If you change your configuration by adding or deleting endpoints or assigning IP addresses to a VM, this will not impact the monitoring configuration.
- You change the size of your Azure VM e.g. from A5 to any other size of the VM.
- You add new network interfaces to your Azure VM

In order to update the monitoring configuration, proceed as follows:

- Update the monitoring infrastructure by following the steps explained in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#) of this document. A re-run of the script described in this chapter will detect that a monitoring configuration is deployed and will perform the necessary changes to the monitoring configuration.

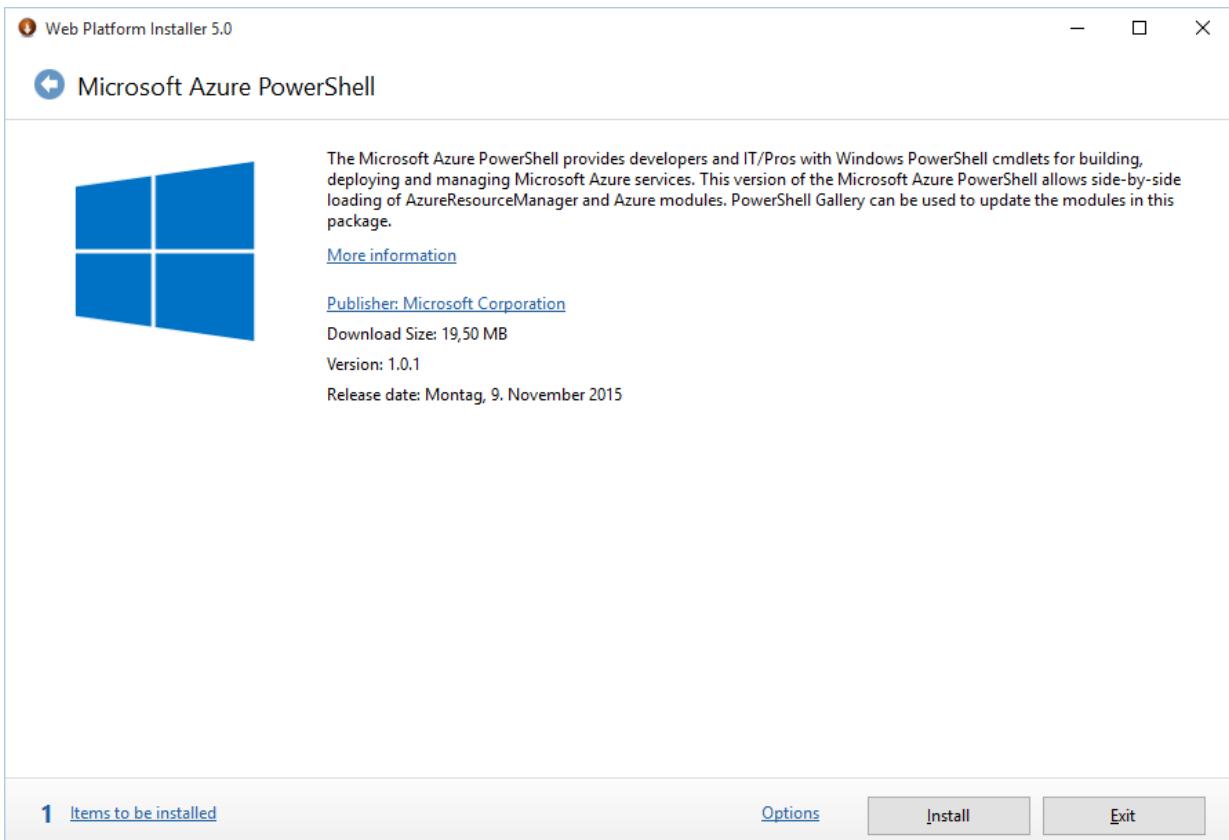
Detailed Single Deployment Steps

Deploying Azure PowerShell cmdlets

- Go to <https://azure.microsoft.com/downloads/>
- Under the section 'Command-line tools' there is a section called 'PowerShell'. Follow the 'Windows install'

link.

- Microsoft Download Manager will pop-up with a line item ending with .exe. Select the option 'Run'.
- A pop-up will come up asking whether to run Microsoft Web Platform Installer. Press YES.
- A screen like this one appears:



- Press Install and accept the EULA.

Check frequently whether the PowerShell cmdlets have been updated. Usually there are updates on a monthly period. The easiest way to do this is to follow the installation steps as described above up to the installation screen shown in [this figure](#). In this screen, the release date of the cmdlets is shown as well as the actual release number. Unless stated differently in SAP Notes [1928533](#) or [2015553](#), it is recommended to work with the latest version of Azure PowerShell cmdlets.

The current installed version of the Azure cmdlets on the desktop/laptop can be checked with the PS command:

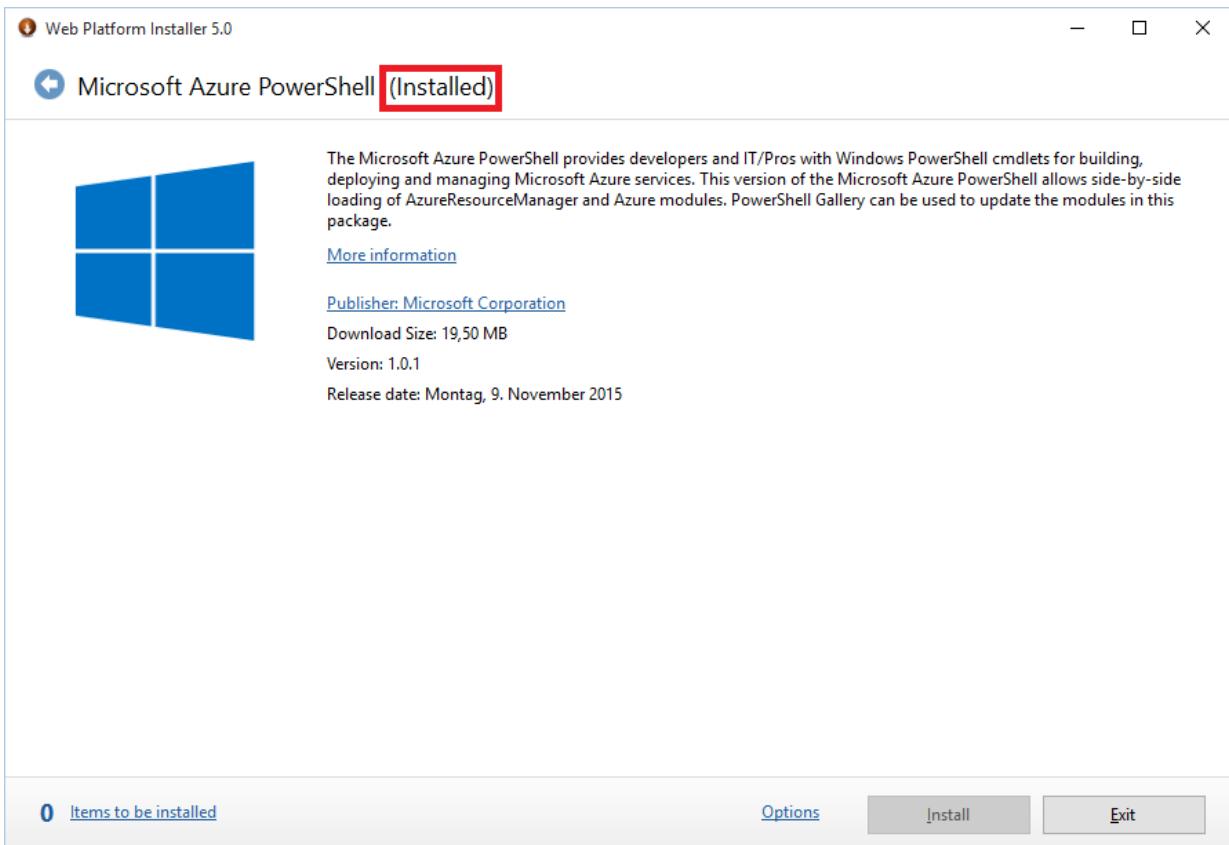
```
Import-Module Azure  
(Get-Module Azure).Version
```

The result should be presented as shown below in [this figure](#).



If the Azure cmdlet version installed on the desktop/laptop is the current one, the first screen after starting the Microsoft Web Platform Installer will look slightly different compared to the one shown in [this figure](#).

Please notice the red circle below in the [figure](#) below.



If the screen looks as [above](#), indicating that the most recent Azure cmdlet version is already installed, there is no need to continue with the installation. In this case you can 'Exit' the installation at this stage.

Deploying Azure CLI

- Go to <https://azure.microsoft.com/downloads/>
- Under the section 'Command-Line Tools' there is a section called 'Azure command-line interface'. Follow the Install link for your operating system.

Check frequently whether the Azure CLI have been updated. Usually there are updates on a monthly period. The easiest way to do this is to follow the installation steps as described above.

The current installed version of Azure CLI on the desktop/laptop can be checked with the command:

```
azure --version
```

The result should be presented as shown below in [this](#) figure.

```
C:\>azure --version
0.9.11 (node: 0.12.7)
C:\>
```

Join VM into On-Premises Domain (Windows only)

If you deploy SAP VMs into a cross-premises scenario where on-premises AD and DNS is extended into Azure, it is expected that the VMs join an on-premises domain. The detailed steps of joining a VM to an on-premises domain and additional software required to be a member of an on-premises domain is customer dependent. Usually joining a VM to an on-premises domain means installing additional software like Malware Protection software or various agents of backup or monitoring software.

Additionally, you need to make sure for cases where Internet proxy settings are forced when joining a domain,

that the Windows Local System Account(S-1-5-18) in the Guest VM has these settings as well. Easiest is to force the proxy with Domain Group Policy which apply to systems within the domain.

Download, Install and Enable Azure VM Agent

For virtual machines that are deployed from an OS image that is not generalized (e.g. not sysprep'ed for Windows), you need to download, install and enable the Azure VM Agent manually.

For virtual machines deployed from the Azure marketplace, this step is not required as these images already contain the Azure VM Agent.

Windows

- Download the Azure VM Agent:
 - Download the Azure VM Agent installer package from: <https://go.microsoft.com/fwlink/?LinkId=394789>
 - Store the VM Agent MSI package locally on the laptop or a server
- Install the Azure VM Agent:
 - Connect to the deployed Azure VM with Remote Desktop (RDP)
 - Open a Windows Explorer window on the VM and choose the target directory for the MSI file of the VM Agent
 - Drag and drop the Azure VM Agent Installer MSI file from your local laptop/server into the target directory of the VM Agent in the VM
 - Double-click on the MSI file in the VM
 - For VMs joined to on-premises domains, make sure that eventual Internet proxy settings apply for the Windows Local System account (S-1-5-18) in the VM as well as described in chapter [Configure Proxy](#). The VM Agent will run in this context and needs to be able to connect to Azure.

The update of the Azure VM Agent requires no user intervention. VM Agent auto updates itself and does not require a VM reboot.

Linux

Please install the VM Agent for Linux using the following command

- **SLES**

```
sudo zypper install WALinuxAgent
```

- **RHEL**

```
sudo yum install WALinuxAgent
```

Please follow the steps in [this article](#) to update the Azure Linux Agent if it is already installed.

Configure Proxy

The steps for configuring the proxy differ between Windows and Linux.

Windows

These settings must also be valid for the LocalSystem account to access the Internet. If your proxy settings are not set by group policy, you can configure them for the LocalSystem account. Follow these steps:

1. Open gredit.msc
2. Navigate to Computer Configuration -> Administrative Templates -> Windows Components -> Internet Explorer. Make sure that the policy setting "Make proxy settings per-machine (rather than per-user)" is set to Enabled.
3. Open the Control Panel and navigate to Network and Internet -> Internet Options

4. Open the Connections tab and click on LAN settings
5. Disable "Automatically detect settings"
6. Enable "Use a proxy server for your LAN" and enter the proxy address and port

Linux

Configure the correct proxy in the configuration file of the Microsoft Azure Guest Agent, which is located at /etc/waagent.conf. The following parameters must be set:

```
HttpProxy.Host=<proxy host e.g. proxy.corp.local>
HttpProxy.Port=<port of the proxy host e.g. 80>
```

Restart the agent after you have changed its configuration with

```
sudo service waagent restart
```

The proxy settings in /etc/waagent.conf do also apply for the required VM extensions. If you want to use the Azure repositories, make sure that the traffic to these repositories is not going through the on-premises intranet. If you created User Defined Routes to enable Forced Tunneling, make sure to add a route that routes traffic to the repositories directly to the Internet and not through your site-to-site connection.

- **SLES** You also need to add routes for the IP addresses listed in /etc/regionserverclnt.cfg. An example is shown in the screenshot below.
- **RHEL** You also need to add routes for the IP addresses of the hosts listed in /etc/yum.repos.d/rhui-load-balancers. An example is shown in the screenshot below.

For more details about User Defined Routes, see [this article](#).

| NAME | ADDRESS PREFIX | NEXT HOP | |
|-----------|--------------------|-------------------------|-----|
| Default | 0.0.0.0/0 | Virtual network gateway | ... |
| NTP | 23.101.187.68/32 | Internet | ... |
| RHEL | 104.47.166.223/32 | Internet | ... |
| RHEL2 | 13.79.158.41/32 | Internet | ... |
| RHEL3 | 104.45.4.188/32 | Internet | ... |
| RHEL4 | 104.47.154.145/32 | Internet | ... |
| SUSE | 104.45.17.148/32 | Internet | ... |
| SUSE1 | 104.45.154.114/32 | Internet | ... |
| SUSE2 | 104.45.31.195/32 | Internet | ... |
| SUSE3 | 191.237.254.253/32 | Internet | ... |
| SUSE4 | 23.100.36.229/32 | Internet | ... |
| SUSE5 | 23.101.26.184/32 | Internet | ... |
| SUSE6 | 104.45.19.56/32 | Internet | ... |
| TESIFile | 168.61.58.146/32 | Internet | ... |
| TESIFile2 | 168.61.61.210/32 | Internet | ... |

Configure Azure Enhanced Monitoring Extension for SAP

Once the VM is prepared as described in chapter [Deployment Scenarios of VMs for SAP on Microsoft Azure](#), the Azure VM Agent is installed in the virtual machine. As the next important step, you deploy the Azure Enhanced Monitoring Extension for SAP, which is available in the Azure Extension Repository in the global datacenters of Microsoft Azure. For more details, please check the [Planning and Implementation Guide](#).

You can use Azure PowerShell or Azure CLI to install and configure the Azure Enhanced Monitoring Extension for SAP. Please read chapter [Azure PowerShell](#) if you want to install the extension on a Windows or Linux VM using a Windows machine. For installing the extension on a Linux VM using a Linux desktop read chapter [Azure CLI](#).

Azure PowerShell for Linux and Windows VMs

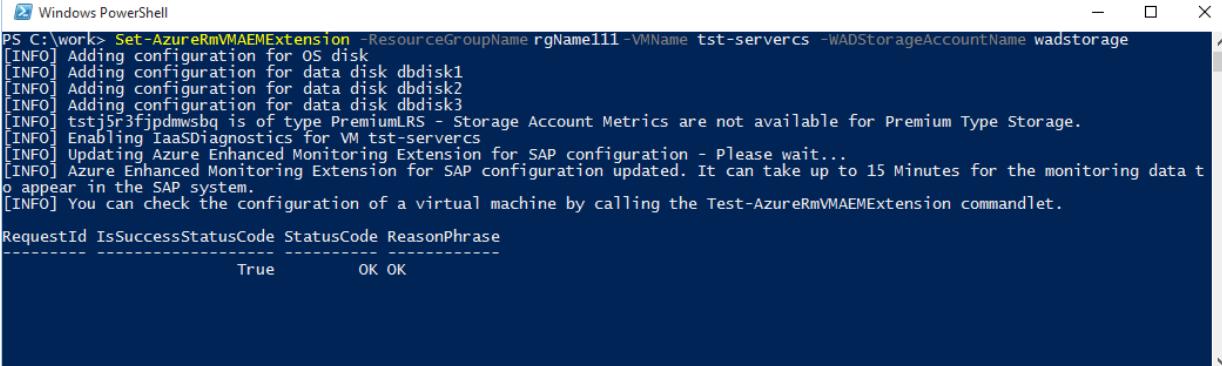
In order to perform the task of installing the Azure Enhanced Monitoring Extension for SAP, perform the following steps:

- Make sure that you have installed the latest version of the Microsoft Azure PowerShell cmdlet. See chapter [Deploying Azure PowerShell cmdlets](#) of this document.
- Run the following PowerShell cmdlet. For a list of available environments, run commandlet Get-AzureRmEnvironment. If you want to use public Azure, your environment is AzureCloud. For Azure in China, select AzureChinaCloud.

```
$env = Get-AzureRmEnvironment -Name <name of the environment>
Login-AzureRmAccount -Environment $env
Set-AzureRmContext -SubscriptionName <subscription name>

Set-AzureRmVMAEMExtension -ResourceGroupName <resource group name> -VMName <virtual machine name>
```

After you provided your account data and the Azure Virtual Machine, the script will deploy the required extensions and enable the required features. This can take several minutes. Please read [this MSDN article](#) for more information about the Set-AzureRmVMAEMExtension.



```
PS C:\work> Set-AzureRmVMAEMExtension -ResourceGroupName rgName111 -VMName tst-servercs -WADStorageAccountName wadstorage
[INFO] Adding configuration for OS disk
[INFO] Adding configuration for data disk dbdisk1
[INFO] Adding configuration for data disk dbdisk2
[INFO] Adding configuration for data disk dbdisk3
[INFO] tstj3r3fjpdmsbq is of type PremiumLRS - Storage Account Metrics are not available for Premium Type Storage.
[INFO] Enabling IaaSDiagnostics for VM tst-servercs
[INFO] Updating Azure Enhanced Monitoring Extension for SAP configuration - Please wait...
[INFO] Azure Enhanced Monitoring Extension for SAP configuration updated. It can take up to 15 Minutes for the monitoring data to appear in the SAP system.
[INFO] You can check the configuration of a virtual machine by calling the Test-AzureRmVMAEMExtension commandlet.

RequestId IsSuccessStatusCode StatusCode ReasonPhrase
----- -----
True      OK    OK
```

A successful run of Set-AzureRmVMAEMExtension will do all the steps necessary to configure the host monitoring functionality for SAP.

The output the script should deliver looks like:

- Confirmation that the monitoring configuration for the Base VHD (containing the OS) plus all additional VHDS mounted to the VM has been configured.
- The next two messages are confirming the configuration of Storage Metrics for a specific storage account.
- One line of output will give a status on the actual update of the monitoring configuration.
- Another will show up confirming that the configuration has been deployed or updated.
- The last line of the output is informational showing the possibility to test the monitoring configuration.
- To check, that all steps of the Azure Enhanced Monitoring have been executed successfully and that the Azure Infrastructure provides the necessary data, proceed with the Readiness Check for Azure Enhanced Monitoring Extension for SAP, as described in chapter [Readiness Check for Azure Enhanced Monitoring for SAP](#) in this document.
- To continue doing this, wait for 15-30 minutes until the Azure Diagnostics will have the relevant data collected.

Azure CLI for Linux VMs

In order to perform the task of installing the Azure Enhanced Monitoring Extension for SAP using Azure CLI, perform the following steps:

1. Install Azure CLI as described in [this article](#)
2. Login with your Azure account

```
azure login
```

3. Switch to Azure Resource Manager mode

```
azure config mode arm
```

4. Enable Azure Enhanced Monitoring

```
azure vm enable-aem <resource-group-name> <vm-name>
```

5. Verify that the Azure Enhanced Monitoring is active on the Azure Linux VM. Check if the file /var/lib/AzureEnhancedMonitor/PerfCounters exists. If exists, display information collected by AEM with:

```
cat /var/lib/AzureEnhancedMonitor/PerfCounters
```

Then you will get output like:

```
2;cpu;Current Hw Frequency ;0;2194.659;MHz;60;1444036656;saplnxmon;
2;cpu;Max Hw Frequency ;0;2194.659;MHz;0;1444036656;saplnxmon;
...
...
```

Checks and Troubleshooting for End-to-End Monitoring Setup for SAP on Azure

After you have deployed your Azure VM and set up the relevant Azure monitoring infrastructure, check whether all the components of the Azure Enhanced Monitoring are working in a proper way.

Therefore, execute the Readiness Check for the Azure Enhanced Monitoring Extension for SAP as described in chapter [Readiness Check for Azure Enhanced Monitoring for SAP](#). If the result of this check is positive and you get all relevant performance counters, the Azure monitoring has been setup successfully. In this case, proceed with the installation of the SAP Host Agent as described in the SAP Notes listed in chapter [SAP Resources](#) of this document. If the result of the Readiness Check indicates missing counters, proceed executing the Health check for the Azure Monitoring Infrastructure as described in chapter [Health Check for Azure Monitoring Infrastructure Configuration](#). In case of any problem with the Azure Monitoring Configuration, check chapter [Further Troubleshooting of Azure Monitoring Infrastructure for SAP](#) for further help on troubleshooting.

Readiness Check for Azure Enhanced Monitoring for SAP

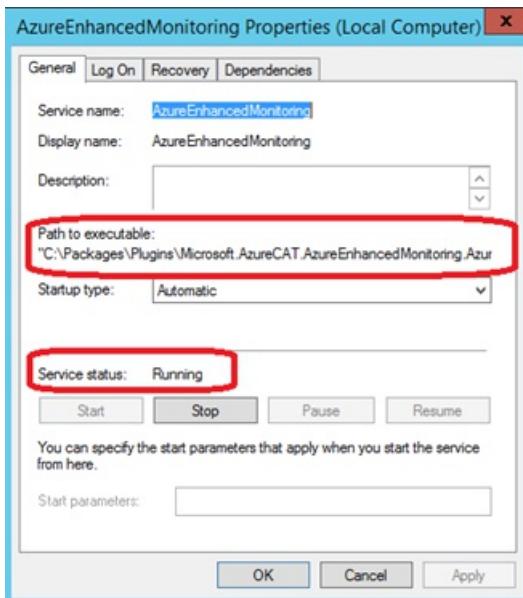
With this check, you make sure that the metrics which will be shown inside your SAP application are provided completely by the underlying Azure Monitoring Infrastructure.

Execute the Readiness Check on a Windows VM

In order to execute the Readiness Check, logon to the Azure Virtual Machine (admin account is not necessary) and execute the following steps:

- Open a Windows Command prompt and change to the installation folder of the Azure Monitoring Extension for SAP C:\Packages\Plugins\Microsoft.AzureCAT.AzureEnhancedMonitoring.AzureCATEExtensionHandler\<version>\drop

The version part provided in the path to the monitoring extension above may vary. If you see multiple folders of the monitoring extension version in the installation folder, check the configuration of the Windows service 'AzureEnhancedMonitoring' and switch to the folder indicated as 'Path to executable'.



- Execute azperflib.exe in the command window without any parameters.

NOTE

The azperflib.exe runs in a loop and updates the collected counters every 60 seconds. In order to finish the loop, close the command window.

If the Azure Enhanced Monitoring Extension is not installed or the service 'AzureEnhancedMonitoring' is not running, the extension has not been configured correctly. In this case, follow chapter [Further Troubleshooting of Azure Monitoring Infrastructure for SAP](#) for detailed instructions how to redeploy the extension.

Check the output of azperflib.exe

The output of azperflib.exe shows all populated Azure performance counters for SAP. At the bottom of the list of collected counters, you find a summary and a health indicator, which indicates the status of the Azure Monitoring.

```
Administrator: Command Prompt - azperflib.exe
category: vm name: SLA Max Disk IOPS per VM Ops/sec
value: 6400 <long> unit: none
refreshInterval: 0 sec UTC time: Wed Oct 19 13:31:39 2016

category: disk name: SLA Ops/sec instance: 0 C:
value: 500 <long> unit: none
refreshInterval: 0 sec UTC time: Wed Oct 19 13:31:40 2016

category: disk name: Read Ops/sec instance: 0 C:
value: 0.000000 <double> unit: ops/s
refreshInterval: 60 sec UTC time: Wed Oct 19 14:23:41 2016

category: disk name: Write Ops/sec instance: 0 C:
value: 1.682917 <double> unit: ops/s
refreshInterval: 60 sec UTC time: Wed Oct 19 14:23:41 2016

category: disk name: Read Response Time sec instance: 0 C:
value: 0.000000 <double> unit: s
refreshInterval: 60 sec UTC time: Wed Oct 19 14:23:41 2016

category: disk name: Write Response Time sec instance: 0 C:
value: 0.000265 <double> unit: s
refreshInterval: 60 sec UTC time: Wed Oct 19 14:23:41 2016

category: disk name: Caching instance: 0 C:
value: 'ReadWrite' <string> unit: none
refreshInterval: 0 sec UTC time: Wed Oct 19 13:31:40 2016

category: disk name: Storage Type instance: 0 C:
value: 'Premium' <string> unit: none
refreshInterval: 0 sec UTC time: Wed Oct 19 13:31:40 2016

API Calls total: 54, not available: 11
Counters total: 43, empty: 0
Health status: OK
Diagnostics: OK
Sleeping 60 sec...
```

Check the result returned for the output of the amount of 'Counters total', which are reported as empty and for the 'Health check' as shown above in the figure [above](#).

You can interpret the result values as follows:

| AZPERFLIB.EXE RESULT VALUES | AZURE MONITORING HEALTH STATUS |
|----------------------------------|---|
| API Calls - not available | Counters that are not available can either be not applicable to the virtual machine configuration or errors - see Health status |
| Counters total: empty | The following 2 Azure storage counters can be empty: <ul style="list-style-type: none">• Storage Read Op Latency Server msec• Storage Read Op Latency E2E msec All other counters must contain values. |
| Health check | Only OK if return status shows OK |
| Diagnostics | Detailed information about health status |

If the Health check value is not OK, follow the instructions of the Health check for the Azure Monitoring Infrastructure Configuration as described in chapter [Health check for Azure Monitoring Infrastructure Configuration](#) below.

Execute the Readiness Check on a Linux VM

In order to execute the readiness check, connect with SSH to the Azure Virtual Machine and execute the following steps:

- Check the output of the Azure Enhanced Monitoring Extension
 - more /var/lib/AzureEnhancedMonitor/PerfCounters
 - Should give you a list of performance counters. The file should not be empty
 - cat /var/lib/AzureEnhancedMonitor/PerfCounters | grep Error
 - Should return one line where the error is none e.g. 3;config;Error;0;0;**none**;0;1456416792;tst-servercs;
 - more /var/lib/AzureEnhancedMonitor/LatestErrorRecord
 - Should be empty or should not exist
- If the first check above was not successful, perform these additional tests:
 - Make sure that the waagent is installed and started
 - sudo ls -al /var/lib/waagent/
 - should list the content of the waagent directory
 - ps -ax | grep waagent
 - should show one entry similar to 'python /usr/sbin/waagent -daemon'
 - Make sure that the Linux Diagnostic extension is installed and started
 - sudo sh -c 'ls -al /var/lib/waagent/Microsoft.OSTCExtensions.LinuxDiagnostic-*'
 - should list the content of the Linux Diagnostic Extension directory
 - ps -ax | grep diagnostic
 - should show one entry similar to 'python /var/lib/waagent/Microsoft.OSTCExtensions.LinuxDiagnostic-2.0.92/diagnostic.py -daemon'
 - Make sure that the Azure Enhanced Monitoring Extension is installed and started
 - sudo sh -c 'ls -al /var/lib/waagent/Microsoft.OSTCExtensions.AzureEnhancedMonitorForLinux-*'
 - should list the content of the Azure Enhanced Monitoring Extension directory
 - ps -ax | grep AzureEnhanced

- should show one entry similar to 'python /var/lib/waagent/Microsoft.OSTCExtensions.AzureEnhancedMonitorForLinux-2.0.0.2/handler.py daemon'
- Install the SAP Host Agent as described in SAP Note [1031096](#) and check the output of saposcol
 - Execute /usr/sap/hostctrl/exe/saposcol -d
 - Execute dump ccm
 - Check if the metric "Virtualization_Configuration\Enhanced Monitoring Access" is true
- If you already have a SAP NetWeaver ABAP application server installed, open transaction ST06 and check if the enhanced monitoring is enabled.

If any of the checks above fail, follow chapter [Further troubleshooting of Azure Monitoring infrastructure for SAP](#) for detailed instructions how to redeploy the extension.

Health check for Azure Monitoring Infrastructure Configuration

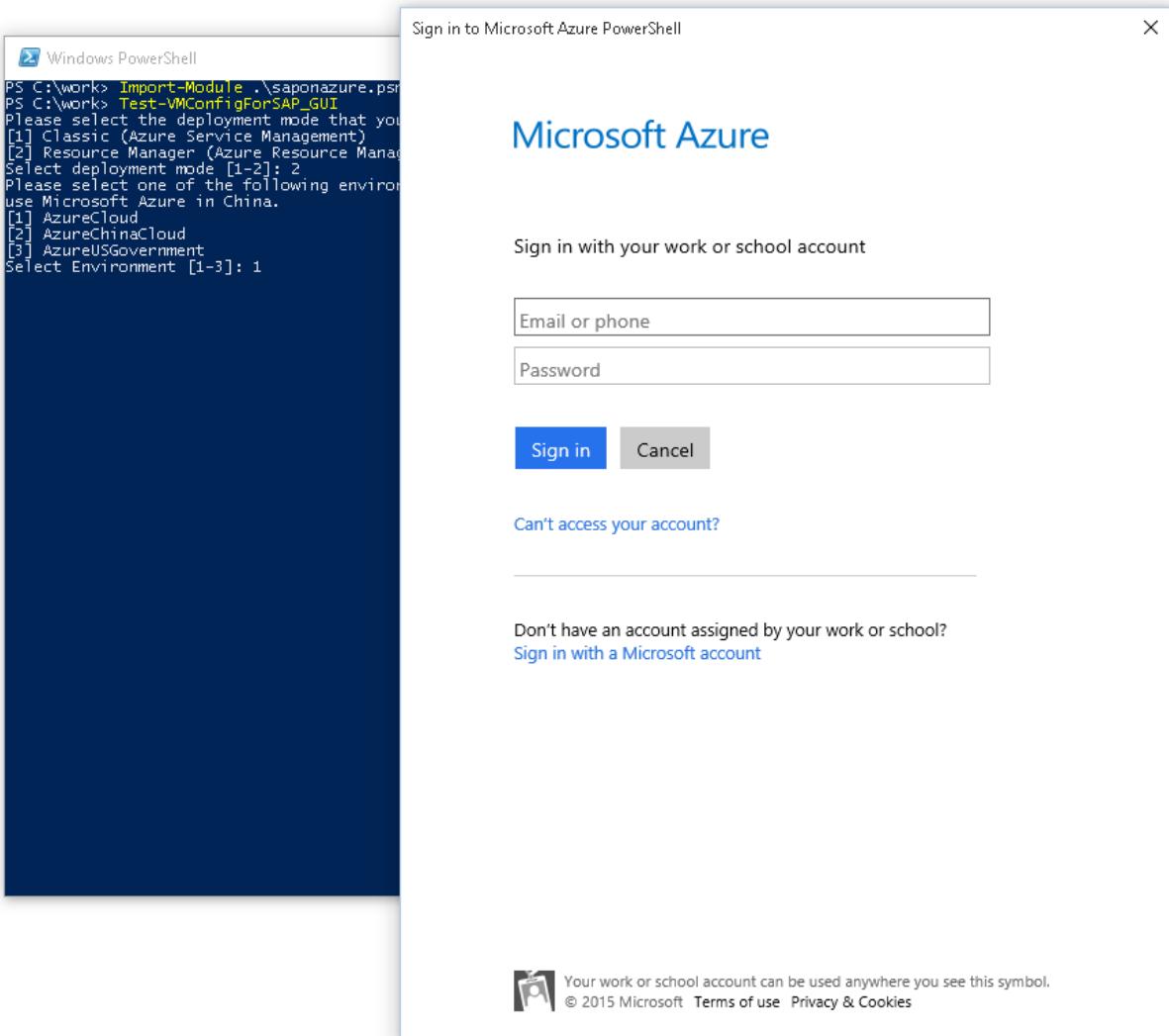
If some of the monitoring data is not delivered correctly as indicated by the test described in chapter [Readiness Check for Azure Enhanced Monitoring for SAP](#) above, execute the Test-AzureRmVMAEMExtension cmdlet to test if the current configuration of the Azure Monitoring infrastructure and the Monitoring extension for SAP is correct.

In order to test the monitoring configuration, please execute the following sequence:

- Make sure that you have installed the latest version of the Microsoft Azure PowerShell cmdlet as described in chapter [Deploying Azure PowerShell cmdlets](#) of this document.
- Run the following PowerShell cmdlet. For a list of available environments, run commandlet Get-AzureRmEnvironment. If you want to use public Azure, your environment is AzureCloud. For Azure in China, select AzureChinaCloud.

```
$env = Get-AzureRmEnvironment -Name <name of the environment>
Login-AzureRmAccount -Environment $env
Set-AzureRmContext -SubscriptionName <subscription name>
Test-AzureRmVMAEMExtension -ResourceGroupName <resource group name> -VMName <virtual machine name>
```

- After you provided your account data and the Azure Virtual Machine, the script will test the configuration of the virtual machine you choose.



After you entered the information about your account and the Azure Virtual Machine, the script will test the configuration of the virtual machine you choose.

```
PS C:\work> Test-AzureRmVMAEMExtension -ResourceGroupName rgName111 -VMName tst-servercs
VM Existence check for tst-servercs ...OK
VM Guest Agent check...OK
Azure Enhanced Monitoring Extension for SAP Installation check...OK
Storage Metrics check...
    Storage Metrics check for tstj5r3fjpdmwsbq...
        Storage Metrics not available for Premium Storage account tstj5r3fjpdmwsbq...OK
Azure Enhanced Monitoring Extension for SAP public configuration check...
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Size...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Memory...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM CPU...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: Script Version...OK
OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM SLA IOPS...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM SLA Throughput...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: WAD name...OK
OK
Azure Enhanced Monitoring Extension for SAP public configuration check: WAD URI...OK
OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM OS Disk Type...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM OS Disk SLA IOPS...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM OS Disk SLA Throughput...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM OS disk name...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Data Disk 1 LUN...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Data Disk 1 Type...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Data Disk 1 SLA IOPS...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Data Disk 1 SLA Throughput...OK
Azure Enhanced Monitoring Extension for SAP public configuration check: VM Data Disk 1 name...OK
```

Make sure that every check is marked with OK. If some of the checks are not ok, please execute the update cmdlet as described in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#) of this document. Wait another 15 minutes and perform the checks described in chapter [Readiness Check for Azure Enhanced Monitoring for SAP](#) and [Health check for Azure Monitoring Infrastructure Configuration](#) again. If the checks still indicate a problem with some or all counters, please proceed to chapter [Further troubleshooting of Azure Monitoring infrastructure for SAP](#).

Further Troubleshooting of Azure Monitoring Infrastructure for SAP



Azure Performance Counters do not Show up at all

The collection of the performance metrics on Azure is done by the Windows service 'AzureEnhancedMonitoring'. If the service has not been installed correctly or if it is not running in your VM, no performance metrics can be collected at all.

Issue
The installation directory of the Azure Enhanced Monitoring extension is empty

The installation directory

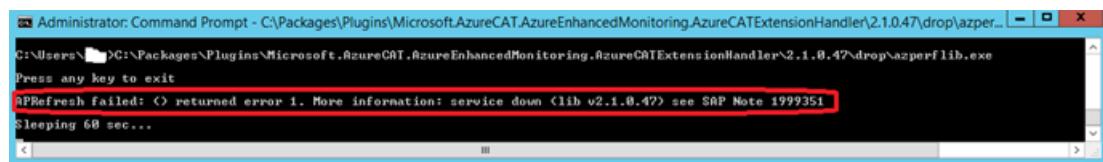
C:\Packages\Plugins\Microsoft.AzureCAT.AzureEnhancedMonitoring.AzureCATExtensionHandler\<version>\drop is empty.

Solution

The extension is not installed. Please check if it is a proxy issue (as described before). You may need to reboot the machine and/or re-run the configuration script Set-AzureRmVMAEMExtension

Issue
Service for Azure Enhanced Monitoring does not exist

Windows service 'AzureEnhancedMonitoring' does not exist. Azperflib.exe: The azperlib.exe output throws an error as shown in the [figure below](#).



Solution

If the service does not exist as shown in the [figure above](#), the Azure Monitoring Extension for SAP has not been installed correctly. Redeploy the extension according to the steps described for your deployment scenario in chapter [Deployment Scenarios of VMs for SAP on Microsoft Azure](#).

After deployment of the extension, recheck whether the Azure performance counters are provided within the Azure VM after 1 hour.

Issue
Service for Azure Enhanced Monitoring exists, but fails to start

Windows service 'AzureEnhancedMonitoring' exists and is enabled but fails to start. Check the application event log for more information.

Solution

Bad configuration. Re-enable the monitoring extension for the VM as described in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#).



Some Azure performance counters are missing

The collection of the performance metrics on Azure is done by the Windows service 'AzureEnhancedMonitoring', which gets data from several sources. Some configuration data are collected locally, performance metrics are read from Azure Diagnostics and storage counters are used from your logging on storage subscription level.

If troubleshooting using SAP Note [1999351](#) did not help, please re-run the configuration script Set-AzureRmVMAEMExtension. You may have to wait for an hour because storage analytics or diagnostics counters may not be created immediately after they are enabled. If the problem still exists, open an SAP customer support message on the component BC-OP-NT-AZR for Windows or BC-OP-LNX-AZR for a Linux virtual machine..



Azure Performance Counters do not Show up at all

The collection of the performance metrics on Azure is done by a deamon. If the deamon is not running, no performance metrics can be collected at all.

Issue
The installation directory of the Azure Enhanced Monitoring extension is empty

The directory /var/lib/waagent/ does not contain a subdirectory for the Azure Enhanced Monitoring extension.

Solution

The extension is not installed. Please check if it is a proxy issue (as described before). You may need to reboot the machine and/or re-run the configuration script Set-AzureRmVMAEMExtension



Some Azure Performance Counters are Missing

The collection of the performance metrics on Azure is done by a daemon, which gets data from several sources. Some configuration data are collected locally, performance metrics are read from Azure Diagnostics and storage counters are used from your logging on storage subscription level.

For a complete and up to date list of known issues please see SAP Note [1999351](#) containing additional troubleshooting information for the Enhanced Azure Monitoring for SAP.

If troubleshooting using SAP Note [1999351](#) did not help, please re-run the configuration script Set-AzureRmVMAEMExtension as described in chapter [Configure Azure Enhanced Monitoring Extension for SAP](#). You may have to wait for an hour because storage analytics or diagnostics counters may not be created immediately after they are enabled. If the problem still exists, open an SAP customer support message on the component BC-OP-NT-AZR for Windows or BC-OP-LNX-AZR for a Linux virtual machine.

Using SAP on Azure Virtual Machines (VMs)

1/17/2017 • 5 min to read • [Edit on GitHub](#)

By choosing Microsoft Azure as your SAP ready cloud partner, you will be able to reliably run your mission critical SAP workloads 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, Linux to Windows, SAP HANA to SQL, we have you covered.

With Microsoft Azure virtual machine services, and SAP HANA on Azure large instances, Microsoft offers a comprehensive Infrastructure As A Service (IaaS) platform. Because a wide range of SAP solutions are supported on Azure, this "getting started" document will serve as a Table of Contents for our current set of SAP documents. As more titles get added to our document library - you will see them added here.

SAP HANA Certifications on Microsoft Azure

| SAP PRODUCT | SUPPORTED OS | AZURE OFFERINGS |
|---|---|--|
| SAP HANA Developer Edition (including the HANA client software comprised of SQLODBC, ODBC-Windows only, ODBC, JDBC drivers, HANA studio, and HANA database) | Red Hat Enterprise Linux, SUSE Linux Enterprise | A7, A8 |
| MHANA One | Red Hat Enterprise Linux, SUSE Linux Enterprise | DS14_v2 (upon general availability) |
| SAP S/4HANA | Red Hat Enterprise Linux, SUSE Linux Enterprise | Controlled Availability for GS5, SAP HANA on Azure (Large instances) |
| Suite on HANA, OLTP | Red Hat Enterprise Linux, SUSE Linux Enterprise | SAP HANA on Azure (Large instances) |
| HANA Enterprise for BW, OLAP | Red Hat Enterprise Linux, SUSE Linux Enterprise | GS5 for single node deployments, SAP HANA on Azure (Large instances) |
| SAP BW/4HANA | Red Hat Enterprise Linux, SUSE Linux Enterprise | GS5 for single node deployments, SAP HANA on Azure (Large instances) |

SAP NetWeaver Certifications

Microsoft Azure is certified for the following SAP products, with full support from Microsoft and SAP.

| SAP PRODUCT | GUEST OS | RDBMS | VIRTUAL MACHINE TYPES |
|-----------------------------|--|---|---|
| SAP Business Suite Software | Windows, SUSE Linux Enterprise, Red Hat Enterprise Linux | SQL Server, Oracle (Windows only), DB2, SAP ASE | A5 to A11, D11 to D14, DS11 to DS14, GS1 to GS5 |

| SAP PRODUCT | GUEST OS | RDBMS | VIRTUAL MACHINE TYPES |
|-------------------------|--|---|---|
| SAP Business All-in-One | Windows, SUSE Linux Enterprise, Red Hat Enterprise Linux | SQL Server, Oracle (Windows only), DB2, SAP ASE | A5 to A11, D11 to D14, DS11 to DS14, GS1 to GS5 |
| SAP BusinessObjects BI | Windows | N/A | A5 to A11, D11 to D14, DS11 to DS14, GS1 to GS5 |
| SAP NetWeaver | Windows, SUSE Linux Enterprise, Red Hat Enterprise Linux | SQL Server, Oracle (Windows only), DB2, SAP ASE | A5 to A11, D11 to D14, DS11 to DS14, GS1 to GS5 |

Getting Started with SAP HANA on Azure

Title: Quickstart guide for manual installation of SAP HANA on Azure VMs

Summary: This quickstart guide will help to set up a single-instance SAP HANA prototype/demo 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: September 2016

[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). As of May 19 2016 SAP NetWeaver is officially supported on SUSE Linux VMs on Azure. All details regarding Linux versions, SAP kernel versions and so on 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](#)

Deploying SAP IDES EHP7 SP3 for SAP ERP 6.0 on Microsoft Azure

Title: Quickstart guide for manual installation of SAP HANA on Azure VMs

Summary: This article describes how to deploy SAP IDES running with SQL Server and Windows OS on Microsoft Azure via SAP Cloud Appliance Library 3.0.

Updated: September 2016

[This guide can be found here](#)

Planning and Implementation

Title: SAP NetWeaver on Azure Virtual Machines (VMs) – Planning and Implementation Guide

Summary: This is the paper to start with if you are thinking about running SAP NetWeaver in Azure Virtual Machines. This planning and implementation guide will help 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: August 2016

[This guide can be found here](#)

Deployment

Title: SAP NetWeaver on Azure Virtual Machines (VMs) – Deployment Guide

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: December 2016

[This guide can be found here](#)

DBMS Deployment Guide

Title: SAP NetWeaver on Azure Virtual Machines (VMs) – DBMS Deployment Guide

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: August 2016

[This guide can be found here](#)

High Availability Deployment Guide

Title: SAP NetWeaver on Azure Virtual Machines (VMs) – High Availability Deployment Guide

Summary: This document describes how SAP single point of failure components like the SAP ASCS/SCS and DBMS can be protected in Azure. Components of the SAP ASCS/SCS, DBMS and application servers are essential for the functionality of SAP NetWeaver systems, e.g. SAP NetWeaver ABAP, SAP NetWeaver Java, SAP NetWeaver ABAP+Java systems. Therefore, high-availability functionality needs to be put in place to make sure that those components can sustain a failure of a server or a VM as done with Windows Cluster configurations for bare-metal and Hyper-V environments.

Updated: December 2016

[This guide can be found here](#)

Create SharePoint server farms

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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.

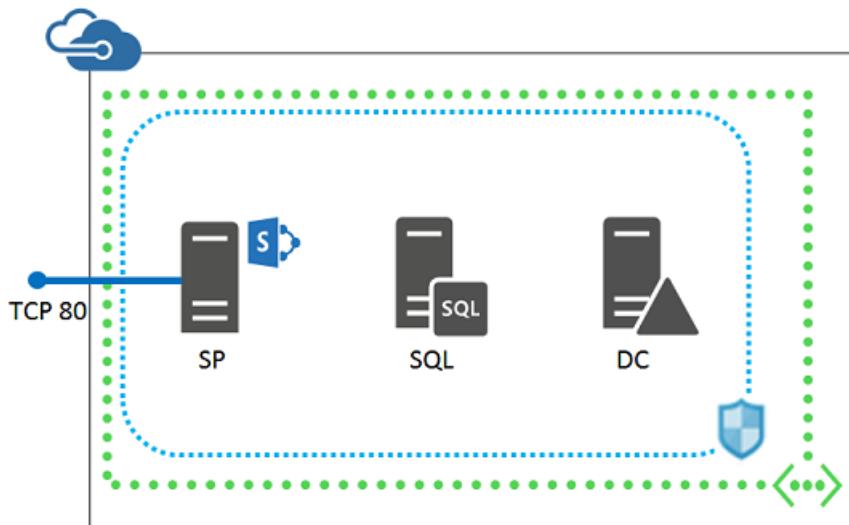
SharePoint 2013 farms

With the Microsoft Azure portal marketplace, you can quickly create pre-configured SharePoint Server 2013 farms. This can save you a lot of time when you need a basic or high-availability SharePoint farm for a dev/test environment or if you are evaluating SharePoint Server 2013 as a collaboration solution for your organization.

NOTE

The **SharePoint Server Farm** item in the Azure Marketplace of the Azure portal has been removed. It has been replaced with the **SharePoint 2013 non-HA Farm** and **SharePoint 2013 HA Farm** items.

The basic SharePoint farm consists of three virtual machines in this configuration.

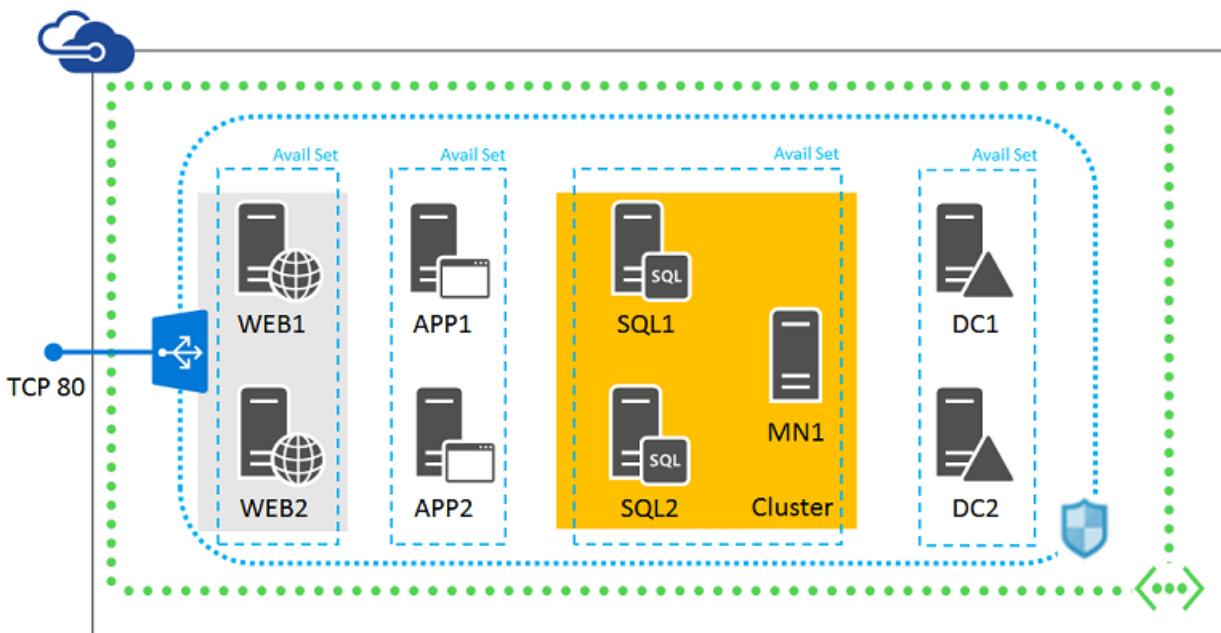


You can use this farm configuration for a simplified setup for SharePoint app development or your first-time evaluation of SharePoint 2013.

To create the basic (three-server) SharePoint farm:

1. Click [here](#).
2. Click **Deploy**.
3. On the **SharePoint 2013 non-HA Farm** pane, click **Create**.
4. Specify settings on the steps of the **Create SharePoint 2013 non-HA Farm** pane, and then click **Create**.

The high-availability SharePoint farm consists of nine virtual machines in this configuration.



You can use this farm configuration to test higher client loads, high availability of the external SharePoint site, and SQL Server AlwaysOn Availability Groups for a SharePoint farm. You can also use this configuration for SharePoint app development in a high-availability environment.

To create the high-availability (nine-server) SharePoint farm:

1. Click [here](#).
2. Click **Deploy**.
3. On the **SharePoint 2013 HA Farm** pane, click **Create**.
4. Specify settings on the seven steps of the **Create SharePoint 2013 HA Farm** pane, and then click **Create**.

NOTE

You cannot create the **SharePoint 2013 non-HA Farm** or **SharePoint 2013 HA Farm** with an Azure Free Trial.

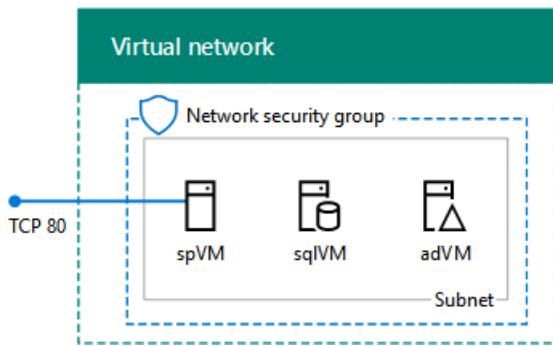
The Azure portal creates both of these farms in a cloud-only virtual network with an Internet-facing web presence. There is no site-to-site VPN or ExpressRoute connection back to your organization network.

NOTE

When you create the basic or high-availability SharePoint farms using the Azure portal, you cannot specify an existing resource group. To work around this limitation, create these farms with Azure PowerShell. For more information, see [Create SharePoint 2013 dev/test farms with Azure PowerShell](#).

SharePoint 2016 farms

See [this article](#) for the instructions to build the following single-server SharePoint Server 2016 farm.



Managing the SharePoint farms

You can administer the servers of these farms through Remote Desktop connections. For more information, see [Log on to the virtual machine](#).

From the Central Administration SharePoint site, you can configure My sites, SharePoint applications, and other functionality. For more information, see [Configure SharePoint](#).

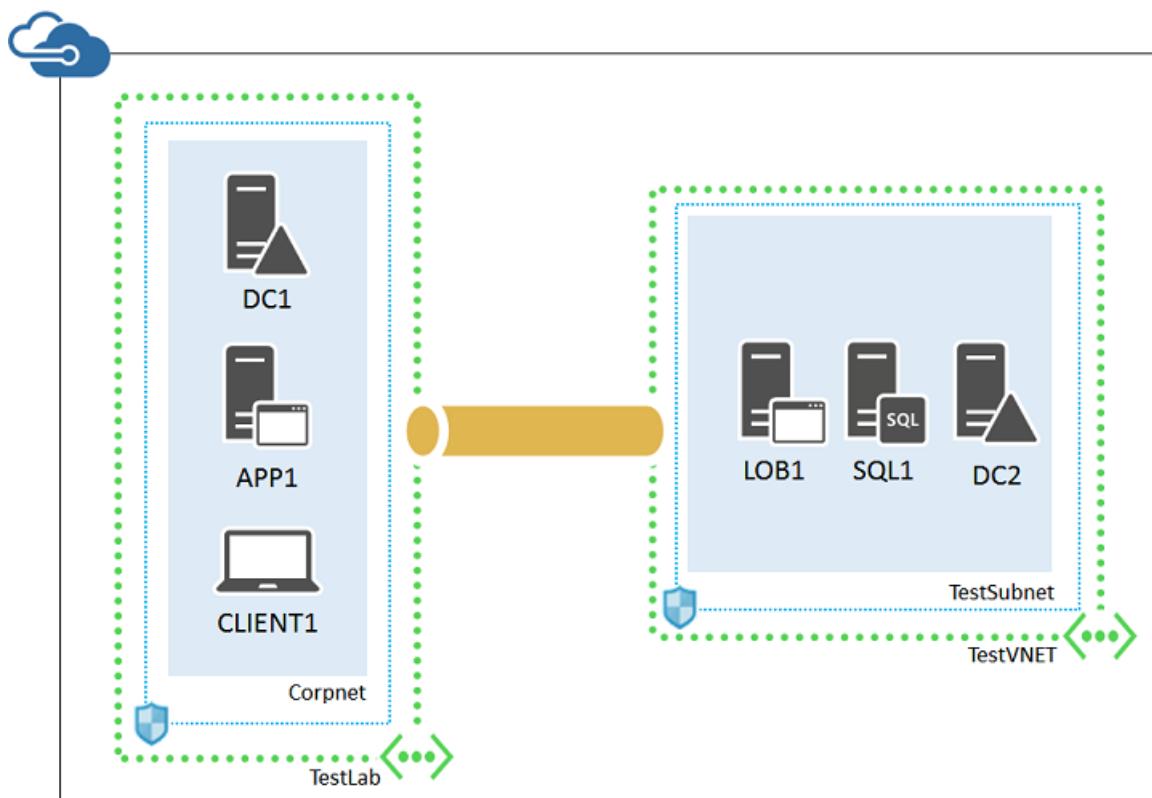
Next steps

- Discover additional [SharePoint configurations](#) in Azure infrastructure services.

Set up a web-based LOB application in a hybrid cloud for testing

1/17/2017 • 6 min to read • [Edit on GitHub](#)

This topic steps you through creating a simulated hybrid cloud environment for testing a web-based line of business (LOB) application hosted in Microsoft Azure. Here is the resulting configuration.



This configuration consists of:

- A simulated on-premises network hosted in Azure (the TestLab VNet).
- A cross-premises virtual network hosted in Azure (TestVNET).
- A VNet-to-VNet VPN connection.
- A web-based LOB server, SQL server, and secondary domain controller in the TestVNET virtual network.

This configuration provides a basis and common starting point from which you can:

- Develop and test LOB applications hosted on Internet Information Services (IIS) with a SQL Server 2014 database backend in Azure.
- Perform testing of this simulated hybrid cloud-based IT workload.

There are three major phases to setting up this hybrid cloud test environment:

1. Set up the simulated hybrid cloud environment.
2. Configure the SQL server computer (SQL1).
3. Configure the LOB server (LOB1).

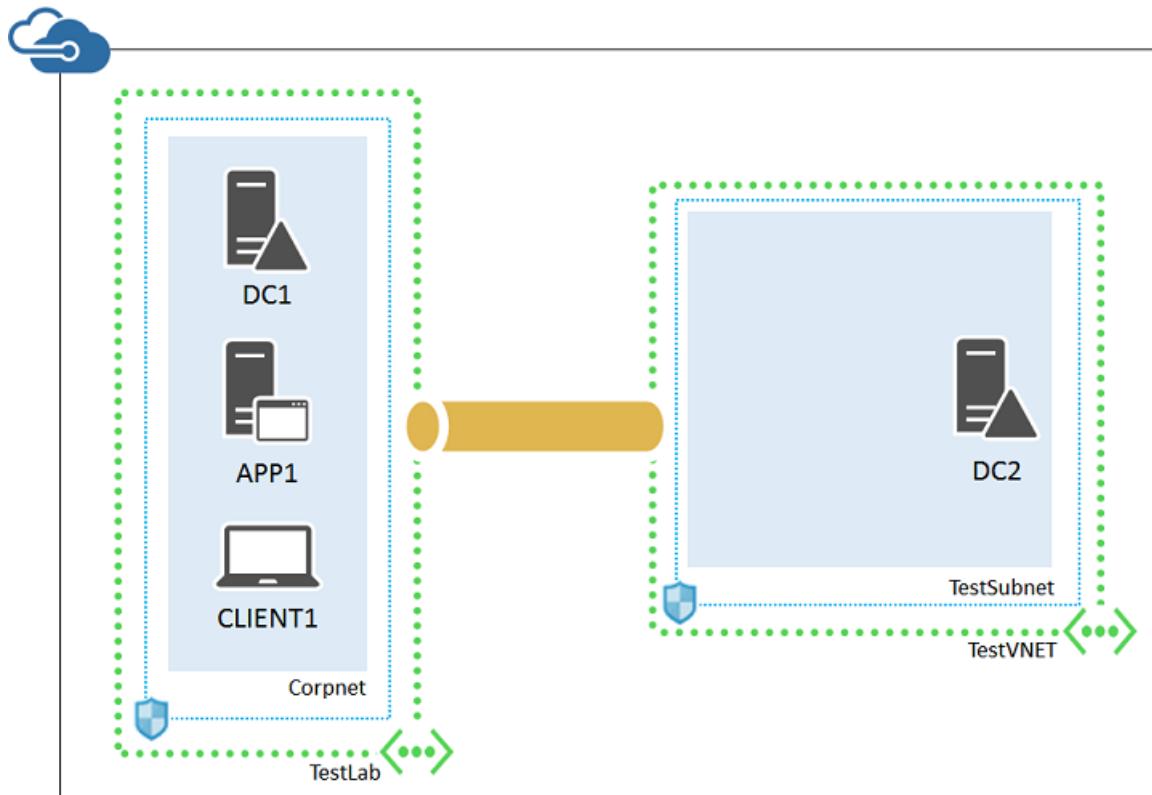
This workload requires an Azure subscription. If you have an MSDN or Visual Studio subscription, see [Monthly Azure credit for Visual Studio subscribers](#).

For an example of a production LOB application hosted in Azure, see the **Line of business applications** architecture blueprint at [Microsoft Software Architecture Diagrams and Blueprints](#).

Phase 1: Set up the simulated hybrid cloud environment

Create the [simulated hybrid cloud test environment](#). Because this test environment does not require the presence of the APP1 server on the Corpnet subnet, you can shut it down for now.

This is your current configuration.



Phase 2: Configure the SQL server computer (SQL1)

From the Azure portal, start the DC2 computer if needed.

Next, create a virtual machine for SQL1 with these commands at an Azure PowerShell command prompt on your local computer. Prior to running these commands, fill in the variable values and remove the < and > characters.

```

$rgName=<your resource group name>
$locName=<the Azure location of your resource group>
$saName=<your storage account name>

$vnet=Get-AzureRMVirtualNetwork -Name "TestVNET" -ResourceGroupName $rgName
$subnet=Get-AzureRmVirtualNetworkSubnetConfig -VirtualNetwork $vnet -Name "TestSubnet"
$PIP=New-AzureRMPublicIpAddress -Name SQL1-NIC -ResourceGroupName $rgName -Location $locName -AllocationMethod Dynamic
$nic=New-AzureRMNetworkInterface -Name SQL1-NIC -ResourceGroupName $rgName -Location $locName -Subnet $subnet - PublicIpAddress $PIP
$vm=New-AzureRMVMConfig -VMName SQL1 -VMSize Standard_A4
$storageAcc=Get-AzureRMStorageAccount -ResourceGroupName $rgName -Name $saName
$vhURI=$storageAcc.PrimaryEndpoints.Blob.ToString() + "vhds/SQL1-SQLDataDisk.vhd"
Add-AzureRMVMDataDisk -VM $vm -Name "Data" -DiskSizeInGB 100 -VhdUri $vhURI -CreateOption empty

$cred=Get-Credential -Message "Type the name and password of the local administrator account for the SQL Server computer."
$vm=Set-AzureRMVMOperatingSystem -VM $vm -Windows -ComputerName SQL1 -Credential $cred -ProvisionVMAgent - EnableAutoUpdate
$vm=Set-AzureRMVMSourceImage -VM $vm -PublisherName MicrosoftSQLServer -Offer SQL2014-WS2012R2 -Skus Standard - Version "latest"
$vm=Add-AzureRMVMNetworkInterface -VM $vm -Id $nic.Id
$storageAcc=Get-AzureRMStorageAccount -ResourceGroupName $rgName -Name $saName
$osDiskURI=$storageAcc.PrimaryEndpoints.Blob.ToString() + "vhds/SQL1-OSDisk.vhd"
$vm=Set-AzureRMVMDisk -VM $vm -Name "OSDisk" -VhdUri $osDiskURI -CreateOption fromImage
New-AzureRMVM -ResourceGroupName $rgName -Location $locName -VM $vm

```

Use the Azure portal to connect to SQL1 using the local administrator account of SQL1.

Next, configure Windows Firewall rules to allow basic connectivity testing and SQL Server traffic. From an administrator-level Windows PowerShell command prompt on SQL1, run these commands.

```

New-NetFirewallRule -DisplayName "SQL Server" -Direction Inbound -Protocol TCP -LocalPort 1433,1434,5022 -Action allow
Set-NetFirewallRule -DisplayName "File and Printer Sharing (Echo Request - ICMPv4-In)" -enabled True
ping dc2.corp.contoso.com

```

The ping command should result in four successful replies from IP address 192.168.0.4.

Next, add the extra data disk on SQL1 as a new volume with the drive letter F:.

1. In the left pane of Server Manager, click **File and Storage Services**, and then click **Disk**s.
2. In the contents pane, in the **Disk**s group, click **disk 2** (with the **Partition** set to **Unknown**).
3. Click **Tasks**, and then click **New Volume**.
4. On the Before you begin page of the New Volume Wizard, click **Next**.
5. On the Select the server and disk page, click **Disk 2**, and then click **Next**. When prompted, click **OK**.
6. On the Specify the size of the volume page, click **Next**.
7. On the Assign to a drive letter or folder page, click **Next**.
8. On the Select file system settings page, click **Next**.
9. On the Confirm selections page, click **Create**.
10. When complete, click **Close**.

Run these commands at the Windows PowerShell command prompt on SQL1:

```

md f:\Data
md f:\Log
md f:\Backup

```

Next, join SQL1 to the CORP Windows Active Directory domain with these commands at the Windows

PowerShell prompt on SQL1.

```
Add-Computer -DomainName corp.contoso.com  
Restart-Computer
```

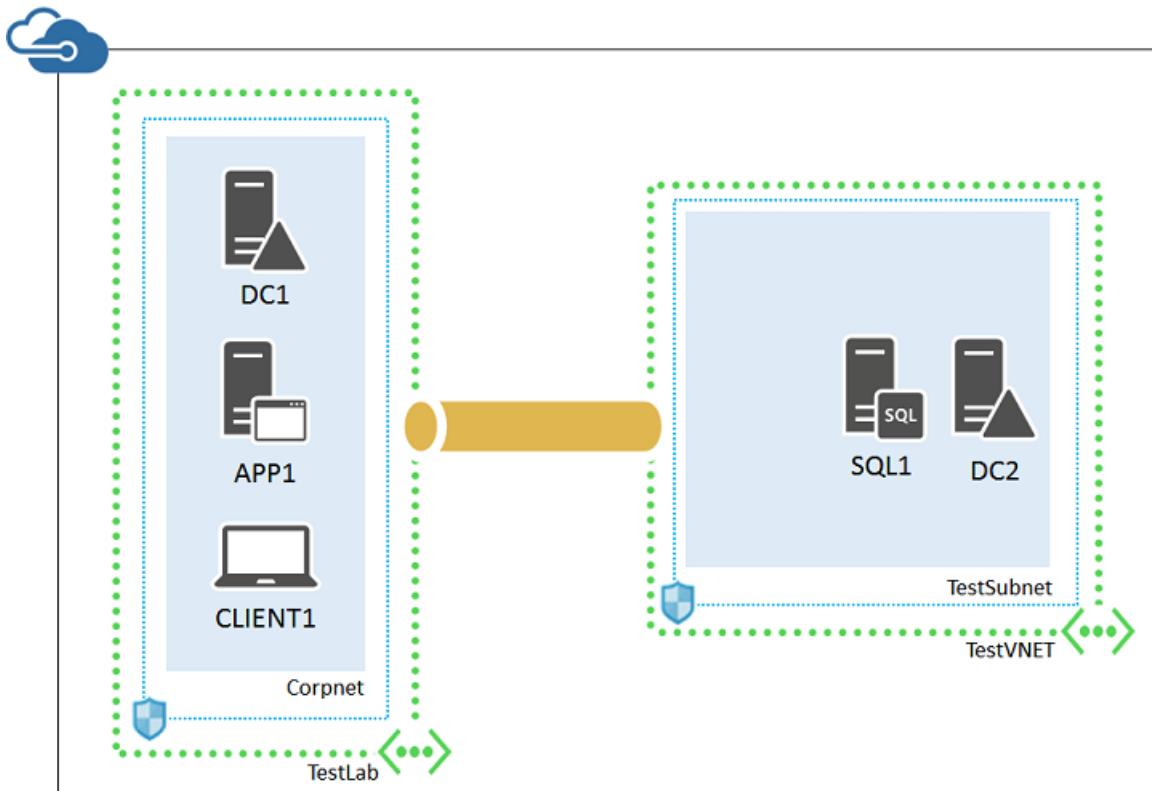
Use the CORP\User1 account when prompted to supply domain account credentials for the **Add-Computer** command.

After restarting, use the Azure portal to connect to SQL1 *with the local administrator account of SQL1*.

Next, configure SQL Server 2014 to use the F: drive for new databases and for user account permissions.

1. From the Start screen, type **SQL Server Management**, and then click **SQL Server 2014 Management Studio**.
2. In **Connect to Server**, click **Connect**.
3. In the Object Explorer tree pane, right-click **SQL1**, and then click **Properties**.
4. In the **Server Properties** window, click **Database Settings**.
5. Locate the **Database default locations** and set these values:
 - For **Data**, type the path **f:\Data**.
 - For **Log**, type the path **f:\Log**.
 - For **Backup**, type the path **f:\Backup**.
 - Note: Only new databases use these locations.
6. Click the **OK** to close the window.
7. In the **Object Explorer** tree pane, open **Security**.
8. Right-click **Logins** and then click **New Login**.
9. In **Login name**, type **CORP\User1**.
10. On the **Server Roles** page, click **sysadmin**, and then click **OK**.
11. Close Microsoft SQL Server Management Studio.

This is your current configuration.



Phase 3: Configure the LOB server (LOB1)

First, create a virtual machine for LOB1 with these commands at the Azure PowerShell command prompt on your local computer.

```
$rgName=<your resource group name>
$locName=<your Azure location, such as West US>
$saName=<your storage account name>

$vnet=Get-AzureRMVirtualNetwork -Name "TestVNET" -ResourceGroupName $rgName
$subnet=Get-AzureRMVirtualNetworkSubnetConfig -VirtualNetwork $vnet -Name "TestSubnet"
$PIP=New-AzureRMPublicIpAddress -Name LOB1-NIC -ResourceGroupName $rgName -Location $locName -AllocationMethod Dynamic
$nic=New-AzureRMNetworkInterface -Name LOB1-NIC -ResourceGroupName $rgName -Location $locName -Subnet $subnet - PublicIpAddress $PIP
$vm=New-AzureRMVMConfig -VMName LOB1 -VMSize Standard_A2
$storageAcc=Get-AzureRMStorageAccount -ResourceGroupName $rgName -Name $saName
$cred=Get-Credential -Message "Type the name and password of the local administrator account for LOB1."
$vm=Set-AzureRMVMOperatingSystem -VM $vm -Windows -ComputerName LOB1 -Credential $cred -ProvisionVMAgent - EnableAutoUpdate
$vm=Set-AzureRMVMSourceImage -VM $vm -PublisherName MicrosoftWindowsServer -Offer WindowsServer -Skus 2012-R2- Datacenter -Version "latest"
$vm=Add-AzureRMVMNetworkInterface -VM $vm -Id $nic.Id
$osDiskUri=$storageAcc.PrimaryEndpoints.Blob.ToString() + "vhds/LOB1-TestLab-OSDisk.vhd"
$vm=Set-AzureRMVMOSDisk -VM $vm -Name LOB1-TestVNET-OSDisk -VhdUri $osDiskUri -CreateOption fromImage
New-AzureRMVM -ResourceGroupName $rgName -Location $locName -VM $vm
```

Next, use the Azure portal to connect to LOB1 with the credentials of the local administrator account of LOB1.

Next, configure a Windows Firewall rule to allow traffic for basic connectivity testing. From an administrator-level Windows PowerShell command prompt on LOB1, run these commands.

```
Set-NetFirewallRule -DisplayName "File and Printer Sharing (Echo Request - ICMPv4-In)" -enabled True
ping dc2.corp.contoso.com
```

The ping command should result in four successful replies from IP address 192.168.0.4.

Next, join LOB1 to the CORP Active Directory domain with these commands at the Windows PowerShell prompt.

```
Add-Computer -DomainName corp.contoso.com
Restart-Computer
```

Use the CORP\User1 account when prompted to supply domain account credentials for the **Add-Computer** command.

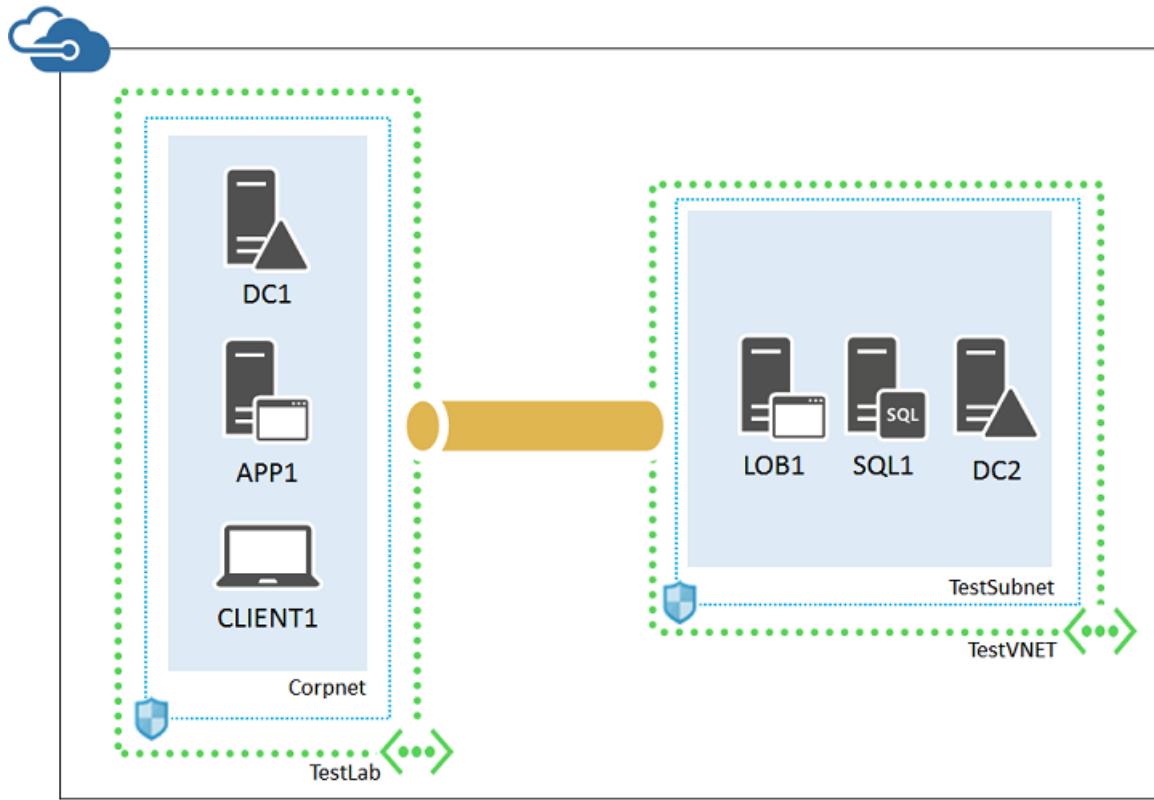
After restarting, use the Azure portal to connect to LOB1 with the CORP\User1 account and password.

Next, configure LOB1 for IIS and test access from CLIENT1.

1. From Server Manager, click **Add roles and features**.
2. On the **Before you begin** page, click **Next**.
3. On the **Select installation type** page, click **Next**.
4. On the **Select destination server** page, click **Next**.
5. On the **Server roles** page, click **Web Server (IIS)** in the list of **Roles**.
6. When prompted, click **Add Features**, and then click **Next**.
7. On the **Select features** page, click **Next**.
8. On the **Web Server (IIS)** page, click **Next**.
9. On the **Select role services** page, select or clear the check boxes for the services you need for testing your LOB application, and then click **Next**.
10. On the **Confirm installation selections** page, click **Install**.

11. Wait until the installation of components has completed and then click **Close**.
12. From the Azure portal, connect to the CLIENT1 computer with the CORP\User1 account credentials, and then start Internet Explorer.
13. In the Address bar, type **http://lob1** and then press ENTER. You should see the default IIS 8 web page.

This is your current configuration.



This environment is now ready for you to deploy your web-based application on LOB1 and test functionality from CLIENT1 on the Corpnet subnet.

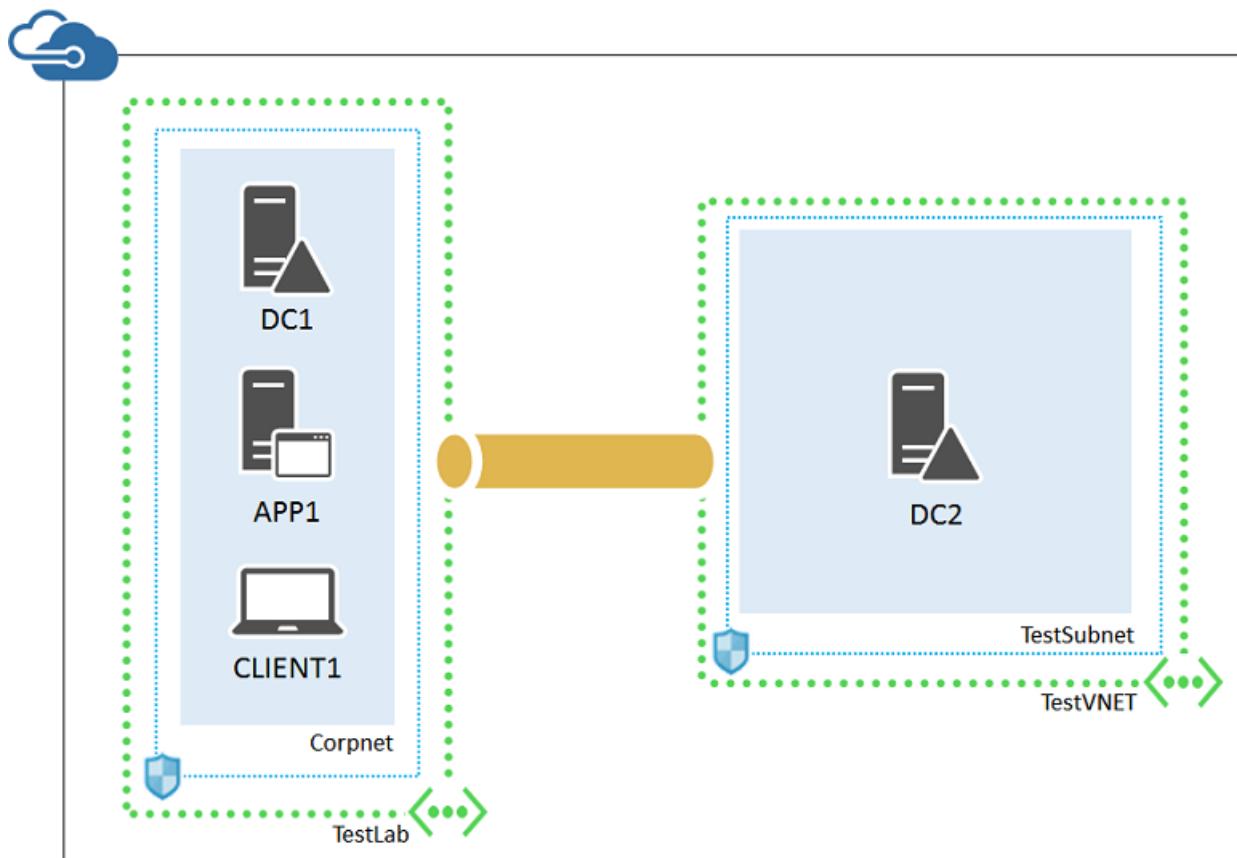
Next step

- Add a new virtual machine using the [Azure portal](#).

Set up a simulated hybrid cloud environment for testing

1/17/2017 • 6 min to read • [Edit on GitHub](#)

This article steps you through creating a simulated hybrid cloud environment with Microsoft Azure using two Azure virtual networks. Here is the resulting configuration.



This simulates a hybrid cloud production environment and consists of:

- A simulated and simplified on-premises network hosted in an Azure virtual network (the TestLab virtual network).
- A simulated cross-premises virtual network hosted in Azure (TestVNET).
- A VNet-to-VNet connection between the two virtual networks.
- A secondary domain controller in the TestVNET virtual network.

This provides a basis and common starting point from which you can:

- Develop and test applications in a simulated hybrid cloud environment.
- Create test configurations of computers, some within the TestLab virtual network and some within the TestVNET virtual network, to simulate hybrid cloud-based IT workloads.

There are four major phases to setting up this hybrid cloud test environment:

1. Configure the TestLab virtual network.
2. Create the cross-premises virtual network.
3. Create the VNet-to-VNet VPN connection.
4. Configure DC2.

This configuration requires an Azure subscription. If you have an MSDN or Visual Studio subscription, see [Monthly Azure credit for Visual Studio subscribers](#).

NOTE

Virtual machines and virtual network gateways in Azure incur an ongoing monetary cost when they are running. This cost is billed against your MSDN or paid subscription. An Azure VPN gateway is implemented as a set of two Azure virtual machines. To minimize the costs, create the test environment and perform your needed testing and demonstration as quickly as possible.

Phase 1: Configure the TestLab virtual network

Use the instructions in the [Base Configuration test environment](#) topic to configure the DC1, APP1, and CLIENT1 computers in the Azure virtual network named TestLab.

Next, start an Azure PowerShell prompt.

NOTE

The following command sets use Azure PowerShell 1.0 and later.

Sign in to your account.

```
Login-AzureRMAccount
```

Get your subscription name using the following command.

```
Get-AzureRMSubscription | Sort SubscriptionName | Select SubscriptionName
```

Set your Azure subscription. Use the same subscription that you used to build your base configuration in Phase 1. Replace everything within the quotes, including the < and > characters, with the correct name.

```
$subscr=<subscription name>
Get-AzureRmSubscription -SubscriptionName $subscr | Select-AzureRmSubscription
```

Next, add a gateway subnet to the TestLab virtual network of your base configuration, which will be used to host the Azure gateway.

```
$rgName=<name of your resource group that you used for your TestLab virtual network>
$locName=<Azure location name where you placed the TestLab virtual network, such as West US>
$vnet=Get-AzureRmVirtualNetwork -ResourceGroupName $rgName -Name TestLab
Add-AzureRmVirtualNetworkSubnetConfig -Name "GatewaySubnet" -AddressPrefix 10.255.255.248/29 -VirtualNetwork
$vnet
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet
```

Next, request a public IP address to assign to the gateway for the TestLab virtual network.

```
$gwpip=New-AzureRmPublicIpAddress -Name TestLab_pip -ResourceGroupName $rgName -Location $locName -
AllocationMethod Dynamic
```

Next, create your gateway.

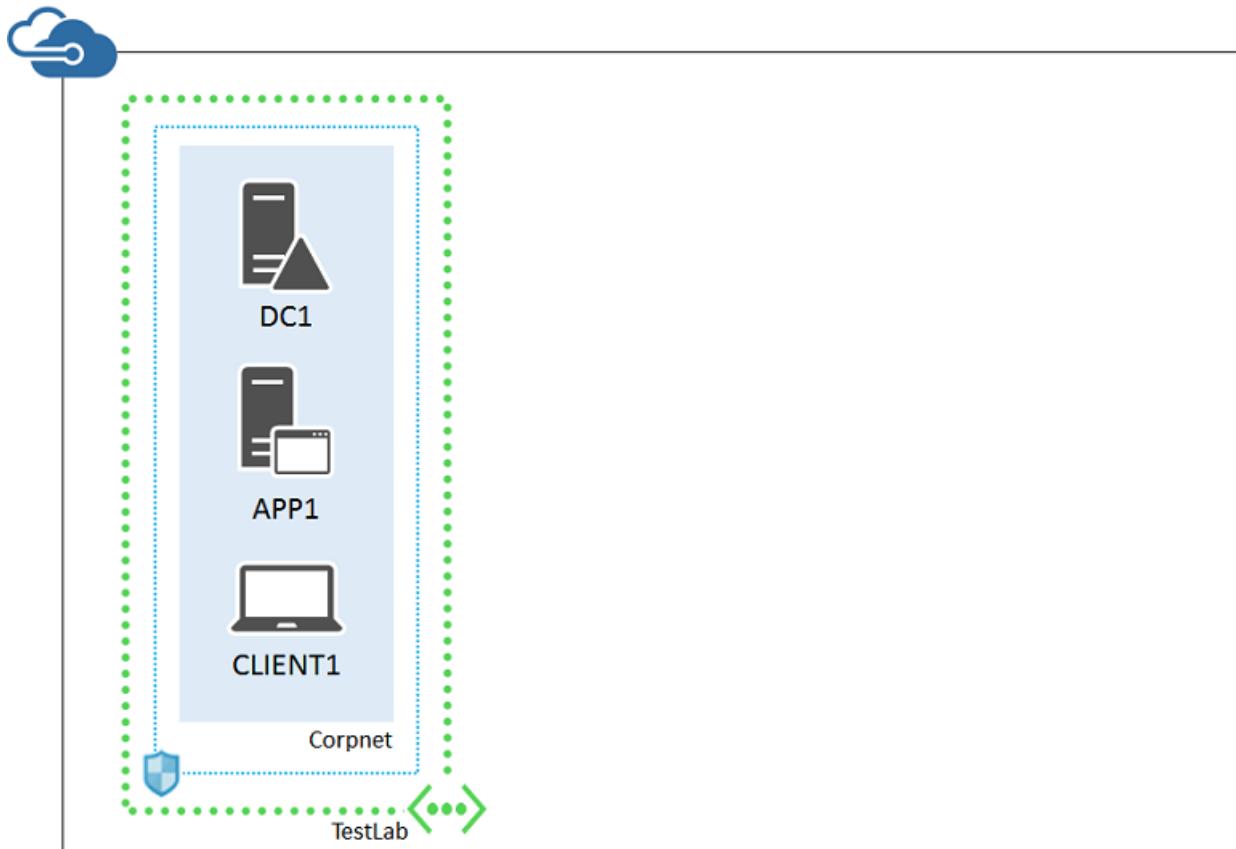
```
$vnet=Get-AzureRmVirtualNetwork -Name TestLab -ResourceGroupName $rgName
$subnet=Get-AzureRmVirtualNetworkSubnetConfig -Name "GatewaySubnet" -VirtualNetwork $vnet
$gwipconfig=New-AzureRmVirtualNetworkGatewayIpConfig -Name TestLab_GWConfig -SubnetId $subnet.Id -
PublicIpAddressId $gwip.Id
New-AzureRmVirtualNetworkGateway -Name TestLab_GW -ResourceGroupName $rgName -Location $locName -
IpConfigurations $gwipconfig -GatewayType Vpn -VpnType RouteBased
```

Keep in mind that new gateways can take 20 minutes or more to create.

From the Azure portal on your local computer, connect to DC1 with the CORP\User1 credentials. To configure the CORP domain so that computers and users use their local domain controller for authentication, run these commands from an administrator-level Windows PowerShell command prompt on DC1.

```
New-ADReplicationSite -Name "TestLab"
New-ADReplicationSite -Name "TestVNET"
New-ADReplicationSubnet -Name "10.0.0.0/8" -Site "TestLab"
New-ADReplicationSubnet -Name "192.168.0.0/16" -Site "TestVNET"
```

This is your current configuration.



Phase 2: Create the TestVNET virtual network

First, create the TestVNET virtual network and protect it with a network security group.

```

$rgName=<name of the resource group that you used for your TestLab virtual network>
$locName=<Azure location name where you placed the TestLab virtual network, such as West US>
$locShortName=<Azure location name from $locName in all lowercase letters with spaces removed. Example:
westus>
$testSubnet=New-AzureRMVirtualNetworkSubnetConfig -Name "TestSubnet" -AddressPrefix 192.168.0.0/24
$gatewaySubnet=New-AzureRMVirtualNetworkSubnetConfig -Name "GatewaySubnet" -AddressPrefix 192.168.255.248/29
New-AzureRMVirtualNetwork -Name "TestVNET" -ResourceGroupName $rgName -Location $locName -AddressPrefix
192.168.0.0/16 -Subnet $testSubnet,$gatewaySubnet -DNSServer 10.0.0.4
$rule1=New-AzurermNetworkSecurityRuleConfig -Name "RDPTraffic" -Description "Allow RDP to all VMs on the subnet"
-Access Allow -Protocol Tcp -Direction Inbound -Priority 100 -SourceAddressPrefix Internet -SourcePortRange *
-DestinationAddressPrefix * -DestinationPortRange 3389
New-AzurermNetworkSecurityGroup -Name "TestSubnet" -ResourceGroupName $rgName -Location $locShortName -
SecurityRules $rule1
$vnet=Get-AzureRMVirtualNetwork -ResourceGroupName $rgName -Name TestVNET
$nsg=Get-AzurermNetworkSecurityGroup -Name "TestSubnet" -ResourceGroupName $rgName
Set-AzureRMVirtualNetworkSubnetConfig -VirtualNetwork $vnet -Name "TestSubnet" -AddressPrefix 192.168.0.0/24 -
NetworkSecurityGroup $nsg

```

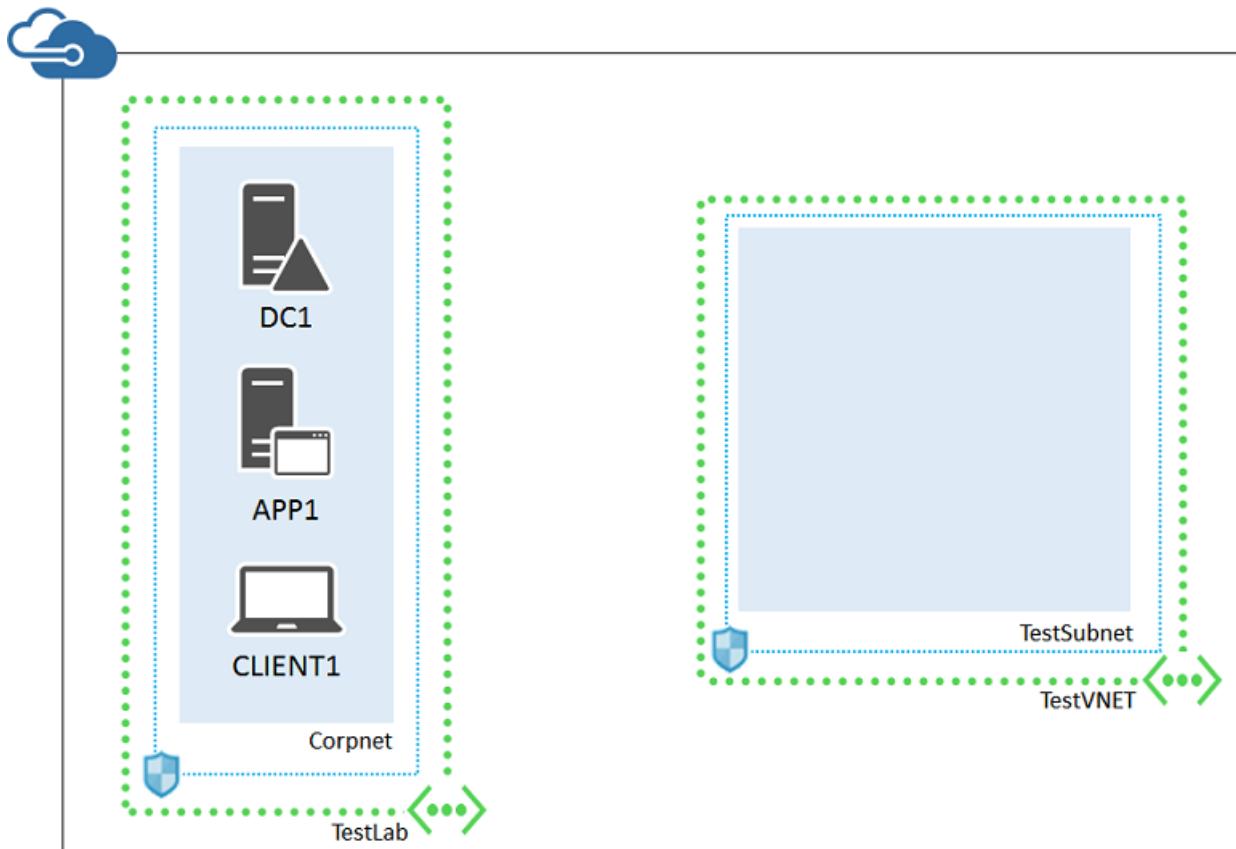
Next, request a public IP address to be allocated to the gateway for the TestVNET virtual network and create your gateway.

```

$gwpip=New-AzureRmPublicIpAddress -Name TestVNET_pip -ResourceGroupName $rgName -Location $locName -
AllocationMethod Dynamic
$vnet=Get-AzureRmVirtualNetwork -Name TestVNET -ResourceGroupName $rgName
$subnet=Get-AzureRmVirtualNetworkSubnetConfig -Name "GatewaySubnet" -VirtualNetwork $vnet
$gwipconfig=New-AzureRmVirtualNetworkGatewayIpConfig -Name "TestVNET_GWConfig" -SubnetId $subnet.Id -
PublicIpAddressId $gwpip.Id
New-AzureRmVirtualNetworkGateway -Name "TestVNET_GW" -ResourceGroupName $rgName -Location $locName -
IpConfigurations $gwipconfig -GatewayType Vpn -VpnType RouteBased

```

This is your current configuration.



Phase 3: Create the VNet-to-VNet connection

First, obtain a random, cryptographically strong, 32-character pre-shared key from your network or security

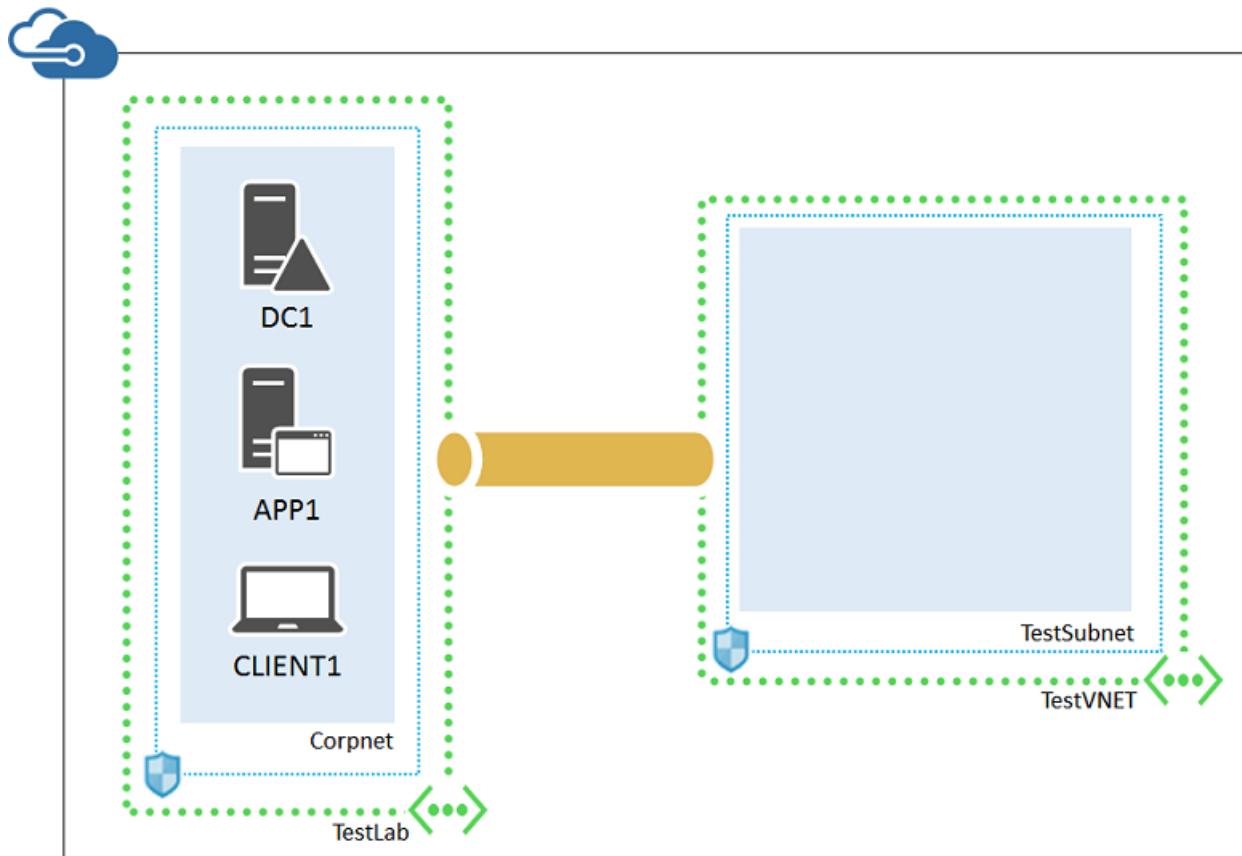
administrator. Alternately, use the information at [Create a random string for an IPsec preshared key](#) to obtain a pre-shared key.

Next, use these commands to create the VNet-to-VNet VPN connection, which can take some time to complete.

```
$sharedKey=<pre-shared key value>
$gwTestLab=Get-AzureRmVirtualNetworkGateway -Name TestLab_GW -ResourceGroupName $rgName
$gwTestVNET=Get-AzureRmVirtualNetworkGateway -Name TestVNET_GW -ResourceGroupName $rgName
New-AzureRmVirtualNetworkGatewayConnection -Name TestLab_to_TestVNET -ResourceGroupName $rgName -
VirtualNetworkGateway1 $gwTestLab -VirtualNetworkGateway2 $gwTestVNET -Location $locName -ConnectionType
Vnet2Vnet -SharedKey $sharedKey
New-AzureRmVirtualNetworkGatewayConnection -Name TestVNET_to_TestLab -ResourceGroupName $rgName -
VirtualNetworkGateway1 $gwTestVNET -VirtualNetworkGateway2 $gwTestLab -Location $locName -ConnectionType
Vnet2Vnet -SharedKey $sharedKey
```

After a few minutes, the connection should be established.

This is your current configuration.



Phase 4: Configure DC2

First, create a virtual machine for DC2. Run these commands at the Azure PowerShell command prompt on your local computer.

```

$rgName=<your resource group name>
$locName=<your Azure location, such as West US>
$saName=<the storage account name for the base configuration>
$vnet=Get-AzureRMVirtualNetwork -Name TestVNET -ResourceGroupName $rgName
$pip>New-AzureRMPublicIpAddress -Name DC2-NIC -ResourceGroupName $rgName -Location $locName -AllocationMethod Dynamic
$nic=New-AzureRMNetworkInterface -Name DC2-NIC -ResourceGroupName $rgName -Location $locName -SubnetId $vnet.Subnets[0].Id -PublicIpAddressId $pip.Id -PrivateIpAddress 192.168.0.4
$vm=New-AzureRMVMConfig -VMName DC2 -VMSize Standard_A1
$storageAcc=Get-AzureRMStorageAccount -ResourceGroupName $rgName -Name $saName
$vhdURI=$storageAcc.PrimaryEndpoints.Blob.ToString() + "vhds/DC2-TestVNET-ADSDisk.vhd"
Add-AzureRMVMDataDisk -VM $vm -Name ADDS-Data -DiskSizeInGB 20 -VhdUri $vhdURI -CreateOption empty
$cred=Get-Credential -Message "Type the name and password of the local administrator account for DC2."
$vm=Set-AzureRMVMOperatingSystem -VM $vm -Windows -ComputerName DC2 -Credential $cred -ProvisionVMAgent -EnableAutoUpdate
$vm=Set-AzureRMVMSourceImage -VM $vm -PublisherName MicrosoftWindowsServer -Offer WindowsServer -Skus 2012-R2-Datacenter -Version "latest"
$vm=Add-AzureRMVMNetworkInterface -VM $vm -Id $nic.Id
$osDiskURI=$storageAcc.PrimaryEndpoints.Blob.ToString() + "vhds/DC2-TestLab-OSDisk.vhd"
$vm=Set-AzureRMVMOSDisk -VM $vm -Name DC2-TestVNET-OSDisk -VhdUri $osDiskURI -CreateOption fromImage
New-AzureRMVM -ResourceGroupName $rgName -Location $locName -VM $vm

```

Next, connect to the new DC2 virtual machine from the Azure portal.

Next, configure a Windows Firewall rule to allow traffic for basic connectivity testing. From an administrator-level Windows PowerShell command prompt on DC2, run these commands.

```

Set-NetFirewallRule -DisplayName "File and Printer Sharing (Echo Request - ICMPv4-In)" -enabled True
ping dc1.corp.contoso.com

```

The ping command should result in four successful replies from IP address 10.0.0.4. This is a test of traffic across the VNet-to-VNet connection.

Next, add the extra data disk on DC2 as a new volume with the drive letter F:

1. In the left pane of Server Manager, click **File and Storage Services**, and then click **Disks**.
2. In the contents pane, in the **Disks** group, click **disk 2** (with the **Partition** set to **Unknown**).
3. Click **Tasks**, and then click **New Volume**.
4. On the Before you begin page of the New Volume Wizard, click **Next**.
5. On the Select the server and disk page, click **Disk 2**, and then click **Next**. When prompted, click **OK**.
6. On the Specify the size of the volume page, click **Next**.
7. On the Assign to a drive letter or folder page, click **Next**.
8. On the Select file system settings page, click **Next**.
9. On the Confirm selections page, click **Create**.
10. When complete, click **Close**.

Next, configure DC2 as a replica domain controller for the corp.contoso.com domain. Run these commands from the Windows PowerShell command prompt on DC2.

```

Install-WindowsFeature AD-Domain-Services -IncludeManagementTools
Install-ADDSDomainController -Credential (Get-Credential CORP\User1) -DomainName "corp.contoso.com" -
InstallDns:$true -DatabasePath "F:\NTDS" -LogPath "F:\Logs" -SysvolPath "F:\SYSVOL"

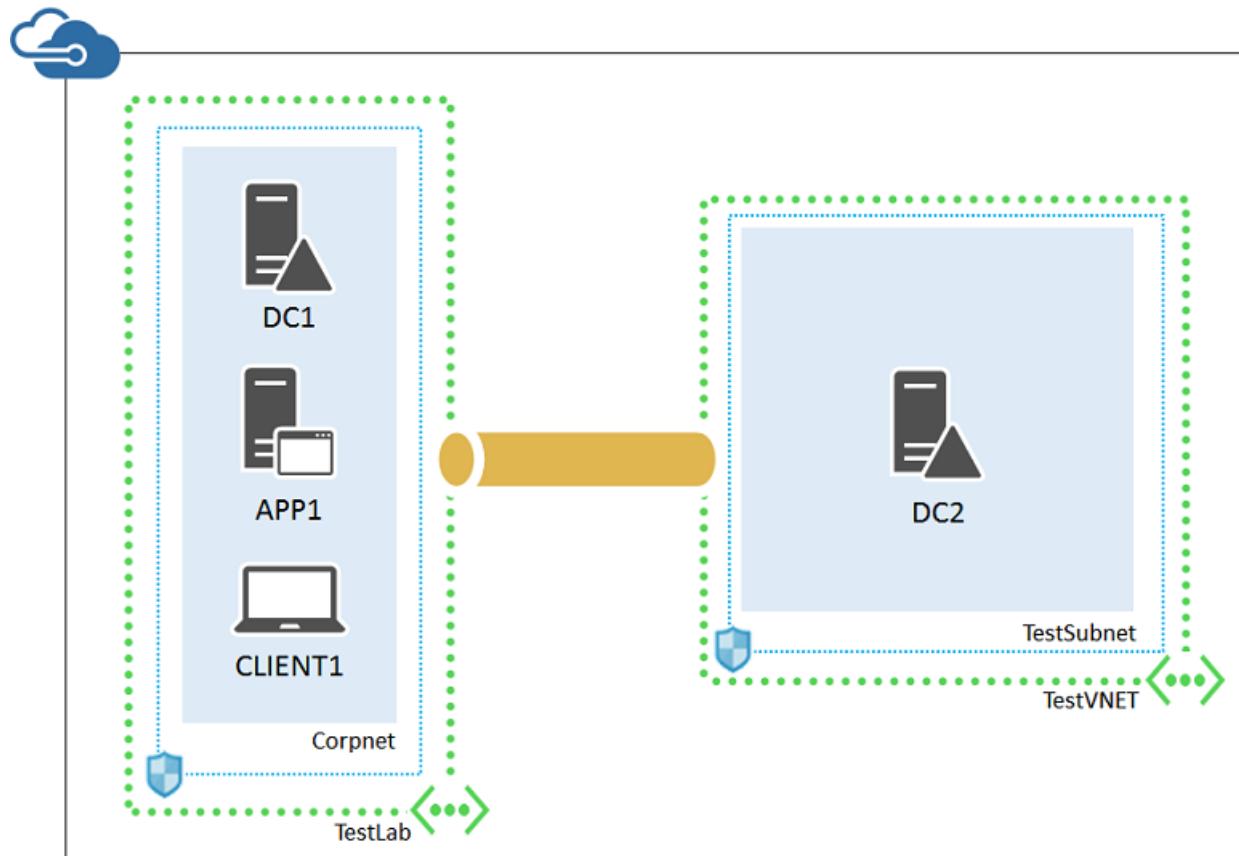
```

Note that you are prompted to supply both the CORP\User1 password and a Directory Services Restore Mode (DSRM) password, and to restart DC2.

Now that the TestVNET virtual network has its own DNS server (DC2), you must configure the TestVNET virtual network to use this DNS server.

1. In the left pane of the Azure portal, click the virtual networks icon, and then click **TestVNET**.
2. On the **Settings** tab, click **DNS servers**.
3. In **Primary DNS server**, type **192.168.0.4** to replace 10.0.0.4.
4. Click **Save**.

This is your current configuration.



Your simulated hybrid cloud environment is now ready for testing.

Next step

- Set up a [web-based, line of business application](#) in this environment.

Troubleshoot Remote Desktop connections to an Azure virtual machine

1/17/2017 • 10 min to read • [Edit on GitHub](#)

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. Check the VM Resource Health.
5. Reset your VM password.
6. Restart your VM.
7. Redeploy your VM.

Continue reading if you need more detailed steps and explanations.

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](#).

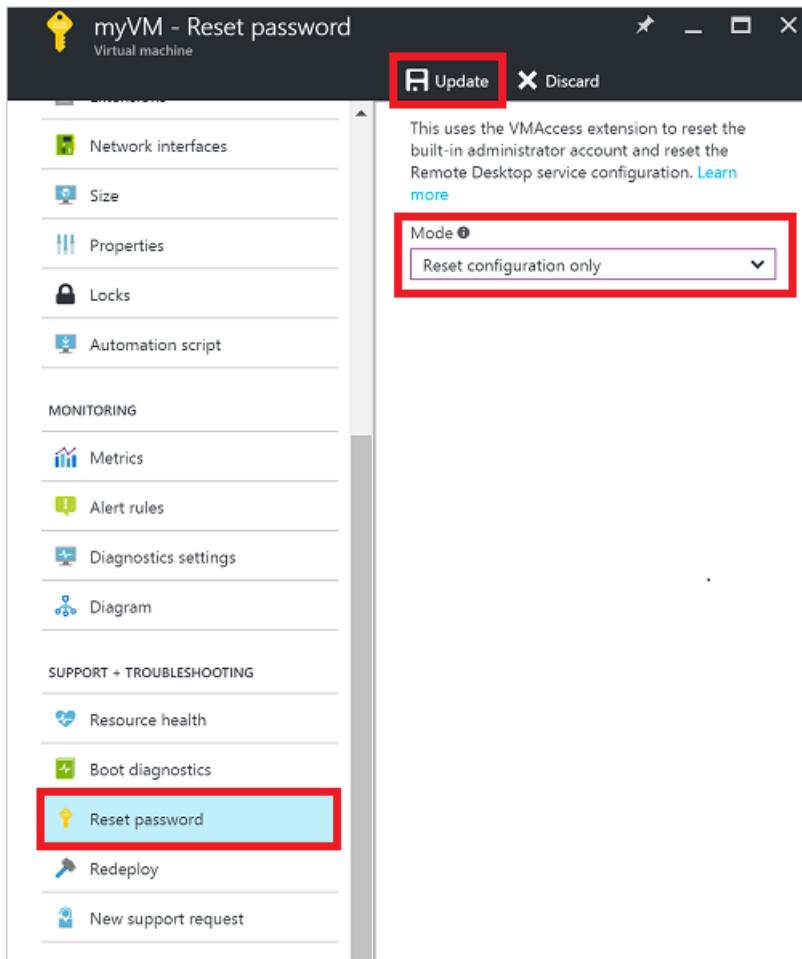
Troubleshoot using the Azure portal

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.** 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.

Select your VM in the Azure portal. Click the **Network interfaces** from the settings pane.

The screenshot shows the Azure portal interface for a virtual machine named 'myVM'. The left sidebar has a 'Network interfaces' option highlighted with a red box. The main content area displays the 'Essentials' section with various configuration details:

- Resource group: myResourceGroup
- Status: Running
- Location: West US
- Subscription name: (not specified)
- Subscription ID: (not specified)
- Computer name: myVM
- Operating system: Windows
- Size: Standard D1 v2 (1 core, 3.5 GB memory)
- Public IP address/DNS name label: 104.42.99.6/<none>
- Virtual network/subnet: myVnet/mySubnet

The 'Monitoring' section shows CPU percentage levels from 40% to 100%.

Select your network interface from the list (there is typically only one):

The screenshot shows the 'Network interfaces' page for the 'myVM' virtual machine. The table lists a single network interface:

| NAME | PUBLIC IP ADDRE... | PRIVATE IP ADDR... | SECURITY GROUP |
|---------|--------------------|--------------------|--------------------------|
| myvm992 | 104.42.99.6 | 10.1.0.4 | myNetworkSecurity... ... |

Select **Network security group** to view the Network Security Group associated with your network interface:

The screenshot shows the details for the 'myvm992' network interface. The 'Overview' tab is selected in the left sidebar. The 'Essentials' section includes the following information:

- Resource group: myResourceGroup
- Location: West US
- Subscription name: (not specified)
- Subscription ID: (not specified)
- Private IP address: 10.1.0.4
- Virtual network/subnet: myVnet/mySubnet
- Public IP address: 104.42.99.6 (myPublicIP)
- Network security group: myNetworkSecurityGroup

A note at the bottom states: 'Attached to myVM'.

Verify that an inbound rule exists that allows RDP traffic on TCP port 3389. The following example shows a valid security rule that permits RDP traffic. You can see **Service** and **Action** are configured correctly:

myNetworkSecurityGroup
Network security group

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

SETTINGS

Inbound security rules

Outbound security rules

Essentials

Resource group **myResourceGroup**
Location West US
Subscription name
Subscription ID

Security rules
1 inbound, 0 outbound
Associated with 0 subnets, 1 network interfaces

1 Inbound security rule

| PRIORITY | NAME | SOURCE | DESTINATION | SERVICE | ACTION |
|----------|-------------------|--------|-------------|----------------|--------|
| 1000 | default-allow-rdp | Any | Any | RDP (TCP/3389) | Allow |

If you do not have a rule that allows RDP traffic, [create a Network Security Group rule](#). Allow TCP 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

Available Last updated: 10/14/2016, 3:07:00 PM

Resource health watches your resource and tells you if it's running as expected. [Learn more](#)

There aren't any known Azure platform problems affecting this virtual machine.

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

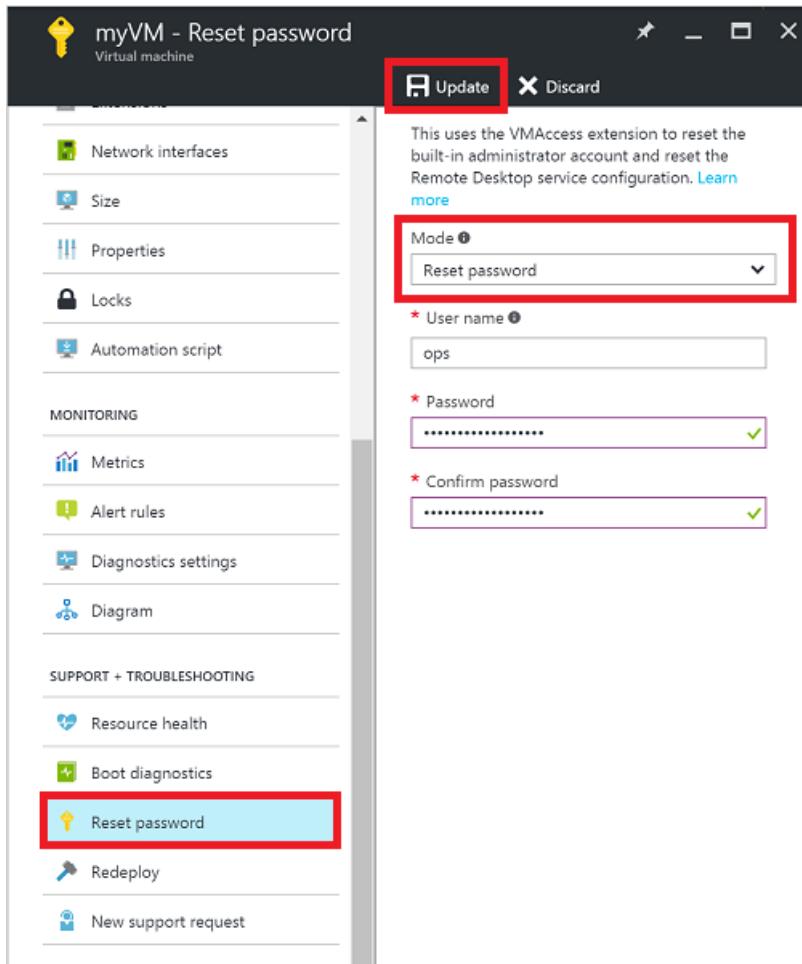
Resource health

Boot diagnostics

5. **Reset user credentials.** This troubleshooting step resets the password on a local administrator account when you are unsure or have forgotten the credentials.

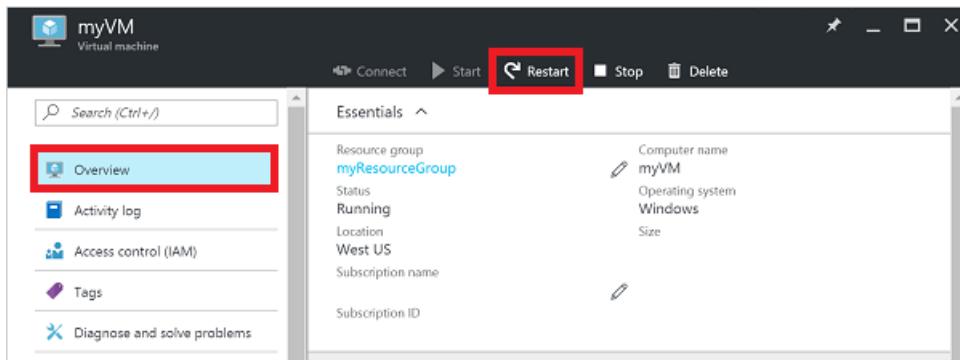
Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting**

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:



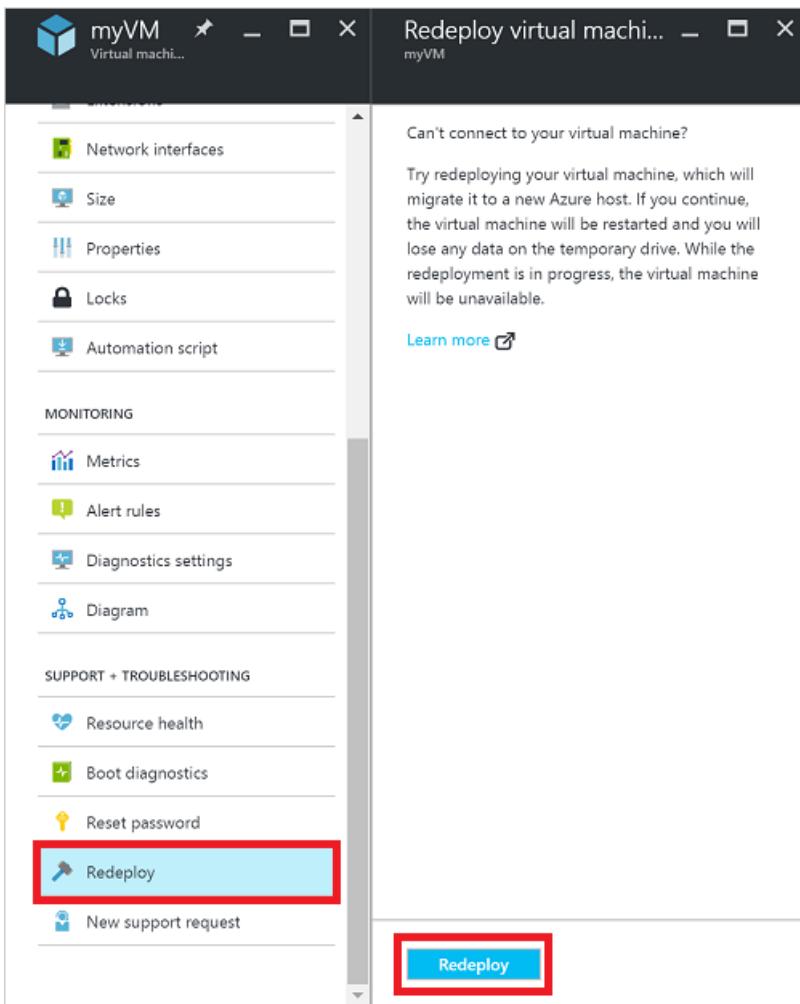
6. **Restart your VM.** This troubleshooting step can correct any underlying issues the VM itself is having.

Select your VM in the Azure portal and click the **Overview** tab. Click the **Restart** button:



7. **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 follow 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/my  
NetworkSecurityGroup/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 3389.

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"
```

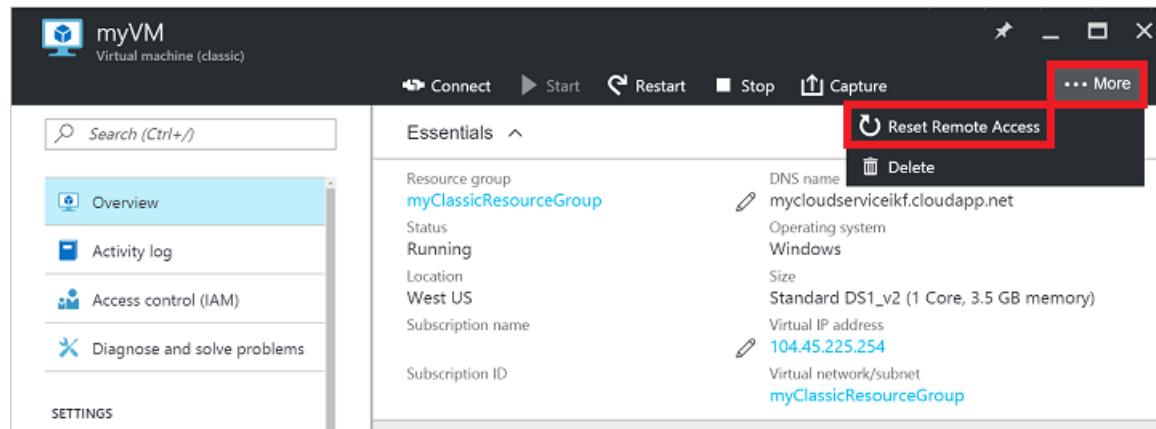
If you are still encountering RDP issues, you can [open a support request](#) or read [more detailed RDP troubleshooting concepts and steps](#).

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 |
|---------|----------|-------------|---------------|-----------|
| 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)

Refresh

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

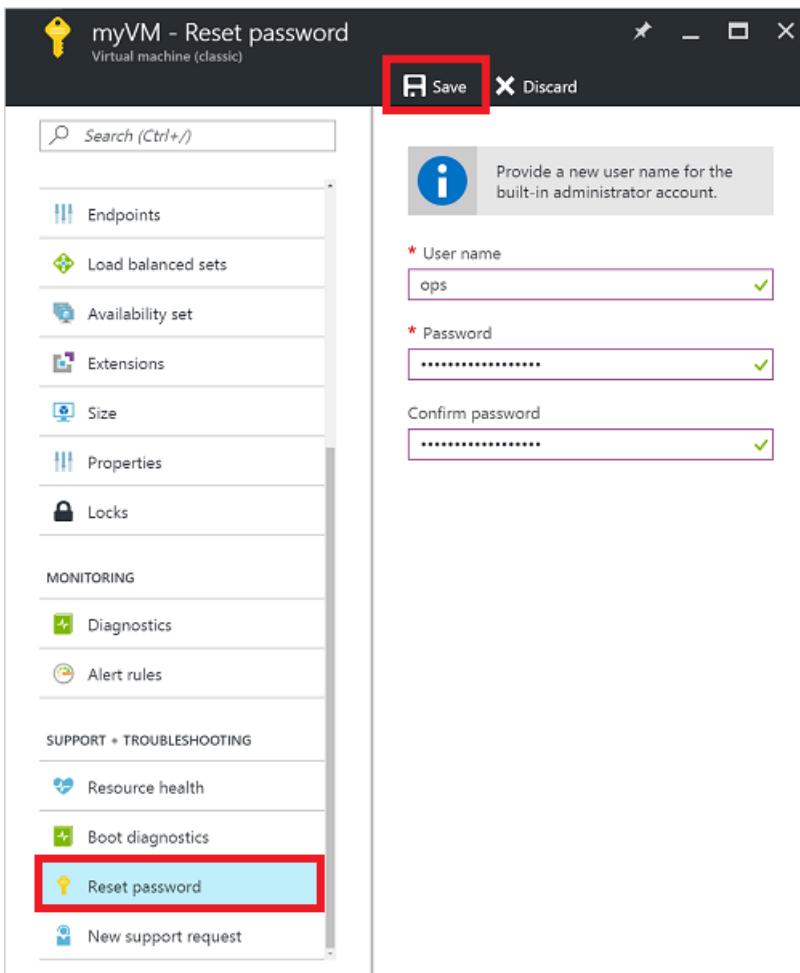
View History

What actions can you take?

- If you're having problems, use the [Troubleshoot tool](#) to get recommended solutions
- 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.

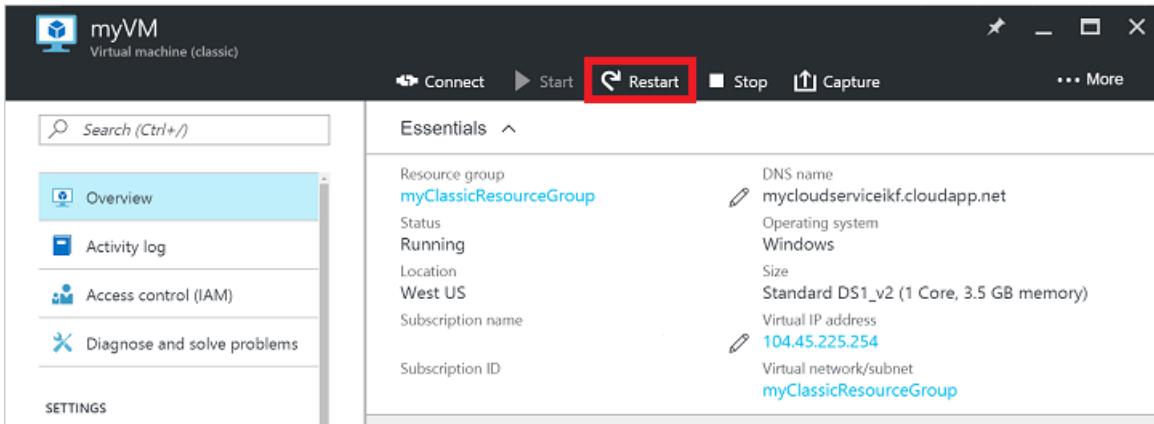
Select your VM in the Azure portal. Scroll down the settings pane to the **Support + Troubleshooting** section near bottom of the list. Click the **Reset password** button. Enter your username and a new password. Finally, click the **Save** button:



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](#).

- [Azure IaaS \(Windows\) diagnostics package](#)
- 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

1/17/2017 • 5 min to read • [Edit on GitHub](#)

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 can't connect to a Windows virtual machine (VM), you can reset the local administrator password or reset the Remote Desktop service configuration. 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 you have the latest PowerShell module installed on your work computer and are signed in to your Azure subscription. For detailed steps, read [How to install and configure Azure PowerShell](#).

TIP

You can check the version of PowerShell that you have installed by using:

```
Import-Module Azure, AzureRM; Get-Module Azure, AzureRM | Format-Table Name, Version
```

Ways to reset configuration or credentials

You can reset Remote Desktop services and credentials in a few different ways, depending on your needs. For VMs created using the Resource Manager deployment model:

- [Reset using the Azure portal](#)
- [Reset using Azure PowerShell](#)

For VMs created using the Classic deployment model:

- [Reset using the Azure portal](#)
- [Reset using Azure PowerShell](#)

Azure portal - Resource Manager

Select your VM by clicking **Browse > Virtual machines > your Windows virtual machine > All settings > Reset password**. The password reset blade is displayed:

This uses the VMAccess extension to reset the built-in administrator account and reset the Remote Desktop service configuration. [Learn more](#)

Mode [?](#)

Reset password

* User name [?](#)

ops

* Password

* Confirm password

MONITORING

Alert rules

Diagnostics

Diagram

SUPPORT + TROUBLESHOOTING

Resource health

Boot diagnostics

Reset password

Redeploy

New support request

Reset

Enter the username and a new password, then click **Save**. Try connecting to your VM again.

VMAccess extension and PowerShell - Resource Manager

Make sure you have Azure PowerShell 1.0 or later installed, and you have signed in to your account using the `Login-AzureRmAccount` cmdlet.

Reset the local administrator account password

You can reset the administrator password or user name by using the `Set-AzureRmVMAccessExtension` PowerShell command.

Create your local administrator account credentials by using the following command:

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

If you type a different name than the current account, the following VMAccess extension command renames the

local administrator account, assigns the password to that account, and issues a Remote Desktop logoff event. If the local administrator account is disabled, the VMAccess extension enables it.

Use the VM access extension to set the new credentials as follows:

```
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" -VMName "myVM" `  
-Name "myVMAccess" -Location WestUS -UserName $cred.GetNetworkCredential().Username `  
-Password $cred.GetNetworkCredential().Password -typeHandlerVersion "2.0"
```

Replace `myResourceGroup`, `myVM`, `myVMAccess`, and location with values relevant to your setup.

Reset the Remote Desktop service configuration

You can reset remote access to your VM by using either [Set-AzureRmVMExtension](#) or [Set-AzureRmVMAccessExtension](#), as follows. (Replace the `myResourceGroup`, `myVM`, `myVMAccess` and location with your own values.)

```
Set-AzureRmVMEextension -ResourceGroupName "myResourceGroup" -VMName "myVM" `  
-Name "myVMAccess" -ExtensionType "VMAccessAgent" -Location WestUS `  
-Publisher "Microsoft.Compute" -typeHandlerVersion "2.0"
```

Or:

```
Set-AzureRmVMAccessExtension -ResourceGroupName "myResourceGroup" -VMName "myVM" `  
-Name "myVMAccess" -Location WestUS -typeHandlerVersion "2.0"
```

TIP

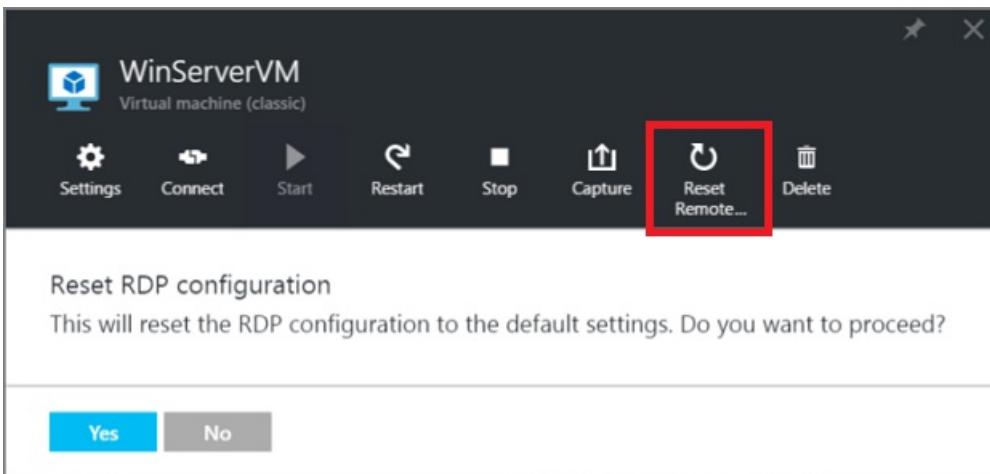
Both commands add a new named VM access agent to the virtual machine. At any point, a VM can have only a single VM access agent. To set the VM access agent properties successfully, remove the access agent set previously by using either `Remove-AzureRmVMAccessExtension` or `Remove-AzureRmVMEextension`.

Starting from Azure PowerShell version 1.2.2, you can avoid this step when using `Set-AzureRmVMExtension` with a `-ForceRerun` option. When using `-ForceRerun`, make sure to use the same name for the VM access agent as set by the previous command.

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](#).

Azure portal - Classic

For virtual machines created using the classic deployment model, you can use the [Azure portal](#) to reset the Remote Desktop service. Click: **Browse > Virtual machines (classic) > your Windows virtual machine > Reset Remote....** The following page appears.



You can also try resetting the name and password of the local administrator account. Click: **Browse > Virtual machines (classic) > your Windows virtual machine > All settings > Reset password**. The following page appears.

The screenshot shows two windows side-by-side. On the left is the 'Settings' blade for a virtual machine named 'WinServerVM'. It includes sections for SUPPORT & TROUBLESHOOTING (Audit logs, Boot diagnostics, Check health, Reset password), GENERAL (Properties, Disks, Network security group, IP addresses, Endpoints, Load balanced sets), and a search bar. The 'Reset password' item is highlighted with a red box. On the right is the 'Password reset' blade for the same virtual machine. It has a 'Save' button highlighted with a red box. It includes fields for User name, Password, and Confirm password, along with a note: 'Provide a new user name for the built-in administrator account.'

After you enter the new user name and password, click **Save**.

VMAccess extension and PowerShell - Classic

Make sure the VM Agent is installed on the virtual machine. The VMAccess extension doesn't need to be installed before you can use it, as long as the VM Agent is available. Verify that the VM Agent is already installed by using the following command. (Replace "myCloudService" and "myVM" by the names of your cloud service and your VM, respectively. You can learn these names by running `Get-AzureVM` without any parameters.)

```
$vm = Get-AzureVM -ServiceName "myCloudService" -Name "myVM"
write-host $vm.VM.ProvisionGuestAgent
```

If the **write-host** command displays **True**, the VM Agent is installed. If it displays **False**, see the instructions and a link to the download in the [VM Agent and Extensions - Part 2](#) Azure blog post.

If you created the virtual machine by using the portal, check whether `$vm.GetInstance().ProvisionGuestAgent` returns **True**. If not, you can set it by using this command:

```
$vm.GetInstance().ProvisionGuestAgent = $true
```

This command prevents the following error when you're running the **Set-AzureVMExtension** command in the next steps: "Provision Guest Agent must be enabled on the VM object before setting IaaS VM Access Extension."

Reset the local administrator account password

Create a sign-in credential with the current local administrator account name and a new password, and then run the `Set-AzureVMAccessExtension` as follows.

```
$cred=Get-Credential
Set-AzureVMAccessExtension -vm $vm -UserName $cred.GetNetworkCredential().Username ` 
    -Password $cred.GetNetworkCredential().Password | Update-AzureVM
```

If you type a different name than the current account, the VMAccess extension renames the local administrator account, assigns the password to that account, and issues a Remote Desktop sign-out. If the local administrator account is disabled, the VMAccess extension enables it.

These commands also reset the Remote Desktop service configuration.

Reset the Remote Desktop service configuration

To reset the Remote Desktop service configuration, run the following command:

```
Set-AzureVMAccessExtension -vm $vm | Update-AzureVM
```

The VMAccess extension runs two commands on the virtual machine:

```
netsh advfirewall firewall set rule group="Remote Desktop" new enable=Yes
```

This command enables the built-in Windows Firewall group that allows incoming Remote Desktop traffic, which uses TCP port 3389.

```
Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Terminal Server' -name "fDenyTSConnections" - 
Value 0
```

This command sets the fDenyTSConnections registry value to 0, enabling Remote Desktop connections.

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

Troubleshooting specific RDP error messages to a Windows VM in Azure

1/17/2017 • 4 min to read • [Edit on GitHub](#)

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](#).

- [Azure IaaS \(Windows\) diagnostics package](#)
- 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](#).

Troubleshoot Resource Manager deployment issues with creating a new Windows virtual machine in Azure

1/17/2017 • 4 min to read • [Edit on GitHub](#)

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 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.

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 Resource Manager deployment issues with restarting or resizing an existing Windows Virtual Machine in Azure

1/17/2017 • 2 min to read • [Edit on GitHub](#)

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 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.

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 an Azure virtual machine

1/17/2017 • 5 min to read • [Edit on GitHub](#)

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 Endpoint Connectivity problems

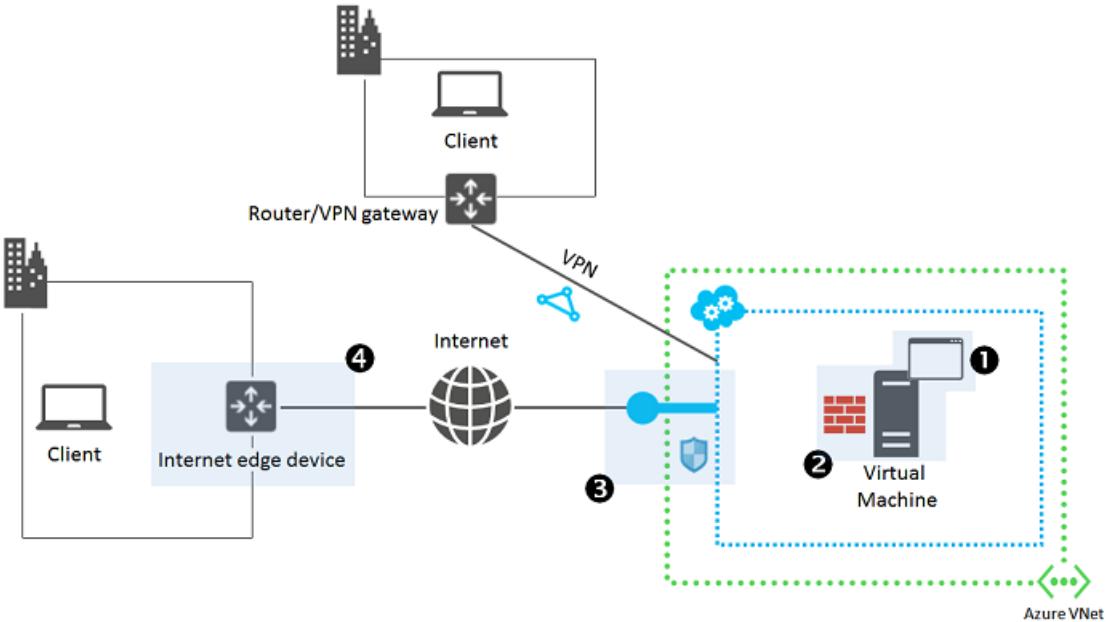
If you have problems connecting to an application, try the following general troubleshooting steps. After each step, try connecting to your application again:

- Restart the virtual machine
- Recreate the endpoint / firewall rules / network security group (NSG) rules
 - [Classic model - Manage Cloud Services endpoints](#)
 - [Resource Manager model - Manage Network Security Groups](#)
- 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.

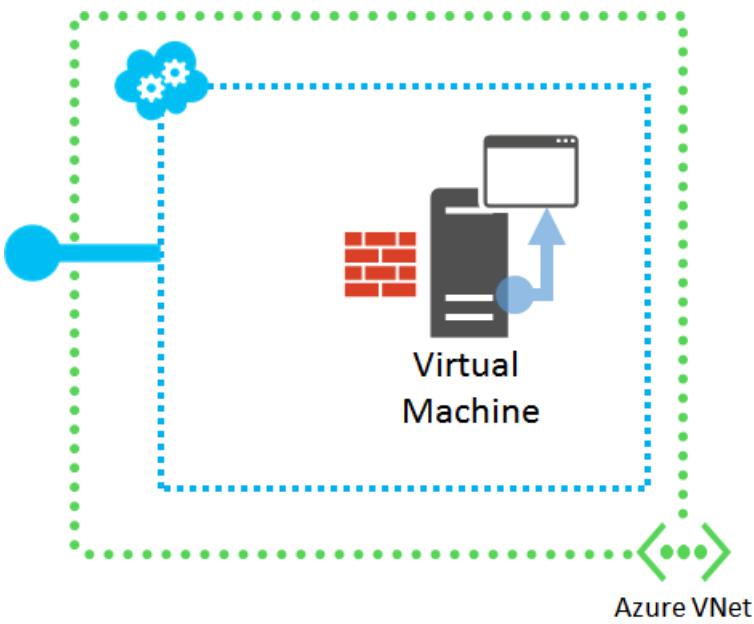


1. The application running on the Azure virtual machine.
 - Is the application itself running correctly?
2. The Azure virtual machine.
 - Is the VM itself running correctly and responding to requests?
3. Azure network endpoints.
 - Cloud service endpoints for virtual machines in the Classic deployment 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](#).

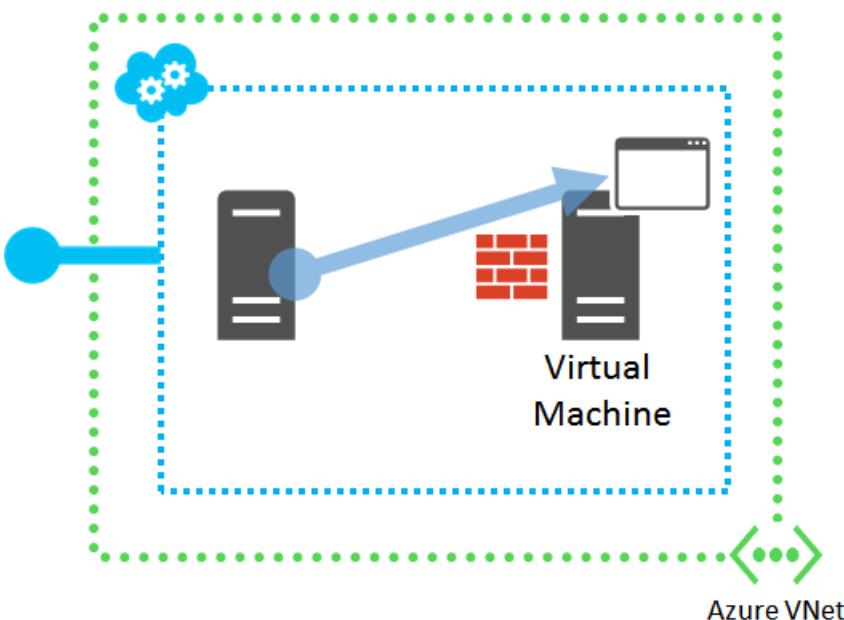
If you cannot access the application, verify the following settings:

- 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](#).

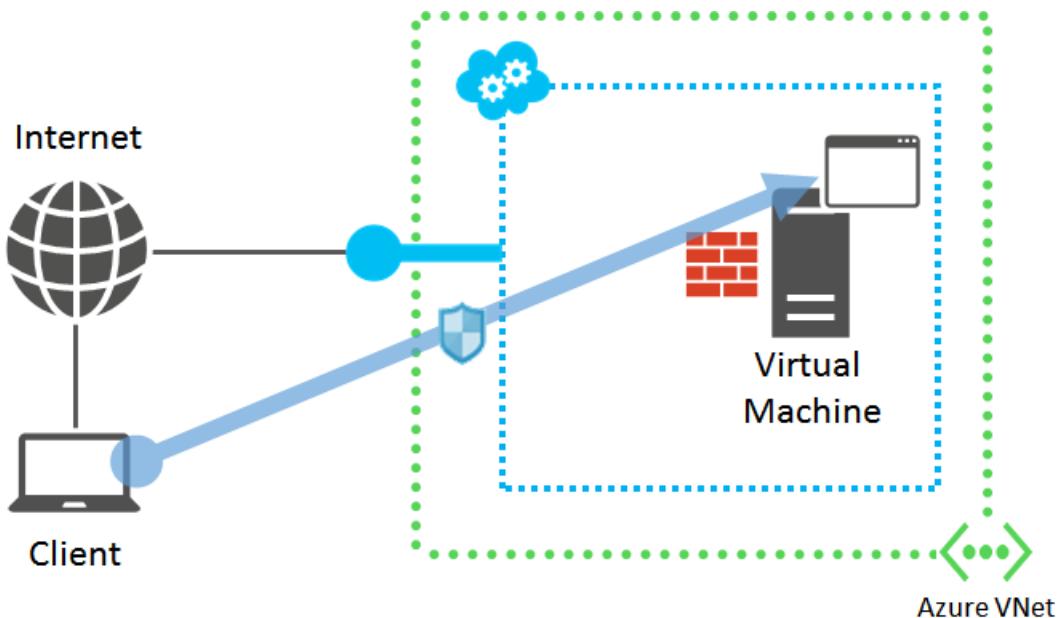
If you cannot access the application, verify the following settings:

- 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.

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 allocation failures when you create, restart, or resize Windows VMs in Azure

1/17/2017 • 12 min to read • [Edit on GitHub](#)

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

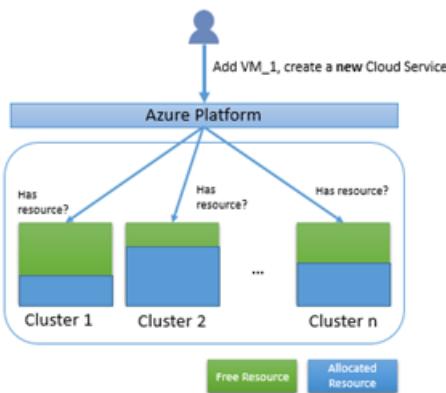
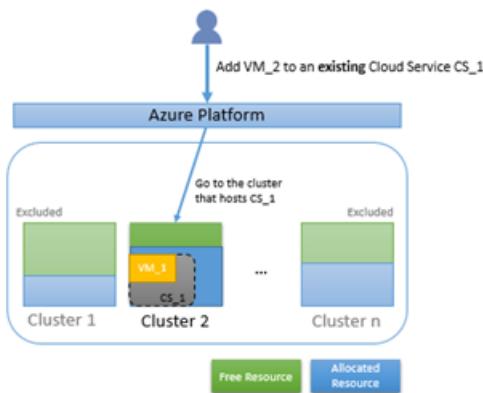


Diagram 2
Allocation Pinned to One Cluster



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

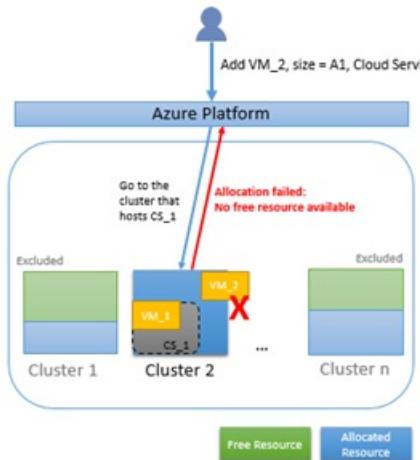
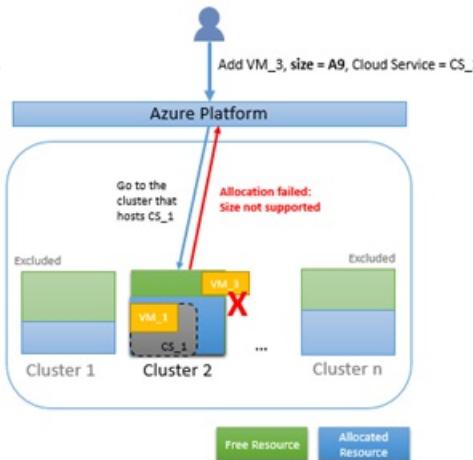


Diagram 4
Allocation Failed at Pinned Cluster:
Size not supported



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.

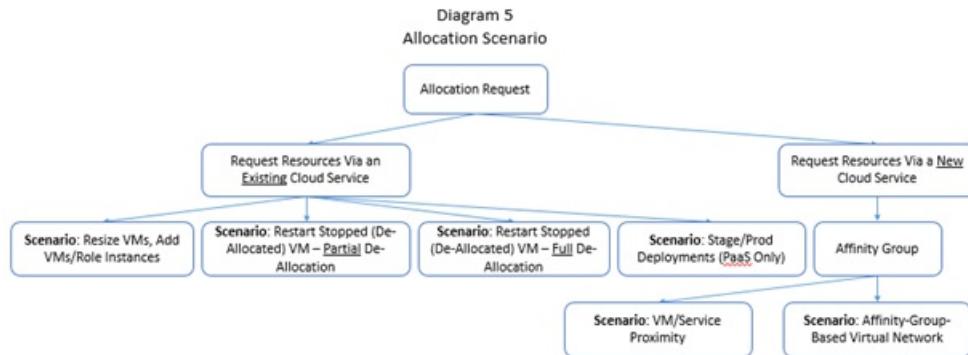
- Resize a VM or add VMs or role instances to an existing cloud service
- Restart partially stopped (deallocated) VMs
- Restart fully stopped (deallocated) VMs
- Staging/production deployments (platform as a service only)
- Affinity group (VM/service proximity)
- Affinity-group-based virtual network

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

1/17/2017 • 1 min to read • [Edit on GitHub](#)

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`:

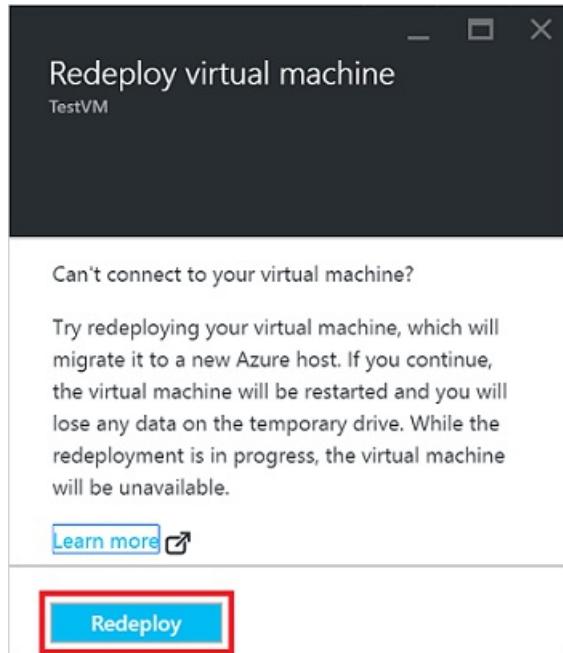
```
Set-AzureRmVM -Redeploy -ResourceGroupName "myResourceGroup" -Name "myVM"
```

Using Azure portal

1. Select the VM you wish to redeploy, and click the 'Redeploy' button in the 'Settings' blade. 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 VM are listed: 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/DNS name label (13.93.235.59/testv-westu-worh9mzo082g...), and Virtual network/subnet (testv-westu-worh9mzo082g vnet/testv-we...). 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, click the 'Redeploy' button:



3. The **Status** of the VM changes to *Updating* as the VM prepares to redeploy, as in the following example:

The screenshot shows the Azure portal interface for a virtual machine named "TestVM". The top navigation bar includes "Settings", "Connect", "Start", "Restart", "Stop", and "Delete" buttons. Below the navigation bar, a blue header bar displays the status "Updating". The main content area is titled "Essentials" and contains the following information:

| Setting | Value |
|----------------------------------|--|
| Resource group | TestRG |
| Status | Updating |
| Location | West US |
| Subscription name | [REDACTED] |
| Subscription ID | [REDACTED] |
| Computer name | TestVM |
| Operating system | Linux |
| Size | Standard D1 v2 (1 core, 3.5 GB memory) |
| Public IP address/DNS name label | 40.78.108.205/<none> |
| Virtual network/subnet | TestRG/default |

A blue button at the bottom right of the content area says "All settings →".

4. The **Status** then changes to *Starting* as the VM boots up on a new Azure host, as in the following example:

The screenshot shows the Azure portal interface for the same virtual machine "TestVM". The top navigation bar and "Essentials" section are identical to the previous screenshot. The status has changed to "Starting". The "Status" field is highlighted with a red box.

| Setting | Value |
|----------------------------------|--|
| Resource group | TestRG |
| Status | Starting |
| Location | West US |
| Subscription name | [REDACTED] |
| Subscription ID | [REDACTED] |
| Computer name | TestVM |
| Operating system | Linux |
| Size | Standard D1 v2 (1 core, 3.5 GB memory) |
| Public IP address/DNS name label | 40.78.108.205/<none> |
| Virtual network/subnet | TestRG/default |

A blue button at the bottom right of the content area says "All settings →".

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 |

At the bottom right of the content area is a blue button labeled '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](#).

Troubleshoot a Windows VM by attaching the OS disk to a recovery VM using Azure PowerShell

1/17/2017 • 6 min to read • [Edit on GitHub](#)

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-AzureRmVMDataDisk -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

1/17/2017 • 5 min to read • [Edit on GitHub](#)

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

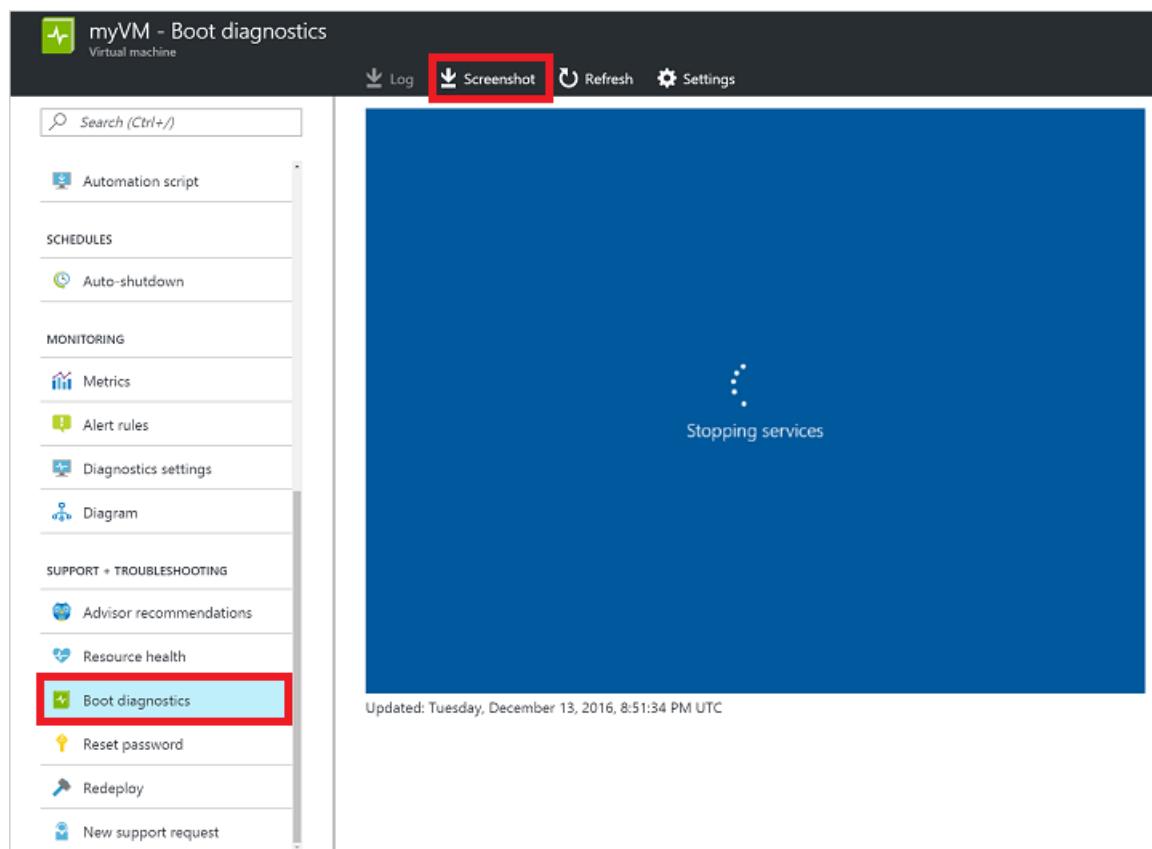
The troubleshooting process is as follows:

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 'mystorageaccountikf' storage account settings in the Azure portal. The left sidebar includes 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'SETTINGS' (with 'Access keys' and 'Configuration'), and a 'Delete' button. The main 'Essentials' section displays the resource group ('myresourcegroup'), status ('Available'), location ('West US'), subscription name, and subscription ID. 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 in the Blob service. The left pane shows the container's properties: storage account ('mystorageaccountikf'), blob service endpoint ('https://mystorageaccountikf.blob.core.windows.net'), and a table of blob names, URLs, and last modified times. The right pane lists blobs under 'Location: vhds'. It shows two entries: 'myDisk.vhd' and 'myVM2016101103813.vhd'. Both blobs are highlighted with red boxes.

Select your existing virtual hard disk from the list and copy the URL for use in the following steps:

Blob properties

NAME
myVM2016101103813.vhd

URL
<https://mystorageaccountikf.blob.core.windows.net/vhds/myVM2016101103813.vhd>

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**:

myVM
Virtual machine

Connect Start Restart Stop Delete

Search (Ctrl+/)

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

Essentials

Resource group
myResourceGroup

Status
Running

Location
West US

Subscription name
mySubscription

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>

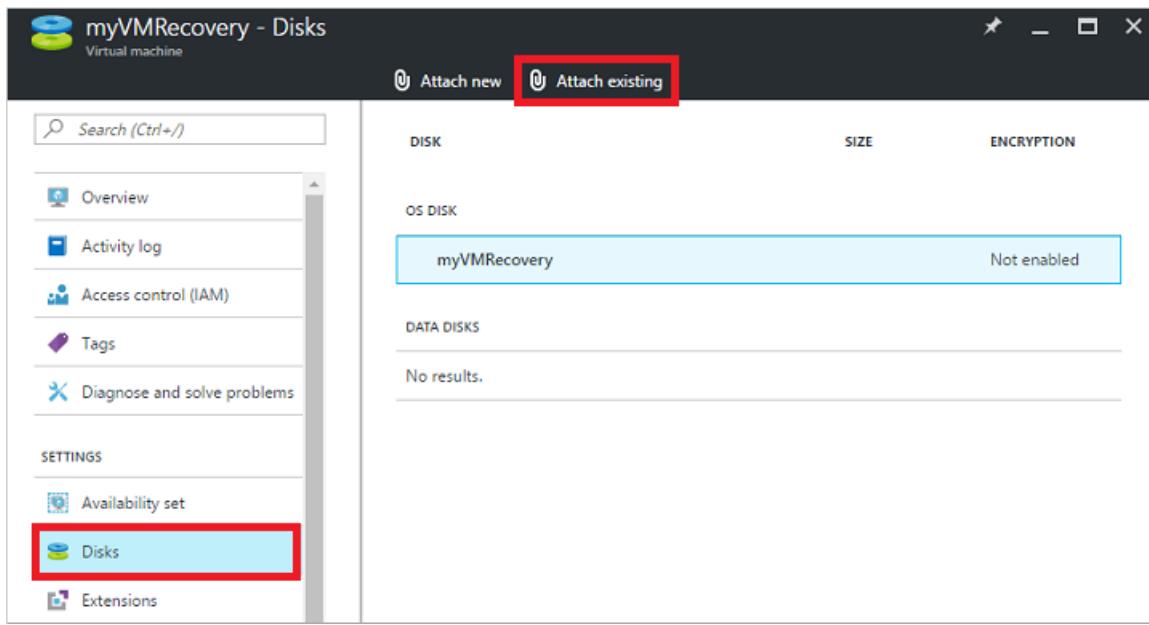
Virtual network/subnet
myVnet/mySubnet

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.

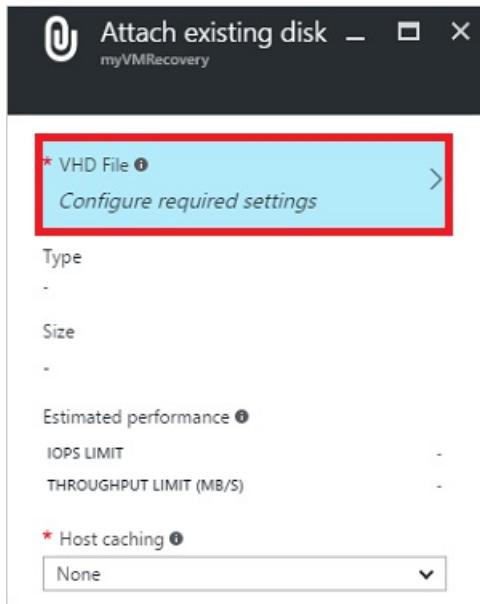
Attach existing virtual hard disk to another VM

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.

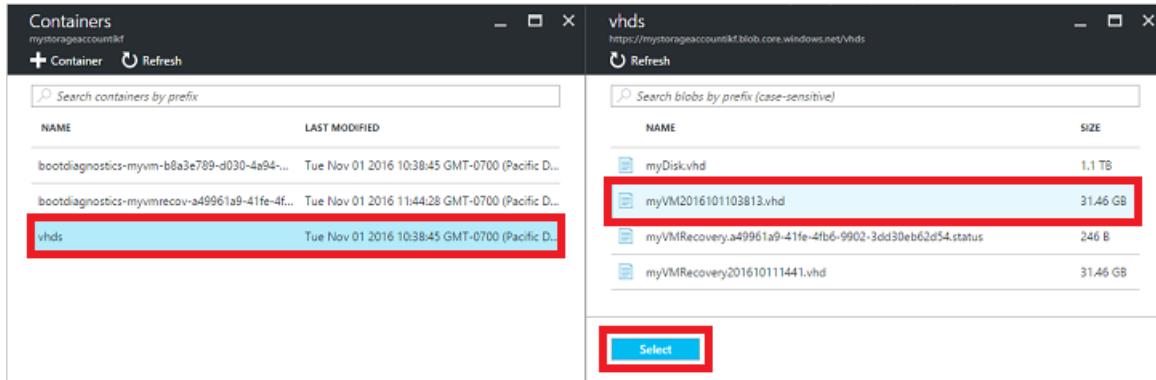
1. Select your resource group from the portal, then select your troubleshooting VM. Select **Disks** and then click **Attach existing**:



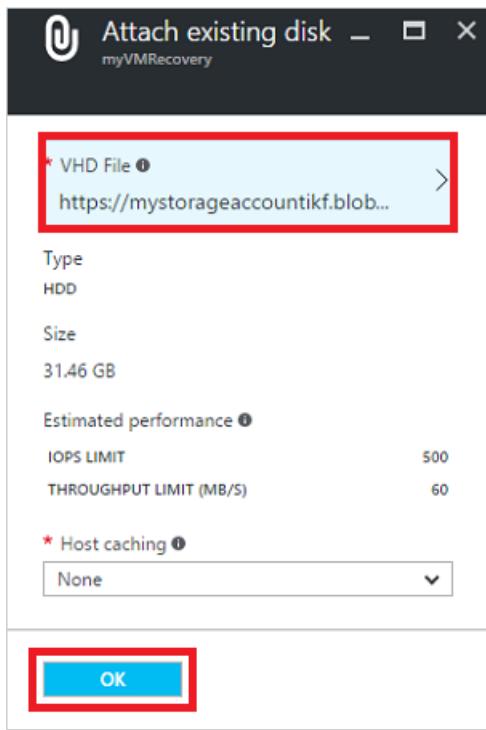
2. To select your existing virtual hard disk, click **VHD File**:



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:



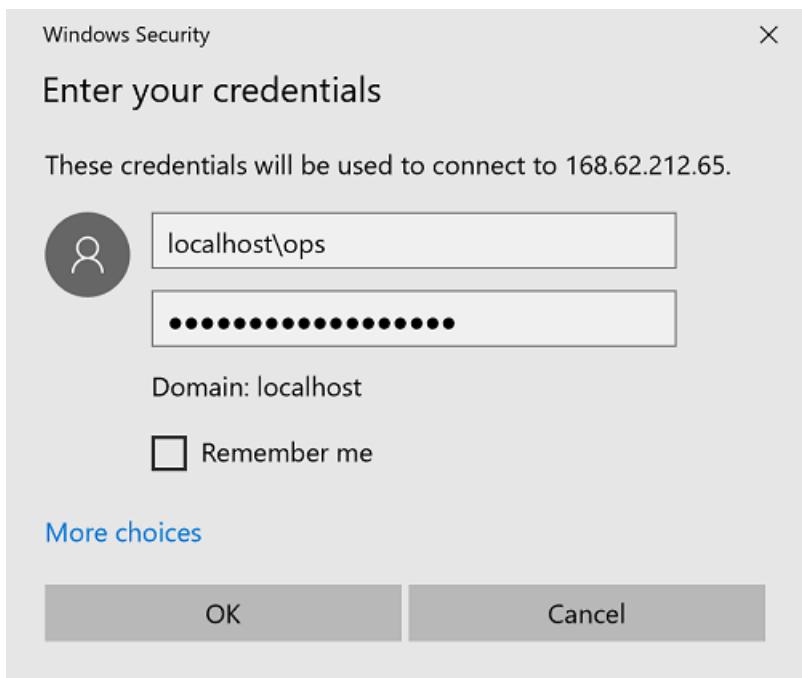
- After a few seconds, the **Disks** pane for your VM lists your existing virtual hard disk connected as a data disk:

The screenshot shows the 'Disks' pane for the 'myVMRecovery' virtual machine. It has a sidebar with options like Overview, Activity log, Access control (IAM), Tags, and Diagnose and solve problems. The main area shows a table of disks. The first row is an OS disk named 'myVMRecovery' with 'Not enabled' encryption. The second row is a data disk named 'myVM2016101103813', which is highlighted with a red box. The table has columns for DISK, SIZE, and ENCRYPTION.

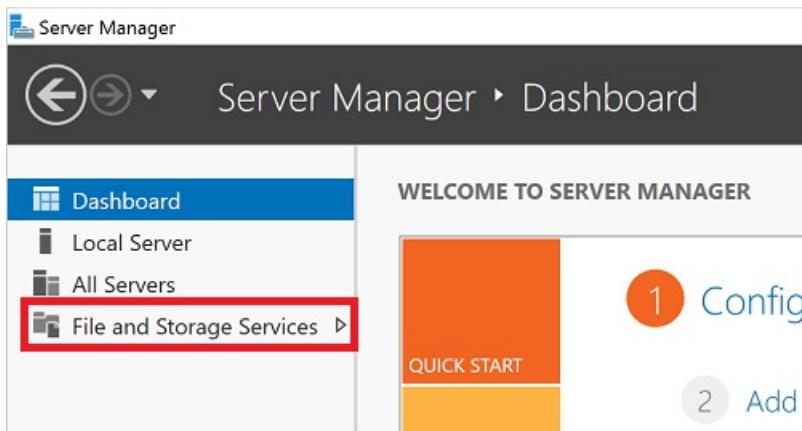
| DISK | SIZE | ENCRYPTION |
|---------------------------------|------|-------------|
| OS DISK
myVMRecovery | | Not enabled |
| DATA DISKS
myVM2016101103813 | | |

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. On the left, a navigation pane has 'Servers', 'Volumes', 'Disks' (which is selected and highlighted in blue), and 'Storage Pools'. The main area is titled 'DISKS' with the sub-section 'All disks | 3 total'. It lists three disks: 'myVMRecovery (3)' with entries 0, 1, and 2, all online, 127 GB capacity, 2.00 MB unallocated, and MBR partition style. Disk entry 2 is highlighted with a red box. Below this is a note 'Last refreshed on 12/13/2016 7:16:34 PM'. The bottom section is titled 'VOLUMES' with 'Related Volumes | 1 total'. It lists 'myVMRecovery (1)' with volume F: as fixed, 127 GB capacity, and 115 GB free space. This volume is also highlighted with a red box.

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 dashboard. The left navigation pane has 'Dashboard' (selected and highlighted in blue), 'Local Server', 'All Servers', and 'File and Storage Services' (with a red box around it). The main area is titled 'WELCOME TO SERVER MANAGER' and features a 'QUICK START' button. To the right, there are two large orange buttons: '1 Config' and '2 Add'.

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

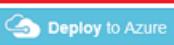
To create a VM from your original virtual hard disk, use [this Azure Resource Manager template](#). The template deploys a VM into an existing virtual network, using the VHD URL from the earlier command. Click the **Deploy to Azure** button as follows:

 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

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   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

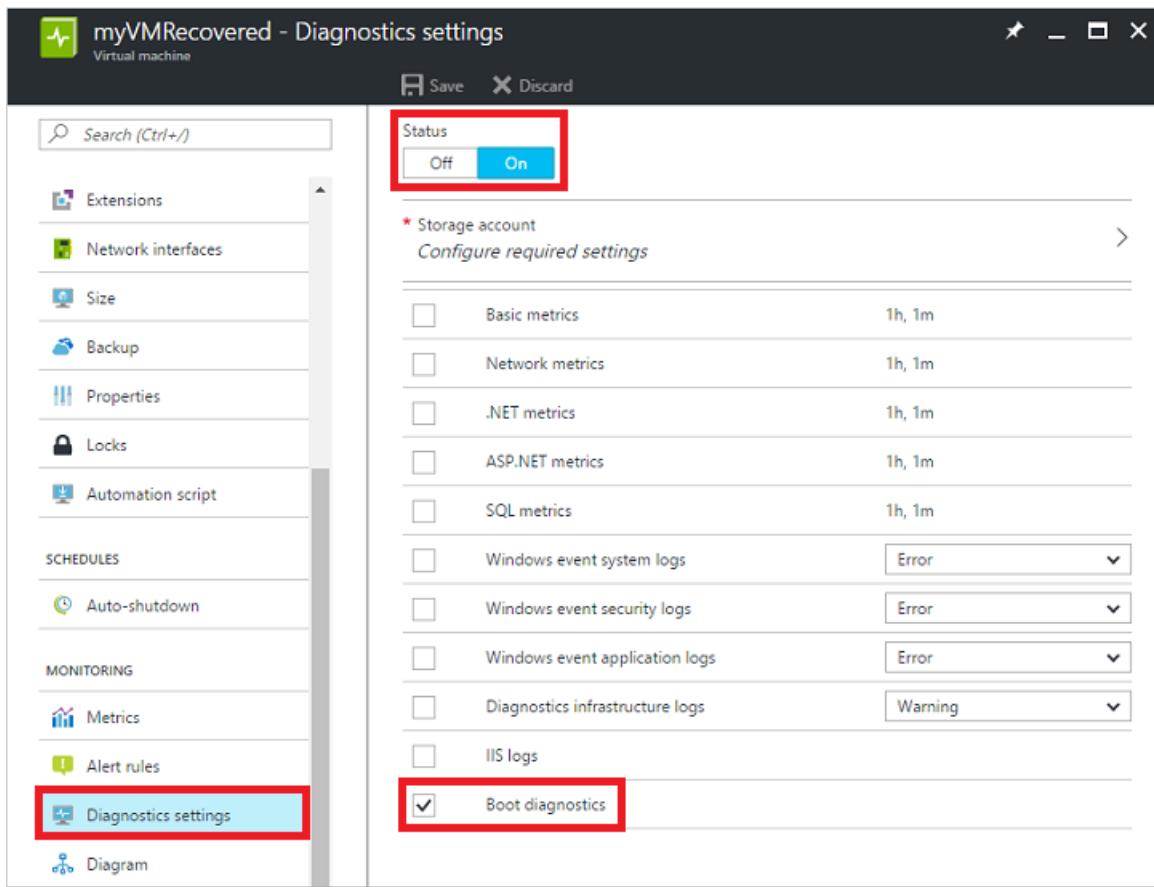
* Existing Virtual Network Name: myVnet

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](#).

Authoring Azure Resource Manager templates

1/17/2017 • 11 min to read • [Edit on GitHub](#)

This topic 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.

To view the template for resources you have already deployed, see [Export an Azure Resource Manager template from existing resources](#). For guidance on creating a template, see [Resource Manager Template Walkthrough](#). For recommendations about creating templates, see [Best practices for creating Azure Resource Manager templates](#).

A good JSON editor can simplify the task of creating templates. For information about using Visual Studio with your templates, see [Creating and deploying Azure resource groups through Visual Studio](#). For information about using VS Code, see [Working with Azure Resource Manager Templates in Visual Studio Code](#).

Limit the size 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.

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. |

| ELEMENT NAME | REQUIRED | DESCRIPTION |
|--------------|----------|--|
| resources | Yes | Resource types that are deployed or updated in a resource group. |
| outputs | No | Values that are returned after deployment. |

We examine the sections of the template in greater detail later in this topic.

Expressions and functions

The basic syntax of the template is JSON. However, expressions and functions extend the JSON that is available in the template. With expressions, you create values that are not strict literal values. Expressions are enclosed with brackets [] and [], and are evaluated when the template is deployed. Expressions can appear anywhere in a JSON string value and always return another JSON value. If you need to use a literal string that starts with a bracket [], you must use two brackets [].

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 values:

```
"variables": {
    "location": "[resourceGroup().location]",
    "usernameAndPassword": "[concat(parameters('username'), ':', parameters('password'))]",
    "authorizationHeader": "[concat('Basic ', base64(variables('usernameAndPassword')))]"
}
```

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.

You can use these parameter values throughout the template to set values for the deployed resources. Only parameters that are declared in the parameters section can be used in other sections of the template.

You define parameters with the following structure:

```

"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>"
    }
  }
}

```

| ELEMENT NAME | REQUIRED | DESCRIPTION |
|---------------|----------|---|
| parameterName | Yes | Name of the parameter. Must be a valid JavaScript identifier. |
| type | Yes | Type of the parameter value. See the list of allowed types after this table. |
| defaultValue | No | Default value for the parameter, if no value is provided for the parameter. |
| allowedValues | No | Array of allowed values for the parameter to make sure that the right value is provided. |
| minValue | No | The minimum value for int type parameters, this value is inclusive. |
| maxValue | No | The maximum value for int type parameters, this value is inclusive. |
| minLength | No | The minimum length for string, secureString, and array type parameters, this value is inclusive. |
| maxLength | No | The maximum length for string, secureString, and array type parameters, this value is inclusive. |
| description | No | Description of the parameter, which is displayed to users of the template through the portal custom template interface. |

The allowed types and values are:

- **string**
- **secureString**
- **int**
- **bool**
- **object**
- **secureObject**

- **array**

To specify a parameter as optional, provide a defaultValue (can be an empty string).

If you specify a parameter name in your template that matches a parameter in the command to deploy the template, there is potential ambiguity about the values you provide. Resource Manager resolves this confusion by adding the postfix **FromTemplate** to the template parameter. For example, if you include a parameter named **ResourceGroupName** in your template, it conflicts with the **ResourceGroupName** parameter in the [New-AzureRmResourceGroupDeployment](#) cmdlet. During deployment, you are prompted to provide a value for **ResourceGroupNameFromTemplate**. In general, you should avoid this confusion by not naming parameters with the same name as parameters used for deployment operations.

NOTE

All passwords, keys, and other secrets should use the **secureString** type. If you pass sensitive data in a JSON object, use the **secureObject** type. Template parameters with secureString or secureObject types cannot be read after resource deployment.

For example, the following entry in the deployment history shows the value for a string and object but not for secureString and secureObject.

| Inputs | |
|---------------|--|
| USERNAME | Example Person <input type="button" value="File"/> |
| PASSWORD | <input type="password"/> <input type="button" value="File"/> |
| REGULAROBJECT | {"test":"a"} <input type="button" value="File"/> |
| SECUREOBJECT | <input type="password"/> <input type="button" value="File"/> |

The following example shows how to define parameters:

```

"parameters": {
    "siteName": {
        "type": "string",
        "defaultValue": "[concat('site', uniqueString(resourceGroup().id))]"
    },
    "hostingPlanName": {
        "type": "string",
        "defaultValue": "[concat(parameters('siteName'), '-plan')]"
    },
    "skuName": {
        "type": "string",
        "defaultValue": "F1",
        "allowedValues": [
            "F1",
            "D1",
            "B1",
            "B2",
            "B3",
            "S1",
            "S2",
            "S3",
            "P1",
            "P2",
            "P3",
            "P4"
        ]
    },
    "skuCapacity": {
        "type": "int",
        "defaultValue": 1,
        "minValue": 1
    }
}

```

For how to input the parameter values during deployment, see [Deploy an application with Azure Resource Manager template](#).

Variables

In the variables section, you construct values that can be used throughout your template. Typically, variables are based on values provided from the parameters. You do not need to define variables, but they often simplify your template by reducing complex expressions.

You define variables with the following structure:

```

"variables": {
    "<variable-name>": "<variable-value>",
    "<variable-name>": {
        <variable-complex-type-value>
    }
}

```

The following example shows how to define a variable that is constructed from two parameter values:

```

"variables": {
    "connectionString": "[concat('Name=', parameters('username'), ';Password=', parameters('password'))]"
}

```

The next example shows a variable that is a complex JSON type, and variables that are constructed from other variables:

```

"parameters": {
  "environmentName": {
    "type": "string",
    "allowedValues": [
      "test",
      "prod"
    ]
  }
},
"variables": {
  "environmentSettings": {
    "test": {
      "instancesSize": "Small",
      "instancesCount": 1
    },
    "prod": {
      "instancesSize": "Large",
      "instancesCount": 4
    }
  },
  "currentEnvironmentSettings": "[variables('environmentSettings')[parameters('environmentName')]]",
  "instancesSize": "[variables('currentEnvironmentSettings').instancesSize]",
  "instancesCount": "[variables('currentEnvironmentSettings').instancesCount]"
}

```

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.

You define resources with the following structure:

```

"resources": [
  {
    "apiVersion": "<api-version-of-resource>",
    "type": "<resource-provider-namespace/resource-type-name>",
    "name": "<name-of-the-resource>",
    "location": "<location-of-resource>",
    "tags": "<name-value-pairs-for-resource-tagging>",
    "comments": "<your-reference-notes>",
    "dependsOn": [
      "<array-of-related-resource-names>"
    ],
    "properties": "<settings-for-the-resource>",
    "copy": {
      "name": "<name-of-copy-loop>",
      "count": "<number-of-iterations>"
    }
    "resources": [
      "<array-of-child-resources>"
    ]
  }
]

```

| ELEMENT NAME | REQUIRED | DESCRIPTION |
|--------------|----------|---|
| apiVersion | Yes | Version of the REST API to use for creating the resource. |

| ELEMENT NAME | REQUIRED | DESCRIPTION |
|--------------|----------|--|
| type | Yes | <p>Type of the resource. This value is a combination of the namespace of the resource provider and the resource type (such as Microsoft.Storage/storageAccounts).</p> |
| name | Yes | <p>Name of the resource. The name must follow URI component restrictions defined in RFC3986. In addition, Azure services that expose the resource name to outside parties validate the name to make sure it is not an attempt to spoof another identity. See Check resource name.</p> |
| location | Varies | <p>Supported geo-locations of the provided resource. You can select any of the available locations, but typically it makes sense to pick one that is close to your users. Usually, it also makes sense to place resources that interact with each other in the same region. Most resource types require a location, but some types (such as a role assignment) do not require a location.</p> |
| tags | No | <p>Tags that are associated with the resource.</p> |
| comments | No | <p>Your notes for documenting the resources in your template</p> |
| dependsOn | No | <p>Resources that must be deployed before this resource is deployed. Resource Manager evaluates the dependencies between resources and deploys them in the correct order. When resources are not dependent on each other, they are deployed in parallel. The value can be a comma-separated list of a resource names or resource unique identifiers. Only list resources that are deployed in this template. Resources that are not defined in this template must already exist. Avoid adding unnecessary dependencies as they can slow your deployment and create circular dependencies. For guidance on setting dependencies, see Defining dependencies in Azure Resource Manager templates.</p> |

| ELEMENT NAME | REQUIRED | DESCRIPTION |
|--------------|----------|---|
| properties | No | Resource-specific configuration settings. The values for the properties are the same as the values you provide in the request body for the REST API operation (PUT method) to create the resource. For links to resource schema documentation or REST API, see Resource Manager providers, regions, API versions, and schemas . |
| copy | No | If more than one instance is needed, the number of resources to create. For more information, see Create multiple instances of resources in Azure Resource Manager . |
| resources | No | Child resources that depend on the resource being defined. Only provide resource types that are permitted by the schema of the parent resource. The fully qualified type of the child resource includes the parent resource type, such as Microsoft.Web/sites/extensions . Dependency on the parent resource is not implied. You must explicitly define that dependency. |

Knowing what values to specify for **apiVersion**, **type**, and **location** is not immediately obvious. Fortunately, you can determine these values through Azure PowerShell or Azure CLI.

To get all the resource providers with **PowerShell**, use:

```
Get-AzureRmResourceProvider -ListAvailable
```

From the returned list, find the resource providers you are interested in. To get the resource types for a resource provider (such as Storage), use:

```
(Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Storage).ResourceTypes
```

To get the API versions for a resource type (such storage accounts), use:

```
((Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Storage).ResourceTypes | Where-Object ResourceTypeName -eq storageAccounts).ApiVersions
```

To get supported locations for a resource type, use:

```
((Get-AzureRmResourceProvider -ProviderNamespace Microsoft.Storage).ResourceTypes | Where-Object ResourceTypeName -eq storageAccounts).Locations
```

To get all the resource providers with **Azure CLI**, use:

```
azure provider list
```

From the returned list, find the resource providers you are interested in. To get the resource types for a resource provider (such as Storage), use:

```
azure provider show Microsoft.Storage
```

To get supported locations and API versions, use:

```
azure provider show Microsoft.Storage --details --json
```

To learn more about resource providers, see [Resource Manager providers, regions, API versions, and schemas](#).

The resources section contains an array of the resources to deploy. Within each resource, you can also define an array of child resources. Therefore, your resources section could have a structure like:

```
"resources": [
  {
    "name": "resourceA",
  },
  {
    "name": "resourceB",
    "resources": [
      {
        "name": "firstChildResourceB",
      },
      {
        "name": "secondChildResourceB",
      }
    ]
  },
  {
    "name": "resourceC",
  }
]
```

The following example shows a **Microsoft.Web/serverfarms** resource and a **Microsoft.Web/sites** resource with a child **Extensions** resource. Notice that the site is marked as dependent on the server farm since the server farm must exist before the site can be deployed. Notice too that the **Extensions** resource is a child of the site.

```

"resources": [
  {
    "apiVersion": "2015-08-01",
    "name": "[parameters('hostingPlanName')]",
    "type": "Microsoft.Web/serverfarms",
    "location": "[resourceGroup().location]",
    "tags": {
      "displayName": "HostingPlan"
    },
    "sku": {
      "name": "[parameters('skuName')]",
      "capacity": "[parameters('skuCapacity')]"
    },
    "properties": {
      "name": "[parameters('hostingPlanName')]",
      "numberOfWorkers": 1
    }
  },
  {
    "apiVersion": "2015-08-01",
    "type": "Microsoft.Web/sites",
    "name": "[parameters('siteName')]",
    "location": "[resourceGroup().location]",
    "tags": {
      "environment": "test",
      "team": "Web"
    },
    "dependsOn": [
      "[concat(parameters('hostingPlanName'))]"
    ],
    "properties": {
      "name": "[parameters('siteName')]",
      "serverFarmId": "[resourceId('Microsoft.Web/serverfarms', parameters('hostingPlanName'))]"
    },
    "resources": [
      {
        "apiVersion": "2015-08-01",
        "type": "extensions",
        "name": "MSDeploy",
        "dependsOn": [
          "[concat('Microsoft.Web/sites/', parameters('siteName'))]"
        ],
        "properties": {
          "packageUri": "https://auxmktp1ceprod.blob.core.windows.net/packages/StarterSite-modified.zip",
          "dbType": "None",
          "connectionString": "",
          "setParameters": {
            "Application Path": "[parameters('siteName')]"
          }
        }
      }
    ]
  }
]

```

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.

The following example shows the structure of an output definition:

```

"outputs": {
    "<outputName>" : {
        "type" : "<type-of-output-value>",
        "value": "<output-value-expression>"
    }
}

```

| ELEMENT NAME | REQUIRED | DESCRIPTION |
|--------------|----------|--|
| outputName | Yes | Name of the output value. Must be a valid JavaScript identifier. |
| type | Yes | Type of the output value. Output values support the same types as template input parameters. |
| value | Yes | Template language expression that is evaluated and returned as output value. |

The following example shows a value that is returned in the Outputs section.

```

"outputs": {
    "siteUri" : {
        "type" : "string",
        "value": "[concat('http://',reference(resourceId('Microsoft.Web/sites',
parameters('siteName'))).hostNames[0])]"
    }
}

```

For more information about working with output, see [Sharing state in Azure Resource Manager templates](#).

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](#).