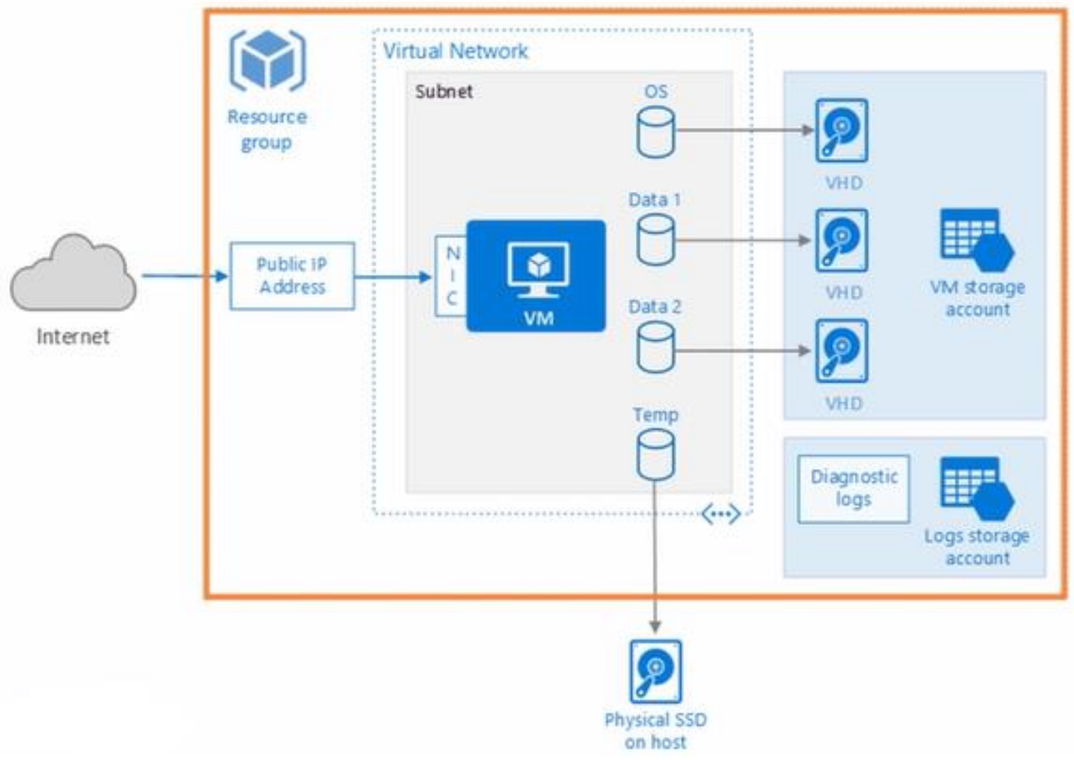
Agenda: Azure Virtual Machine

- Introduction
- Create a Windows Virtual Machine using Portal / PowerShell / ARM Templates
- Deploy popular application frameworks by using Azure Resource Manager templates
- Virtual Machine Disk
- Convert Unmanaged Disk to Managed Disk
- Generalizing VM and Capture VM Images
- Upload an on-premise VHD to Storage Account and attach to VM as Data Disk
- Working with Disk Snapshot
- VM Disk Types
- VM Sizes in Azure
- Configuring VM Disk Encryption
- Perform configuration management
    - VM Extensions & VM Agents
    - Custom Script Extensions
    - Desired State Configuration (DSC)
    - Access Extension
- Virtual Machine Scale Sets

Virtual Machine Introduction



Different ways to create a Windows VM with Azure Resource Manager (ARM)

1. Azure Portal
2. Azure PowerShell
3. Azure CLI
4. ARM Template
5. Programming using Rest API

Creating a VM using Azure portal:

The Management Portal provides many images and scripting tools that help you to create new virtual machines in Azure. The template images that are available in the portal are created and fully supported by either Microsoft or an authorized third-party.

Provisioning VMs to Azure requires planning. Before you create a single VM be sure you have thought about the following:

- Start with the network

- Name the VM
- Decide the location for the VM
- Determine the size of the VM
- Understanding the pricing model
- Storage for the VM
- Select an operating system

Walkthrough:

1. Azure portal → On the Hub menu, click New → Compute → Windows Server 2016 Datacenter.

Note: To find additional images, click Marketplace and then search or filter for available items.

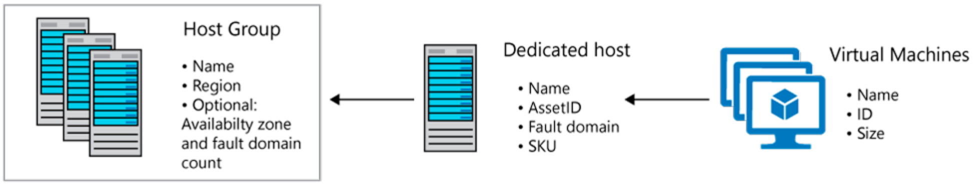
2. On the Windows Server 2012 R2 Datacenter page, under Select a deployment model = Resource Manager → Create.
3. Create virtual machine blade →
   - Basics → provide values for Name, Username and Password, Resource Group → OK
   - Size → Select an appropriate virtual machine size for your needs. Note that Azure recommends certain sizes automatically depending on the image you choose.
   - Settings to see storage and networking settings for the new virtual machine. For a first virtual machine you can generally accept the default settings.
   - Click Summary to review your configuration choices.
4. Click Create

Azure Dedicated Hosts

- Implement hardware isolation at the physical server level
- Control impact of maintenance events initiated by the Azure platform

To deploy highly available VMs to dedicated hosts:

- Create one or more host groups
- Create one or more hosts in each group
- Create a VM on each host



Quotas

- Limit vCPUs for dedicated hosts per region
- Support quota increase

Pricing

- Per dedicated host (regardless of the number of deployed VMs)
- Based on VM family, type, and region

Creating a VM using PowerShell

#Login to Azure Server.

Login-AzAccount

Set-AzContext -SubscriptionName "Visual Studio Enterprise"

# Create the Resource Group

$rgName= "PowershellDemoRG"

$location= "East US"

New-AzResourceGroup -Name $rgName -Location $location

A virtual machine is created based on a VHD image that can be selected from existing Azure prebuilt gallery images, or it can be created from a custom image that was uploaded into a storage account in Azure. Regardless of the selected method, **a storage account is required** to place the VHD(s) for the virtual machine. The storage account needs to be in the same Azure location as the virtual machine is created, so you can use the same variables for the location and the resource group name.

Create a Subnet, Virtual Network (VNet) and NIC Resources

VMs created with the Resource Manager Deployment model require a Resource Manager virtual network. If needed, create a new Resource Manager-based virtual network with at least one subnet for the new virtual machine.

```
$NICName="WebVM1-nic"
$PublicIPName = "WebVM1-ip"
$vNetName= "DemoVirtualNetwork"
$subnetName = "FrontEndSubnet"
$subnet=New-AzVirtualNetworkSubnetConfig -Name $subnetName -AddressPrefix 192.168.1.0/24

# Get the reference to the VNet that has the subnet being targeted
$vNet = New-AzVirtualNetwork -Name $vNetName -ResourceGroupName $rgName –Location $location -AddressPrefix "192.168.0.0/16" -Subnet $subnet
$subnet = Get-AzVirtualNetworkSubnetConfig -Name $subnetName -VirtualNetwork $vnet

#Add the Backend subnet to the VNet
Add-AzVirtualNetworkSubnetConfig -Name BackEndSubnet -VirtualNetwork $vnet -AddressPrefix 192.168.2.0/24
Set-AzVirtualNetwork -VirtualNetwork $vnet

# Create a public IP address object that can be assigned to the NIC (Network Interface Card)
$pubIP = New-AzPublicIpAddress -Name $PublicIPName -ResourceGroupName $rgName -Location $location -AllocationMethod Dynamic -DomainNameLabel "dss-webvm1"
#(change DomainNameLabel when you practice)

#Create the NIC attached to a subnet, with a public facing IP, and a private IP
$NIC = New-AzNetworkInterface -Name $NICName -ResourceGroupName $rgName -Location $location -SubnetId $vNet.Subnets[0].Id -PublicIpAddressId $pubIP.Id -PrivateIpAddress "192.168.1.4"
```

Create VM: Before you can actually create the virtual machine, you must specify the configuration information. In order to do this, you first create the configuration object that will store all the configuration information.

```
$vmName = "WebVM1"
$vmSize = "Standard_DS1"
# Create the virtual machine configuration object and save a reference to it
$vmConfig = New-AzVMConfig -VMName $vmName -VMSize $vmSize
```

Now that the virtual machine configuration object is created, the configuration information can be assigned to it. This includes defining the operating system, the base gallery image, and the previously created network adapter that you want to assign to the virtual machine. Optionally, you can also specify a name for the operating system VHD. If you do not specify one, Azure will assign a name automatically.

# In order to define the gallery image, the publisher, offer, and the SKU for the gallery image is needed. You can refer to the following article to understand how to do determine the available values to use: https://azure.microsoft.com/en-us/documentation/articles/resource-groups-vm-searching/#powershell .

```
# Prompt for credentials that will be used for the local admin password for the VM
$cred = Get-Credential -Message "Type the name and password of the local administrator account."
OR
# To provide fixed username and password.
$UserName = "dssadmin"
$SecurePassword = "Password@123"
$cred = New-Object System.Management.Automation.PSCredential ($UserName, $SecurePassword);

# Assign the operating system to the VM configuration
$vmConfig = Set-AzVMOperatingSystem -VM $vmConfig -Windows -ComputerName $vmName -Credential $cred -ProvisionVMAgent -EnableAutoUpdate

# For this example, a Windows Server 2016 R2 Datacenter image is specified in the configuration information.
$pubName = "MicrosoftWindowsServer"
$offerName = "WindowsServer"
$skuName = "2016-Datacenter"
$diskName = "WebVM1OSDisk"
# Assign the gallery image to the VM configuration
$vmConfig = Set-AzVMSourceImage -VM $vmConfig -PublisherName $pubName -Offer $offerName -Skus $skuName -Version "latest"

# Assign the UNMANAGED OS Disk name and location to the VM configuration
# $vmConfig = Set-AzVMOSDisk -VM $vmConfig DiskSizeInGB 128 -Name $diskName -VhdUri="https://<storage account reference of VHD> -CreateOption FromImage -Caching ReadWrite
# Assign the NIC to the VM configuration
$vmConfig = Add-AzVMNetworkInterface -VM $vmConfig -Id $NIC.Id
```

```
# With the virtual machine configuration defined, the actual virtual machine is created using the New-AzVM cmdlet with the configuration information passed as an argument.
New-AzVM -ResourceGroupName $rgName -Location $location -VM $vmConfig
```

You can check the status of the provisioning using Get-AzVM, passing it the resource group and VM name parameters. This retrieves the virtual machine configuration information. When the ProvisioningState value shows "Succeeded", then the virtual machine creation completed successfully, and the virtual machine should be in the running state.

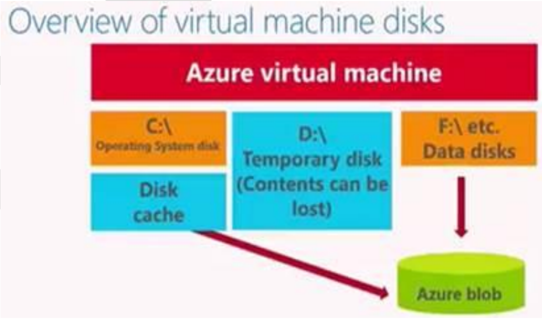
```
Get-AzVM -ResourceGroupName $RGPName -Name $vmName
```

Virtual Machine Disk Types

Just like any other computer, virtual machines in Azure uses disks as a place to store an operating system, applications, and data.

All Azure virtual machines have following disks:

1. Operating Dystem Disk
2. Temporary Disk.
3. Optional: One or more data disks.



Operating System Disks

- 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 2 TiB (Gen1). Gen2 Supports > 2TB

Temporary Disk

- Every virtual machine has a temporary disk that is **automatically** created for you.
- The size varies depending on tier size used for VM.
- On Windows virtual machines, this disk is labeled as the D: drive by default
- It's used for storing non-persistent storage / caching data (eg: pagefile.sys)

Data Disks (Optional)

- Every virtual machine can have data disks to store application data, or other data you need to keep.
- Maximum size of 32TiB. (Generation 1 VM)
- Data disks are registered as SCSI drives and are labeled with a letter that you choose **(E: onwards)**.
- 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.
- 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.

Important Notes:

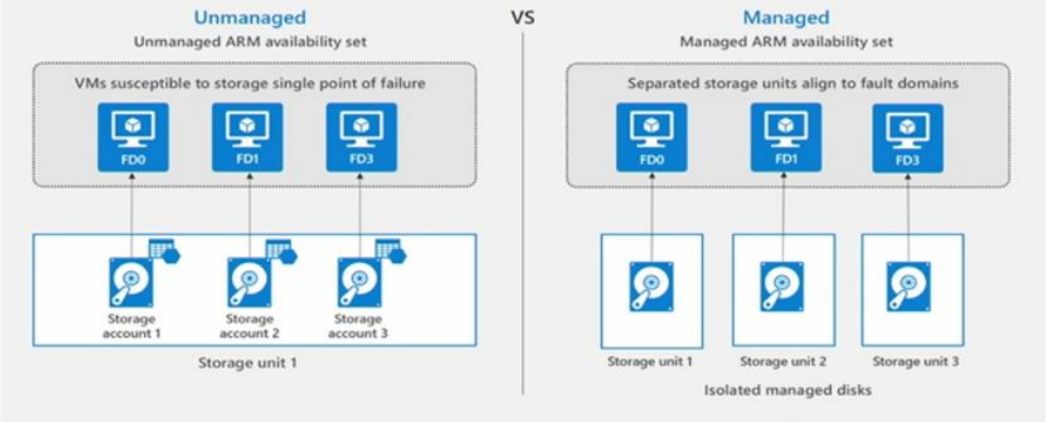
- Operating system and data disks are implemented as **page blob storage** in a storage account.
- To add a new disk to your VM you must provide Name, Type, Size, Location and Host Caching method.
- If your virtual machine is hosted on Server 2012 or above, you can use **storage pools** to virtualize storage by grouping industry-standard disks into pools, and then create **virtual disks called Storage Spaces** from the available capacity in the storage pools. Storage pools make it easy to grow or shrink volumes depending on your needs (and the capacity of the Azure data disks you have attached).

Types of Disks

1. **Unmanaged Disks:**
   - We need to create our own storage account.
   - VHD files are stored as page blobs in Azure storage accounts.
   - We need to ensure that we don't put too many disks in the same storage account because you could exceed the scalability targets of storage account (20,000 IOPS) resulting in VMs being throttled.
2. **Managed Disks:**

- Azure handles the storage account creation/management in the background for you, and ensures that you **do not have to worry about the scalability limits** of the storage account.
- You simply specify the disk size and the performance tier (Standard/Premium), and Azure creates and manages the disk for you. Even as you add disks or scale the VM up and down, you don't have to worry about the storage being used.
- You don't have to worry about placing the disks in multiple storage accounts to ensure that you stay within scalability limits for your storage accounts.

Enhanced availability – Availability set isolation



Steps to Add a Disk to VM:

1. More Services → Virtual machines → Select the VM created earlier.
2. VM → Settings → Disks → click **Attach new**.
3. On the Attach new blade, review the list of available settings. Fill in the following values and click OK.
   - Name: 01-DATADISK1
   - Type: HDD
   - Size (GiB): 100
   - Location: <Select your storage account>
   - Host caching: None
4. On the menu bar, monitor the alerts for progress as the new virtual disk is created and attached to the virtual machine.

Initialize a new data disk

1. Remotely connect to the VM → execute command **diskmgmt.msc** (Disk Management snap-in).

2. Disk Management will recognize that you have a new, un-initialized disk and the Initialize Disk window will pop up.
3. Make sure the new disk is selected and click **OK** to initialize it.
4. The new disk will now appear as **unallocated**. Right-click anywhere on the disk and select **New simple volume**. The **New Simple Volume Wizard** will start.
5. Go through the wizard, keeping all of the defaults, when you are done select **Finish**.
6. Close Disk Management.
7. You will get a pop-up that you need to format the new disk before you can use it. Click **Format disk**.
8. In the **Format new disk** dialog, check the settings and then click **Start**.
9. You will get a warning that formatting the disks will erase all of the data, click **OK**.
10. When the format is complete, click **OK**.

Detach a data disk using the portal

1. **OPTIONAL:** In the portal hub, select **Virtual Machines** → click **Stop** to deallocate the VM.
2. In the virtual machine blade, select **Disks**.
3. At the top of the **Disks** blade, select **Edit**.
4. In the **Disks** blade, to the far right of the data disk that you would like to detach, click the detach button.
5. After the disk has been removed, click Save on the top of the blade.
6. In the virtual machine blade, click **Overview** and then click the **Start** button at the top of the blade to restart the VM.

Virtual Machine Disk Types

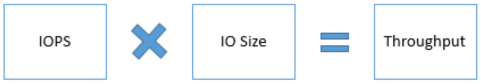
Disk Types:

1. **Premium SSD**: Azure Premium Storage delivers **high-performance, low-latency** disk support for virtual machines (VMs) with **input/output (I/O)-intensive** workloads. Best suitable to run performance-intensive workloads in applications like Dynamics CRM, Exchange Server, SAP, SharePoint, SQL Server, Oracle, Redis, which require **consistent** high performance and low latency.
2. **Standard SSD**: A **cost-effective** storage option optimized for workloads that need **consistent performance** at **lower IOPS** levels. Standard SSDs deliver **better availability**, **consistency**, **reliability** and **latency** compared to HDD Disks, and are suitable for Web servers, low IOPS application servers, lightly used enterprise applications, and Dev/Test workloads.
3. **Standard HDD:** For development and testing purpose. Also can be used for Snapshot.

IOPS is number of requests that your application is sending to the storage disks in one second. An input/output operation could be read or write, sequential or random.

Throughput or Bandwidth is the amount of data that your application is sending to the storage disks in a specified interval.

There is a relation between Throughput and IOPS as shown in the formula below.



Premium SSD Disk of 1TB Size = 5000 IOPS * 1 KB (Size) = 5000 KB/sec (Throughput)

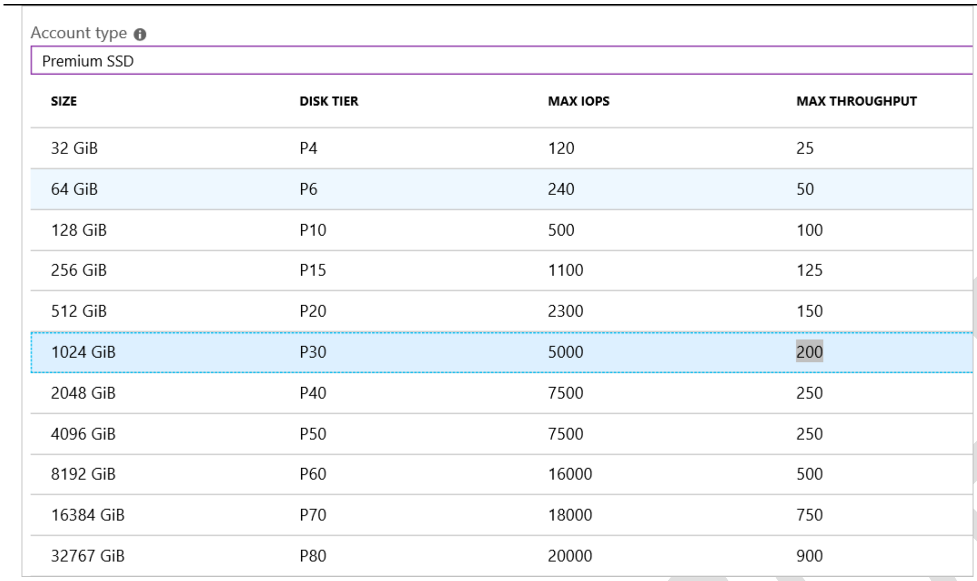
Standard SSD Disk of 1 TB Size = 500 IOPS * 1 KB (Size) = 500 KB/sec (Throughput)

Latency is the time it takes an application to receive a single request, send it to the storage disks and send the response to the client.

About Premium SSD

- In Azure, you can attach several premium storage disks to a VM. Using multiple disks gives your applications up to **256 TB** of storage per VM.
- With Premium Storage, your applications can achieve 80,000 I/O operations per second (IOPS) per VM, and a disk throughput of up to 2,000 megabytes per second (MB/s) per VM. Read operations give you very low latencies.
- Any object placed in a premium storage account will be a page blob. The page blob snaps to one of the supported provisioned sizes. This is why a premium storage account is not intended to be used to store tiny blobs.

Premium Storage disk limits: When you provision a premium storage disk, **the size of the disk determines** the maximum IOPS and throughput (bandwidth)



Account type

Premium SSD

SIZE	DISK TIER	MAX IOPS	MAX THROUGHPUT
32 GiB	P4	120	25
64 GiB	P6	240	50
128 GiB	P10	500	100
256 GiB	P15	1100	125
512 GiB	P20	2300	150
1024 GiB	P30	5000	200
2048 GiB	P40	7500	250
4096 GiB	P50	7500	250
8192 GiB	P60	16000	500
16384 GiB	P70	18000	750
32767 GiB	P80	20000	900

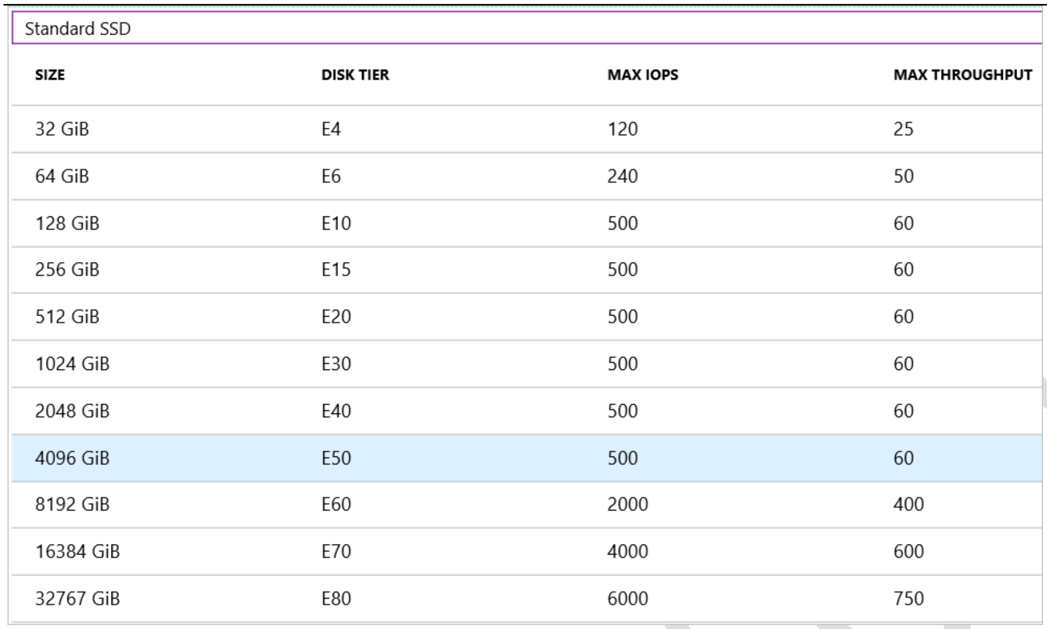
Note: Maximum throughput and IOPS are based on the VM limits (based on VM Size), not on the disk limits described in the preceding table.

Disk size: Azure maps the disk size (rounded up) to the nearest premium storage disk option, as specified in the table in the preceding section. For example, a disk size of 100 GB is classified as a P10 option. It can perform up to 500 IOPS, with up to 100-MB/s throughput.

Standard SSD Features

1. Standard SSDs are only available as Managed Disks. Unmanaged Disks and Page Blobs are **not supported** on Standard SSD.
2. Standard SSDs can be used with all Azure VMs.
3. Standard SSDs are built on the same Azure Disks platform, which has consistently delivered high availability and durability for disks. Azure Disks are designed for 99.999 percent availability.

The following table contains disk sizes, which are currently offered for Standard SSD.



Standard SSD

SIZE	DISK TIER	MAX IOPS	MAX THROUGHPUT
32 GiB	E4	120	25
64 GiB	E6	240	50
128 GiB	E10	500	60
256 GiB	E15	500	60
512 GiB	E20	500	60
1024 GiB	E30	500	60
2048 GiB	E40	500	60
4096 GiB	E50	500	60
8192 GiB	E60	2000	400
16384 GiB	E70	4000	600
32767 GiB	E80	6000	750

Standard SSDs are designed to provide single-digit millisecond latencies for most IO operations, and to deliver the IOPS and throughput up to the limits described in the above table 99% of the time. Actual IOPS and Throughput may vary sometimes depending on the traffic patterns.

IMPORTANT

Unmanaged Disk (we manage Storage Account) = Standard (HDD) / Premium (SSD) (Standard SSD is not supported)

Managed Disk (Azure manages Storage Account) = Standard HDD / **Standard SSD** / Premium SSD

If development machine is moved to Production, we can move from Standard to Premium.

Steps to Convert from Standard Unmanaged Disk to Premium Unmanaged Disk

1. Stop the VM
2. Create a **Premium** Storage Account.
3. Copy all VM Disks (All VHD files) to the New Storage Account (**problem: can take time**)
4. Create a **New VM** using the copied VM Disks.

Problem: We have to reconfigure the new VM and we might forget few configuration steps from old VM.

Steps to Convert from Standard Managed Disk to Premium Managed Disk

1. Stop the VM.
2. Update to Premium capable VM Size.
3. Update Storage Type to Premium.

4. Reboot.

PowerShell Script:

https://docs.microsoft.com/en-us/azure/virtual-machines/windows/convert-disk-storage

Note: We can downgrade the Premium Disk to Standard Disk also.

Configuring VM Disk Encryption

- You need to demonstrate to your trading partners that data stored on your Azure VMs cannot be accessed by unauthorized users, devices, or applications.
- ***Key management:*** *In Azure, your encryption keys can be* ***managed by Microsoft or the customer****.* Often the demand for customer-managed keys comes from organizations that need to demonstrate compliance with HIPAA, or other regulations. Such compliance may require that access to keys is logged, and that regular key changes are made and recorded.

Azure disk encryption technologies

The main encryption-based disk protection technologies for Azure VMs are:

Storage Service Encryption(SSE)

- Azure Storage Service Encryption (SSE) is an encryption service built into Azure used to **protect data at rest**.
- It cannot be disabled.
- The Azure storage platform automatically encrypts data before it's stored to several storage services, including Azure Managed Disks.
- Encryption is enabled by default using 256-bit AES encryption, and is managed by **the storage account administrator**.
- There is be **no noticeable performance impact** on the VM disk IO when using SSE.
- Managed disks with SSE are now the default, and mostly there should be no reason to change it.

Azure Disk Encryption:

- **Azure Disk Encryption** is a capability built into the Azure platform that allows you to **encrypt file system of volumes** residing on Windows and Linux virtual machine disks.

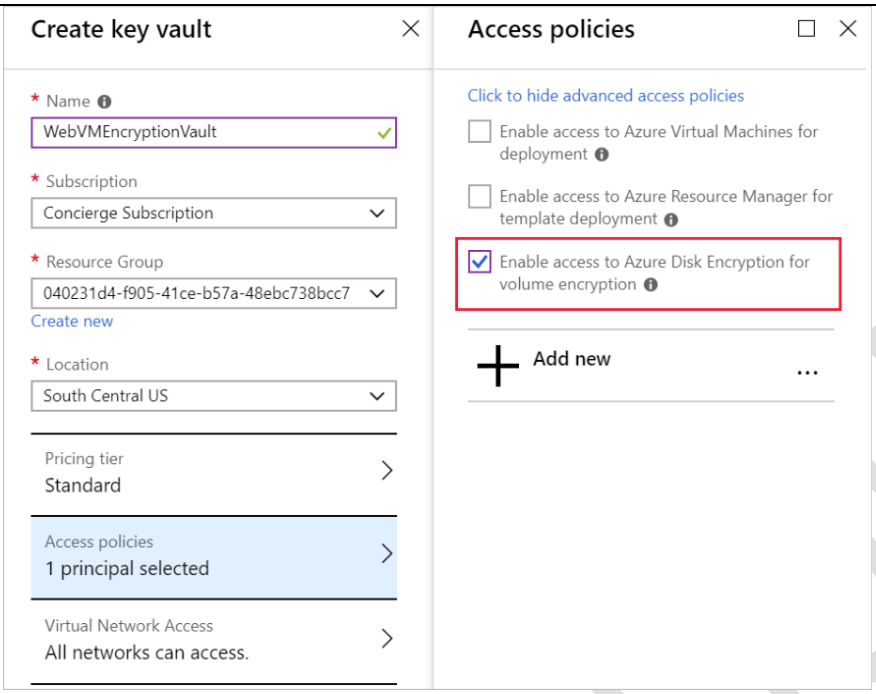
- Azure Disk Encryption (ADE) is **managed by the VM owner**. **VMs boot under customer-controlled keys and policies.**
- Bring your own encryption (BYOE) and bring your own key (BYOK) scenarios, in which the customers use their own encryption keys and store them in an **Azure key vault.**
- ADE ensures that all data on VM disks are encrypted at rest in Azure storage.
- It controls the encryption of Windows and Linux VM-controlled disks, using **BitLocker** on Windows VMs and **DM-Crypt** on Linux VMs.
- ADE makes use of VM operating system tools (BitLocker and DM-Crypt), so the VM itself has to do some work when encryption or decryption on VM disks is being performed. The impact of this additional VM CPU activity is typically negligible, except in certain situations. For instance, if you have a **CPU-intensive application**, there may be a case for leaving the OS disk unencrypted to **maximize performance**. In a situation such as this, you can store application data on a separate encrypted data disk, getting you the performance you need without compromising security.
- ADE is required for VMs backed up to the Recovery Vault.

Azure Disk Encryption is not supported for:

1. Basic tier virtual machines.
2. Integration with on-premises Key Management Service.
3. Linux virtual machines running Red Hat Enterprise Linux.
4. Content of Azure Files (Azure file share), Network file system (NFS), Dynamic volumes, and software based RAID configurations.

Steps:

1. Create a Key Vault Service



Create key vault

* Name: WebVMEncryptionVault

* Subscription: Concierge Subscription

* Resource Group: 040231d4-f905-41ce-b57a-48ebc738bcc7

Create new

* Location: South Central US

Pricing tier: Standard

Access policies: 1 principal selected

Virtual Network Access: All networks can access.

Access policies

Click to hide advanced access policies

☐ Enable access to Azure Virtual Machines for deployment

☐ Enable access to Azure Resource Manager for template deployment

☑ Enable access to Azure Disk Encryption for volume encryption

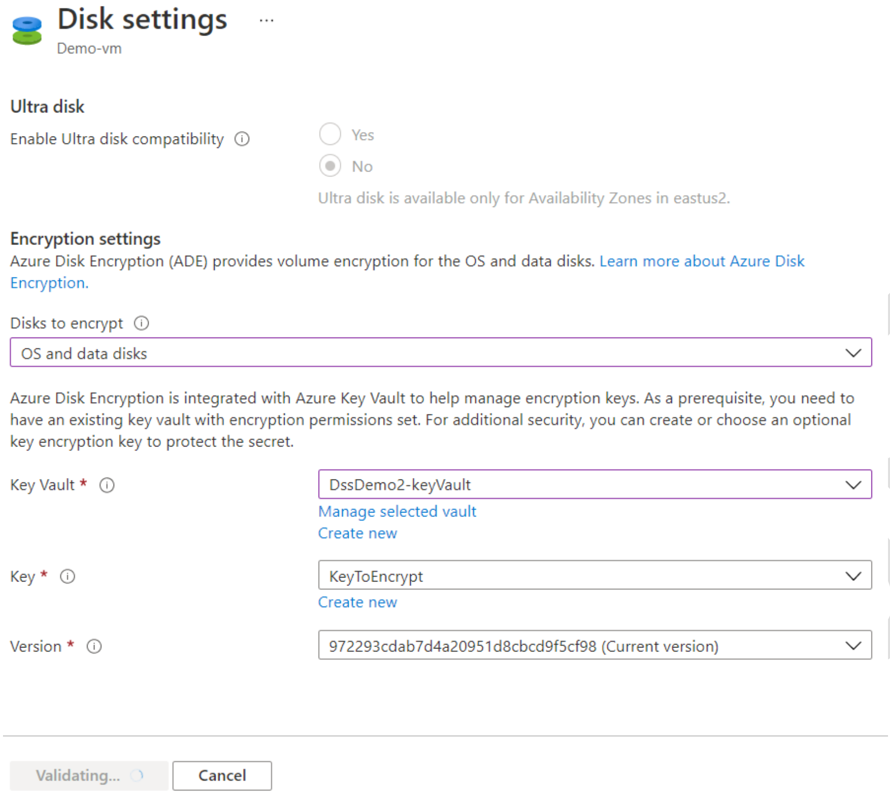
Azure needs access to the encryption keys or secrets in your key vault to make them available to the VM for booting and decrypting the volumes.

2. Add Key "myKey"

Encrypting Disk using ADE

The first time you encrypt a Windows VM, you can choose to encrypt either all disks or the OS disk only. On some Linux distributions, only the data disks may be encrypted. To be eligible for encryption, your Windows disks must be formatted as NTFS volumes.

3. VM → Disks → Settings



Disk settings ···
Demo-vm

Ultra disk

Enable Ultra disk compatibility ⓘ ○ Yes ● No

Ultra disk is available only for Availability Zones in eastus2.

Encryption settings

Azure Disk Encryption (ADE) provides volume encryption for the OS and data disks. Learn more about Azure Disk Encryption.

Disks to encrypt ⓘ: OS and data disks

Azure Disk Encryption is integrated with Azure Key Vault to help manage encryption keys. As a prerequisite, you need to have an existing key vault with encryption permissions set. For additional security, you can create or choose an optional key encryption key to protect the secret.

Key Vault * ⓘ: DssDemo2-keyVault
Manage selected vault
Create new

Key * ⓘ: KeyToEncrypt
Create new

Version * ⓘ: 972293cdab7d4a20951d8cbcd9f5cf98 (Current version)

OR

```
$keyVault = Get-AzKeyVault -VaultName dssdemo -ResourceGroupName DemoRG;


Set-AzVMDiskEncryptionExtension -ResourceGroupName DemoRG `
  -VMName "Demo-vm" `
  -DiskEncryptionKeyVaultUrl $keyVault.VaultUri `
  -DiskEncryptionKeyVaultId $keyVault.ResourceId `
  -KeyEncryptionKeyUrl (Get-AzKeyVaultKey -VaultName dssdemo -Name myKey).Key.kid `
  -KeyEncryptionKeyVaultId $keyVault.ResourceId `
  -VolumeType "Data" `
  -SkipVmBackup
```

Warning: You must take a snapshot or a backup of managed disks before you can turn on encryption. The SkipVmBackup flag specified below tells the tool that the backup is complete on managed disks. Without the backup, you will be unable to recover the VM if the encryption fails for some reason.

4. **Ensure that the Azure virtual machine disk is encrypted:**

   Azure Portal: Go to VM → Disks and you will see that **Encryption** is **Enabled**.

OR

PowerShell: Get-AzVmDiskEncryptionStatus -ResourceGroupName <resource-group> -VMName <vm-name>

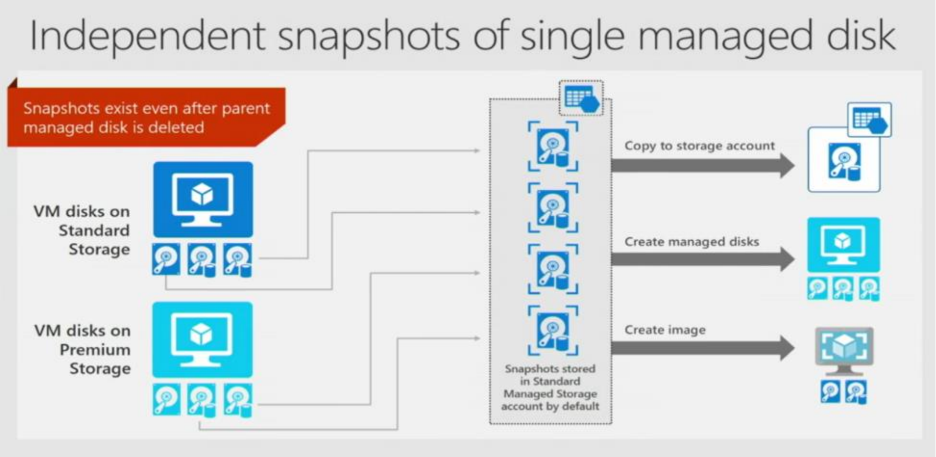
Working with Disk Snapshot

Snapshot can be created from Managed Disk and after creating the snapshot we can delete the managed disk and as snapshot can exist without Managed disk it is called as **Independent Snapshot**

Benefits of Snapshot

Snapshot can be copied onto another storage account in a different region.

Using Snapshot, we can restore our VM to a **Point-In-Time** by creating a Managed Disk from snapshot.


Independent snapshots of single managed disk
Snapshots exist even after parent managed disk is deleted
VM disks on Standard Storage
VM disks on Premium Storage
Snapshots stored in Standard Managed Storage account by default
Copy to storage account
Create managed disks
Create image

Create Snapshot - Copy a disk

1. Azure Portal → All Resources → Select the existing Managed Disk
2. Overview page → **+Create snapshot**
3. Create

Create a New Managed Disk from the snapshot

4. +Create a resource → Managed Disks → Create

Use the Managed Disk to create a new VM if its an OS Disk.

OR Use the Managed Disk to attach to an existing VM if its an Data Disk.

5. As steps provided above.

Realtime Usage Example:

Moving VM From One Subscription to another Subscription in a different Azure Account (may or may not be in same location).

Note: VM cannot be moved but its Snapshot of the Managed Disk can be.

Login to Source Azure Account

1. Stop the Source VM (Optional).
2. Take Snapshot of OS Disk of Source VM
3. **Export** the Snapshot: Generate the URL for Snapshot with SAS Token

Login to Destination Azure Account

4. In Target Account Create StorageAccount and Blob Container
5. Execute following PowerShell commands

   $destContext = **New-AzureStorageContext** -StorageAccountName "<targetstoragename>" -StorageAccountKey "<Target Storage Key>"

   Start-AzureStorageBlobCopy -AbsoluteUri <URL> -DestContainer "<TargetContainer>" -DestContext $destContext -DestBlob "VMDisk.vhd"
6. Create Managed Disk from the Storage Account BLOB
7. Create VM from the Managed Disk. (This works only if the Managed Disk is OS Disk and not data disk)

Note: Follow same steps as in 1, 2, 3, 4, 5 to move **Data** disk. Attach the Data disk to VM

Generalizing VM and Capture VM Images

What is a specialized virtual image?

- A specialized virtual image is a copy of a live virtual machine after it has reached a specific state. For example, a specialized image might contain a copy of the configured operating system, software, user accounts, databases, connection information, and other data for your system.
- You can use a specialized virtual image as a backup of your system at a particular point in time. If you need to recover after a catastrophic failure, or you need to roll back the virtual machine, you can restore your virtual machine from this image.
- If you use a specialized image to create a new virtual machine, the new virtual machine will retain all of the data from the image. That data includes the host name, user accounts, and other settings

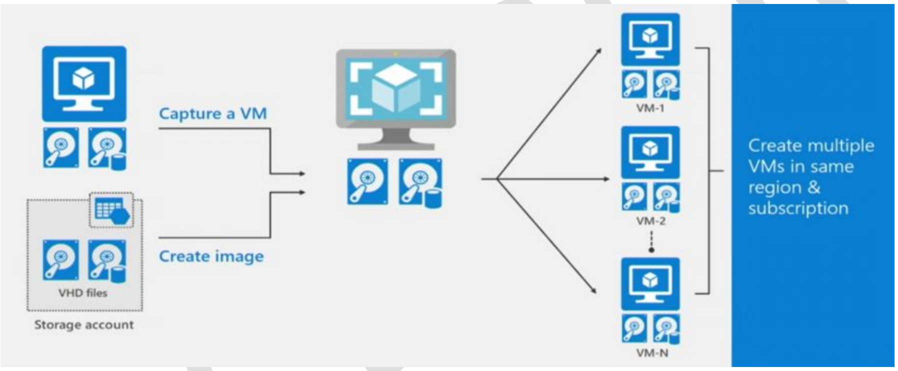
What is a generalized image?

You want to turn the VM you have created into a template from which you can stamp out new cloned instances of the VM, each of them unique with respect to certain settings in the operating system. For example, to scale out a cluster, you add new instances of the template VMs, each of them similarly configured but uniquely identified. In this case, each instance has its own identity and is therefore slightly different. In fact, when you create VMs from images available in the Marketplace, you utilize generalized images.

You can create your own custom virtual machine image in one of two ways:

1. If you're building an image from scratch by using Hyper-V, you first create a blank virtual disk, and then create a virtual machine with this disk. When you start the virtual machine, you install the operating system and any other additional software from source disks (typically DVDs) and other packages.
2. If you're customizing an image from Azure Marketplace, you build a virtual machine by using an existing image. The image provides the operating system and base functionality. You add your own software, operating system updates, and other packages as required.

Sysprep is a process that you could run into a windows installation that will reset the installation of the system and will provide an “out of the box experience” by removing all personal data and resetting several components.



Step 1: Generalizing a Windows VM

1. RDP to the VM
2. If ADE is enabled on the OS or Data Disk, disable the same

   Disable-AzVMDiskEncryption -ResourceGroupName 'Demo-RG' -VMName 'Demo-vm'

   VM → Extension → uninstall Azure Disk Encryption extension.
3. Change the directory to %windir%\system32\sysprep, and then run **sysprep.exe.**

   Note: Sysprep removes any machine-specific information and personal account information from the VHD.
4. In the **System Preparation Tool** dialog box, do the following:
   - In **System Cleanup Action**, select **Enter System Out-of-Box Experience (OOBE)** and make sure that **Generalize** is checked.
   - In **Shutdown Options**, select **Shutdown**.
   - Click **OK**.

OR

Execute the following command at command prompt

```
C:\> Sysprep.exe /oobe /generalize /shutdown
```

Note: Sysprep shuts down the virtual machine. Its status changes to **Stopped** in the Azure portal.

Step 2: Capture a MANAGED Image in the Portal

1. Azure Portal → Select the VM → **Capture**
2. Provide the relevant options → Create

OR

Step 2: Capture the VM as VM Image using PowerShell:

1. Open the Azure PowerShell 1.0.x and login to your Azure account.

```
Login-AzAccount
```

2. Now you will need to **deallocate** the resources used by this virtual machine.

```
Stop-AzVM -ResourceGroupName "DemoRG" -Name "WebVM1"
```

Note: Its status changes to **Stopped (deallocated)** in the Azure portal

3. Next you need to set the status of the virtual machine to *Generalized*. Note that you will need to do this because the generalization step above (sysprep) does not do it in a way that Azure can understand.

```
Set-AzVm -ResourceGroupName "DemoRG" -Name "WebVM1" -Generalized
```

4. Get the VM

```
$vm = Get-AzVM -Name $vmName -ResourceGroupName "DemoRG"
```

5. Create the image configuration.

```
$imageConfig = New-AzImageConfig -Location $location -SourceVirtualMachineId $vm.ID
```

6. Create the image

```
New-AzImage -Image $imageConfig -ImageName "NewImage" -ResourceGroupName "DemoRG"
```

Step 3: Use the Managed Image to Create a New VM

7. Select the New Image Created → Settings → Create VM

Upload an on-premise VHD to Storage Account and Attach to VM as Data Disk

You can migrate on-premises workloads to Azure by uploading the .VHD file to Azure and attaching it to an Azure virtual machine.

You must upload a .VHD file to Azure Storage before you can attach it to the virtual machine, and consider several factors, including that:

- VHD files must be from Hyper-V virtual machines. VHDX files are not supported as Azure virtual machine disks.

- You must generalize the on-premises virtual machine by using sysprep.exe.
- The maximum size allowed for the VHD is 1,023 GB.
- The .VHD file must be a fixed-size virtual disk.

Summary of Steps for uploading .VHD files from local machine:

1. Use Sysprep and Generalize the local virtual machine in Hyper-V.
2. Make the VHD as fixed size file.
3. Using **Convert-vhd** we can convert a .VHDX to .VHD file.
4. Upload the VHD to Azure Storage Account using **Add-AzVhd**
5. Attach the VHD as data disk to VM

Step1: Generalize the VM using sysprep

Step2: Steps to Convert the virtual disk (VHDX) to VHD and fixed size disk

1. Hyper-V Manager → select Local Computer → Menu Action → Edit Disk
2. Select the Virtual Disk to be converted
3. On Choose Action → **Convert** → Next
4. Select **Fixed size** → Next → Provide Path of New VHD → Finish.

OR

PowerShell Command to Convert VHDX to VHD

```
Convert-VHD –Path c:\test\MY-VM.vhdx –DestinationPath c:\test\MY-NEW-VM.vhd -VHDType Fixed
```

Step3: Upload the VHD to Azure Storage Account

Add-AzVhd –Destination http://<storageaccount>.blob.core.windows.net/vhd/Uploaded.vhd -LocalFile 'c:\....\Local.vhd' –ResourceGroupName DemoRG – Verbose –NumberofUploaderThreads 10

Note: The Storage account URL now can be used as Image for create a New VM.

Step4: Attach the disk as data disk to VM

1. Go to VM → Disk → Create Disk
2. Source type = **Storage blob**,

   Storage blog = http://<storageaccount>.blob.core.windows.net/vhd/Uploaded.vhd,

   OS type = None (data disk)
3. Create → Save

Virtual Machine Sizes in Azure

- If you resize the VM in the portal, the only content that will be "lost" is the data on the temporary disk (typically the D:\ drive). All data on the OS disk and any persistent data disks will be retained.
- If your VM(s) are deployed using the *Resource Manager (ARM) deployment model* you can resize VMs by first **stopping** your VM, selecting a new VM size and then restarting the VM.
- The VM size can be changed while the **VM is running**, as long as the new size is **available** in the **current hardware cluster** the VM is running on. The Azure portal makes this obvious by only showing you available size choices. The command line tools will report an error if you attempt to resize a VM to an unavailable size.
- Changing a running VM size will **automatically reboot (not stop deallocate and start)** the machine to complete the request.
- The size of the disk must be decided on the basis of required
  - vCPU
  - RAM
  - Datadisks
  - Max IOPS
  - Temp storage
  - Premium Disk and
  - Cost

Azure offers several virtual-machine size groups that offer different levels of compute resources

Type	**Sizes**	**Description**
General purpose	B, Dsv3, Dv3, DSv2, Dv2, Av2, DC	Balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers.
Compute optimized	Fsv2, Fs, F	High CPU-to-memory ratio. Good for medium traffic web servers, network appliances, batch processes, and application servers.
Memory optimized	Esv3, Ev3, M, GS, G, DSv2, Dv2	High memory-to-CPU ratio. Great for relational database servers, medium to large caches, and in-memory analytics.
Storage optimized	Lsv2, Ls	High disk throughput and IO. Ideal for Big Data, SQL, NoSQL databases, data warehousing and large transactional databases.
GPU	NV, NVv2, NC, NCv2, NCv3, ND,	Specialized virtual machines targeted for **heavy graphic rendering** and video editing, as well as model training and inferencing (ND) with deep learning. Available with single or multiple GPUs.

High performance compute	H	Fastest and most powerful CPU virtual machines with optional high-throughput network interfaces (RDMA).

Pricing:

There are two separate costs the subscription will be charged for every VM: **compute and storage**.

You're able to choose from two payment options for compute costs:

1. Pay as you go
2. Reserved Virtual Machine Instances for one or three years

   https://azure.microsoft.com/en-us/pricing/details/virtual-machines/windows/

When determining sizing for your Azure Virtual Machines, you should consider the following:

- There are two tiers of Azure Storage for storing your virtual machine's virtual disks: **Standard and Premium**. Premium offers higher I/O throughput, but at a higher pricing level. SSD disks are supported in Premium only.
- The size of the virtual machine affects the pricing, and the tier affects some capabilities.
- A1 Standard is the smallest size that is recommended for production workloads.
- When deploying a virtual machine for SQL Server Enterprise Edition, select a virtual machine with at least four CPU cores.
- You can also use the online **Pricing Calculator** tool which enables you to cost out different workloads and services in Microsoft Azure.

Configuration Management using VM Extensions

Microsoft offers a number of different methods that simplify and enhance management of both Windows and Linux operating systems hosted on Azure virtual machines.

In general, you can categorize management options for Azure VMs depending on the operating system support they provide.

Windows Management Options

- RDP (Remote Desktop)

Linux Management Option

- SSH (Secure Shell)

Cross Platform management options

- Azure PowerShell
- Azure CLI

- **VM Agent and VM Extensions**

VM Agents:

- The VM Agent is a set of lightweight software components running within the operating system of an Azure VM. Their primary purpose is to load additional programs and services known as **VM Extensions**.
- The Azure VM agent is **preinstalled** on **Azure Marketplace** images and can be installed on supported operating systems.
- If the agent is not installed at provisioning time, or if you have migrated a **virtual hard disk from on-premises**, you can manually install the agent on these virtual machines by downloading and installing the agent from Microsoft at http://go.microsoft.com/fwlink/?LinkID=394789&clcid=0x409

VM Extensions:

- 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.
- Extensions can be bundled with a new virtual machine deployment or run against any existing system.
- Azure VM extensions can be run by using
    - Azure portal.
    - PowerShell
    - Azure Resource Manager templates
    - Azure CLI
- There are many Available VM extensions and some which are popular are:
    - Custom Script Extension
    - Desired State Configuration
    - Access Extension
    - Disk Encryption Extension.
    - Backup Extension
    - DiagnosticsExtension
    - SqlServerExtension

To see a list of al VM extensions, run the following PowerShell commands.

```
get-command Set-Az*Extension* -Module Az.Compute
```

Custom Script Extension:

- The Custom Script Extension **downloads and executes** scripts on Azure virtual machines.
- The most common use of Custom Script extension involves applying **custom configuration settings** during VM provisioning.
- This extension is also useful for **stopping a VM** or **software installation**, or any other configuration / management task.
- Scripts can be downloaded from either **Azure Storage or GitHub**, or provided to the Azure portal at extension run time.
- The Custom Script Extension for Windows requires that the target virtual machine is connected to the internet.

Use Set-**AzVMExtension** to install the Custom Script Extension.

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

```
Set-AzVMExtension -ResourceGroupName $rgName `
  -ExtensionType CustomScriptExtension `
  -ExtensionName IIS `
  -VMName $vmName `
  -Location $location
  -Publisher Microsoft.Compute `
  -TypeHandlerVersion 1.4 `
  -SettingString '{"commandToExecute":"powershell Add-WindowsFeature Web-Server; powershell Add-Content
-Path \"C:\\inetpub\\wwwroot\\Default.htm\" -Value $($env:computername)"}' `

PS C:\>Remove-AzVMCustomScriptExtension -Name "IIS" -ResourceGroupName $rgName -VMName $vmName
Note: Removing the extension will not remove IIS
```

The **Set-AzVMCustomScriptExtension** cmdlet adds a custom script Virtual Machine Extension to a virtual machine. This extension lets you run your own scripts on the virtual machine

Install-IIS-ASPNET5.ps1

```
#Script to Create or Append content to File
```

```
$folder = "c:\temp"
$log = "c:\temp\AzureLog.txt"
$date = get-date

if (!(Test-Path $log)) {
   New-Item -Path $folder -ItemType Directory
   New-Item -Path $log -ItemType File
   Add-Content -Value "NEW LOG: Azure PowerShell Endpoint - $date" -Path $log
}
else {
   Add-Content -Value "EXSISITNG LOG: Azure PowerShell Endpoint - $date" -Path $log
}

#Script to install IIS on VM
Install-WindowsFeature Web-Server

#Install the .NET Core Windows Server Hosting bundle:
Invoke-WebRequest https://download.visualstudio.microsoft.com/download/pr/24847c36-9f3a-40c1-8e3f-
4389d954086d/0e8ae4f4a8e604a6575702819334d703/dotnet-hosting-5.0.6-win.exe -outfile
$env:temp\DotNetCore.WindowsHosting.exe
Start-Process $env:temp\DotNetCore.WindowsHosting.exe -ArgumentList '/quiet' -Wait
```

Example of Custom Script Extension using FileUrl:

```
Set-AzVMCustomScriptExtension -ResourceGroupName Demo-rg -VMName Demo-vm -Name "installDotNEet50" -
FileUri "https://mystorage123.blob.core.windows.net/scripts/InstallDotNet.ps1" -Run "InstallDotNet.ps1" -
Location "East US"
```

Example of Custom Script Extension using Storage Account Private Container

```
Set-AzVMCustomScriptExtension -ResourceGroupName Demo-rg -VMName Demo-vm -Name "InstallDotNet5" -
StorageAccountName "mystorageacc" -StorageAccountKey " /tB6ZaAW+DqDwku5Zg=="  -ContainerName
"scripts" -FileName "InstallDotNet.ps1" -Run "InstallDotNet.ps1" -Location "East US"
```

- In the above cmdlet change Location and set the value same as VM location.
- On completion, it creates a **c:\temp\AzureLog.txt** in VM.

Note: Restart the VM and test by executing the command "dotnet" at command prompt.

To see the deployment state of extensions for a given VM, run the following command:

```
PS C:\> Get-AzVMExtension -ResourceGroupName "DemoRG" -VMName "DemoVM" -Name "InstallDotNet5"
```

CLI Command to download and execute a PowerShell script that **installs IIS** and configures a basic home page.

```
az vm extension set \
  --resource-group 58aa17ce-beb9-4ddd-8a56-9bc690c17358 \
  --vm-name myVM \
  --name CustomScriptExtension \
  --publisher Microsoft.Compute \
  --settings '{"fileUris":["https://raw.githubusercontent.com/MicrosoftDocs/mslearn-welcome-to-
azure/master/configure-iis.ps1"]}' \
  --protected-settings '{"commandToExecute": "powershell -ExecutionPolicy Unrestricted -File configure-iis.ps1"}'
```

ARM Template for Creating a Virtual Machine along with Custom Script Extension:

https://github.com/Azure/azure-quickstart-templates/blob/master/201-vm-custom-script-output/azuredeploy.json

Following Section must be placed after "Properties" section of a VM Section in the ARM Template used above for creating a VM (Page 9 of this handout)

```
"resources": [
      . . .
      {
        "type": "extensions",
        "name": "myCustomScript",
        "apiVersion": "2016-04-30-preview",
        "location": "eastus",
        "properties": {
          "publisher": "Microsoft.Compute",
          "type": "CustomScriptExtension",
          "typeHandlerVersion": "1.4",
          "autoUpgradeMinorVersion": true,
          "settings": {
```

```
        "fileUris": [
         " https://mystorage123.blob.core.windows.net/scripts/InstallDotNet.ps1"
        ],
        "commandToExecute": "powershell -ExecutionPolicy Unrestricted -file InstallDotNet.ps1 "
       },
      "protectedSettings": {}
     },
    "dependsOn": [
      "[concat('Microsoft.Compute/virtualMachines/', 'myvm1')]"
    ]
   }
  ]
```

Desired State Configuration (DSC) Extension

- It allows you to **declaratively configure** the **state** of the virtual machine. Using built-in resource providers or custom providers with a DSC script enables you to declaratively configure settings such as **roles and features**, registry settings, files and directories, **firewall rules**, and most settings available to Windows.
- Due to its declarative nature, it bears some resemblance to Azure Resource Manager, however, while Azure Resource Manager templates deploy Azure resources such as VMs, **DSC targets operating systems running within these VMs**.
- DSC supports ARM templates, Azure PowerShell, and CLI.
- One of the compelling features of DSC is that, instead of writing logic to detect and correct the state of the machine, the providers do that work for you and make the system state as defined in the script. Every resource has a property named **Ensure** that can be set to **Present** or **Absent**. In the example below, the WindowsFeature resource will verify whether the Web-Server role is present on the target machine and if it is not, the resource will install it.

A DSC script consists of the following:

- The **Configuration** block. This is the outermost script block. You define it by using the Configuration keyword and providing a name. In this case, the name of the configuration is “IISInstall”.
- One or more **Node** blocks. These define the nodes (computers or VMs) that you are configuring. In the above configuration, there is one Node block that targets a computer named “localhost”.

- One or more **resource** blocks. This is where the configuration sets the properties for the resources that it is configuring. In this case, there are two resource blocks, each of which call the WindowsFeature resource.

For example, the following DSC script declares that the Web-Server role should be installed, along with the Web-Asp-Net45 feature.

Create a file D:\FeatureInstall.ps1 (extension must be ps1 only)

```
configuration IISConfig
{
  param
  (
    [string[]]$ComputerName='localhost'
  )
  Node $ComputerName
  {
    WindowsFeature IIS {
      Ensure = "Absent"
      Name = "Web-Server"
    }
    WindowsFeature AspNet45 {
      Ensure = "Present"
      Name = "Web-Asp-Net45"
    }
    File WebsiteContent {
      Ensure = 'Present'
      SourcePath = 'c:\test\index.htm'
      DestinationPath = 'c:\inetpub\wwwroot'
    }
  }
}
```

Deploying DSC: (Azure Cloud Shell cannot be used)

Execute the following PowerShell script after creating a Storage Account: **dssdemostorage**

```
$resourceGroup = "DemoRG"
```

```
$location = "South India"

$vmName = "DemoVM"

$storageName = "dssdemostorage"


Login-AzAccount

Set-AzContext -subscription <SubscriptionID>

#Publish the configuration script into user storage

Publish-AzVMDscConfiguration -ConfigurationPath d:\featureInstall.ps1 -ResourceGroupName $resourceGroup -

StorageAccountName $storageName -Force

# Note: Published file will have .zip extension


#Set the VM to run the DSC configuration

Set-AzVmDscExtension -ResourceGroupName $resourceGroup -VMName $vmName -

ArchiveStorageAccountName $storageName -ArchiveBlobName featureInstall.ps1.zip -AutoUpdate:$true -

ConfigurationName "IISConfig" -version 2.8
```

Note: **Get-AzureVMDscExtension** retrieves the DSC extension status of a particular VM.

DSC Using Template:

```
"resources": [

     {

       "name": "Microsoft.Powershell.DSC",

       "type": "extensions",

       "location": "[resourceGroup().location]",

       "apiVersion": "2015-06-15",

       "dependsOn": [

         "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"

       ],

       "tags": {

         "displayName": "dscExtension"

       },

       "properties": {

         "publisher": "Microsoft.Powershell",

         "type": "DSC",
         "typeHandlerVersion": "2.20",
         "autoUpgradeMinorVersion": false,
         "forceUpdateTag": "[parameters('dscExtensionUpdateTagVersion')]",
         "settings": {
           "configuration": {
             "url": "[concat(parameters('_artifactsLocation'), '/', variables('dscExtensionArchiveFolder'), '/', variables('dscExtensionArchiveFileName'))]",
             "script": "FeatureInstall.ps1",
             "function": "Main"
           },
           "configurationArguments": {
             "nodeName": "[variables('vmName')]"
           }
         },
         "protectedSettings": {
           "configurationUrlSasToken": "[parameters('_artifactsLocationSasToken')]"
         }
       }
     }
   ]
```

A **virtual machine scale set** node has a "**properties**" section with the "VirtualMachineProfile", "extensionProfile" attribute. DSC is added under "extensions"

Following is only sample and not functional:

```
"extensionProfile": {
      "extensions": [
        {
          "name": "Microsoft.Powershell.DSC",
          "properties": {
            "publisher": "Microsoft.Powershell",
            "type": "DSC",
            "typeHandlerVersion": "2.20",
            "autoUpgradeMinorVersion": false,
            "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')]"
            }
          }
        }
        ]
}
```

VM Access Extension:

- Currently, the extension can only be enabled using the **Set-AzureVMAccessExtension** cmdlet.
- This cmdlet can reset the local administrator name, password, and also enable Remote Desktop access if it is accidently disabled.
- This extension does not work against Active Directory domain accounts or on domain controllers.

```
Set-AzVMAccessExtension -Name "DemoAE"  -ResourceGroupName $rgName -Location $location -VMName
$vmName-TypeHandlerVersion "2.0" -UserName "dssadmin" -Password "Password@123"
```

Virtual Machine Scale Sets

- A VM scale set consists of a **group** of automatically provisioned Windows or Linux virtual machines that share **identical configurations** and deliver the same functionality to support a service or application.

- With a VM scale set, it is possible to have the number of virtual machines **increase or decrease**, adjusting dynamically to changes in demand for the service or application. To implement on demand autoscaling, you combine VM Scale Sets with Azure Insights **Autoscale**.
- With scale sets, all VM instances are created from the **same base OS image and configuration**. This approach lets you easily manage hundreds of VMs without additional configuration tasks or network management.
- It's easier to **build large-scale services** targeting big compute, big data, and containerized workloads.

General guidance

- You can create both Linux and Windows VM Scale Sets from the Azure Portal. These scale sets are automatically created with **load balancer NAT rules** to enable SSH or RDP connections.
- With un-managed disk, Max VM = 100 and with Managed Disk it can go upto 1000 VMs
- If you create and upload your own custom VM images, the limit is 300 VM instances.
- When you increase the number of virtual machines in a scale set, VMs are balanced across update and fault domains to ensure, maximum availability. Similarly, when you scale in, VMs are removed with maximum availability in mind.
- You can set the maximum, minimum and default number of VMs, and define triggers – action rules based on resource consumption.

To Create a Virtual Machine Scale Set

2. More Services → Virtual machine scale sets → +Add
3. Basic → Name = DemoScaleSet, single placement group = True . . . → Resource group, Create new = DemoRG → OK
4. Virtual Machine scaleset service settings: Provide Public IP, Domain name label, Operating system disk image = 2016-Datacenter, Auto Scaling=True → OK

Remote Desktop Login to VM of a Scale Set

5. Select **Load balancer** used by Scale set → Settings → Inbound NAT rules → Note the **IP address and Port number** of each instance.
6. Local Windows → Search Remote Desktop → Computer = Provide **IP:PortNo** → Connect
7. Provide the username and password which was provided when scale set was created → OK

To create a VMSS

```
New-AzVmss `
```

```
 -ResourceGroupName "myResourceGroup" `
 -Location "EastUS" `
 -VMScaleSetName "myScaleSet" `
 -VirtualNetworkName "myVnet" `
 -SubnetName "mySubnet" `
 -PublicIpAddressName "myPublicIPAddress" `
 -LoadBalancerName "myLoadBalancer" `
 -UpgradePolicyMode "Automatic" `
 -DataDiskSizeInGb 64,128
```

Note: This creates VM ScaleSet with two data disks. The first disk is *64* GB in size, and the second disk is *128* GB.

Create and use a custom image for virtual machine scale sets with Azure PowerShell

1. Create a New VM
2. RDP and install IIS and all other required softwares into it.
3. Generalize the VM: C:\Windows\system32\sysprep\sysprep.exe /oobe /generalize /shutdown
4. Create a custom VM Image form the source VM (**Capture** option in VM Overview blade)
5. Create a scale set from the custom VM Image

```
New-AzVmss `
 -ResourceGroupName "myResourceGroup" `
 -Location "EastUS" `
 -VMScaleSetName "myScaleSet" `
 -VirtualNetworkName "myVnet" `
 -SubnetName "mySubnet" `
 -PublicIpAddressName "myPublicIPAddress" `
 -LoadBalancerName "myLoadBalancer" `
 -UpgradePolicyMode "Automatic" `
 -ImageName "myImage"
```

Preparing the Disk using Custom Script Extension

The disks that are created and attached to your scale set VM instances are raw disks. Before you can use them with your data and applications, the disks must be prepared. To prepare the disks, you create a partition, create a filesystem, and mount them.

The following example executes a script from a GitHub sample repo on each VM instance with Add-AzVmssExtension that prepares all the raw attached data disks:

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

To confirm that the disks have been prepared correctly, RDP to one of the VM instances.

Install applications in virtual machine scale sets with Azure PowerShell

```
# Get information about the scale set
$vmss = Get-AzVmss `
    -ResourceGroupName "myResourceGroup" `
    -VMScaleSetName "myScaleSet"

# Add the Custom Script Extension to install IIS and configure basic website
$vmss = Add-AzVmssExtension `
 -VirtualMachineScaleSet $vmss `
 -Name "customScript" `
 -Publisher "Microsoft.Compute" `
 -Type "CustomScriptExtension" `
 -TypeHandlerVersion 1.8 `
 -Setting $customConfig

# Update the scale set and apply the Custom Script Extension to the VM instances
Update-AzVmss `
 -ResourceGroupName "myResourceGroup" `
 -Name "myScaleSet" `
 -VirtualMachineScaleSet $vmss
```

To Update app deployment:

Apply the Custom Script Extension configuration to the VM instances in your scale set again with **Add-AzVmssExtension** followed by **Update-AzVmss**

Attach a disk to existing scale set

```
# Get scale set object
$vmss = Get-AzVmss `
      -ResourceGroupName "myResourceGroup" `
      -VMScaleSetName "myScaleSet"

# Attach a 128 GB data disk to LUN 2
Add-AzVmssDataDisk `
 -VirtualMachineScaleSet $vmss `
 -CreateOption Empty `
 -Lun 2 `
 -DiskSizeGB 128

# Update the scale set to apply the change
Update-AzVmss `
 -ResourceGroupName "myResourceGroup" `
 -Name "myScaleSet" `
 -VirtualMachineScaleSet $vmss
```

What is Azure Resource Explorer?

- Download from http://resources.azure.com
- It's a great tool to view and modify resources you have created in your subscription. The tool is web-based and uses your Azure portal logon credentials.
- This tool is particularly useful in viewing Azure scale sets. With the tool you can see the individual virtual machines and their properties.