Agenda

- Prerequisites for PowerShell Management
- Azure Cloud Shell
- Managing Storage Accounts
- Managing Virtual Machines
- Managing Virtual Networks

What is Azure PowerShell?

- Extension of the Windows PowerShell platform and scripting language
- Provides cmdlets for simplifying and automating the management of Azure services
- Use the cmdlets to create, test, deploy, and manage solutions and services delivered through the Azure platform

Azure PowerShell benefits

- Benefits to using PowerShell to manage Azure services:
  - Orchestrate various Azure services together
  - Provides great flexibility when interacting with Azure resources
  - Helps in reducing complexity in the code

Installing Azure PowerShell

- You can install Azure PowerShell:
  - From Microsoft Web Platform (WebPI) Installer
  - From the PowerShell Gallery (ARM)

Installing from WebPI Installer

- Download WebPI installer and start the install
- The WebPI installer will install the Azure modules in %ProgramFiles(x86)%\Microsoft SDKs\Azure\PowerShell

Installing from the PowerShell Gallery (ARM)

- Install the Azure Resource Manager modules: `Install-Module AzureRM` Cmdlet will install the Azure modules in %ProgramFiles%\WindowsPowerShell\Modules
- **Note :**
  - You need to use an elevated (Run as Administrator) Windows PowerShell prompt
  - If an error occurs during install, you can manually remove the Azure* folders in your %ProgramFiles%\WindowsPowerShell\Modules folder, and try the installation again
  - Running Cmdlet `Install-Module` requires PackageManagement installation. PackageManagement modules can be installed independently, or they include into Windows Management Framework 5.0 (WMF 5.0). Windows 10 includes this by default.

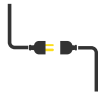
Check if Azure PowerShell is installed correctly

- Open a standard Windows PowerShell console, or PowerShell Integrated Scripting Environment (PowerShell ISE)
- Once the installation completes, your `$env:PSModulePath` setting should include the directories containing the Azure PowerShell cmdlets
- Azure PowerShell Modules and versions:

```
Get-Module -ListAvailable Azure*
```

Connecting Your Azure Subscription

- Open up your PowerShell console and type `Login-AzureRMAccount.` Input your credentials to attach your Azure subscription to the Azure PowerShell module.
- `Get-AzureRmSubscription` shows you all of the Azure subscriptions you have setup with PowerShell.
- If you have more than one subscription, you can set the subscription to be default by using the `Select-AzureRmSubscription` cmdlet. This allows you to set both the Azure subscription in your current session as well as all other PowerShell sessions.



Azure Profile

- `Save-AzureRmProfile -Path C:\AzureProfileFolder\azureprofile.json` will create the file azureprofile.json, which contains some information for your Azure account.

Azure Profile

- To login with a saved profile use:

```
Select-AzureRmProfile `
-Path C:\AzureProfileFolder\azureprofile.json
```

- When you troubleshoot Azure Profile at some point you would like to remove and/or rewrite default cached profile. You can find it here:
  `"%APPDATA%\Roaming\Windows Azure Powershell\AzureProfile.json"`

Azure Cloud Shell

- Azure Cloud Shell is an interactive, browser-accessible shell for managing Azure resources
- Provides the flexibility of choosing the shell experience that best suits the way you work
- Linux users can opt for a Bash experience, while Windows users can opt for PowerShell
- Access Azure cloud shell from within the Azure portal, at https://shell.azure.com, Azure CLI, Azure mobile app or VS Code Azure Account extension

How Azure Cloud Shell works

- Your host machine connects to a temporary host machine in Azure
- The temporary host machine has a 5GB Azure Files share mapped to it which is known as your home directory
- All files you store in your home directory, including scripts and user configuration files like .bashrc and .ps1, persist between sessions
- The machine hosting Cloud Shell is free, with a pre-requisite of a mounted Azure Files share
- Regular storage costs apply

Azure Cloud Shell Concepts

- Cloud Shell runs on a temporary host provided on a per-session, per-user basis
- Cloud Shell times out after 20 minutes without interactive activity
- Cloud Shell requires an Azure file share to be mounted
- Cloud Shell uses the same Azure file share for both Bash and PowerShell
- Cloud Shell is assigned one machine per user account
- Cloud Shell persists $Home using a 5-GB image held in your file share
- Permissions are set as a regular Linux user in Bash

Demo: Azure Cloud Shell

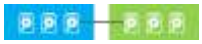
Microsoft

Managing Storage Accounts

Microsoft Services

Creating Azure Storage Accounts

- PowerShell Command for creating a storage Account
  `New-AzureRMStorageAccount`
- Command parameters required

- `-Resource Group` 
- `-SkuName` 
- `-Name` 
- `-Location` 

Creating Azure Storage Accounts

Before attempting to create a storage account, you can check to verify the availability of the storage account name in Azure

- Verify the availability of storage account name in Azure
  `Get-AzureRmStorageAccountNameAvailability -Name`
- Example: A new locally redundant ARM storage account

```
New-AzureRMStorageAccount -ResourceGroup ContosoRG1 -Name contosostore1
-SkuName Standard_LRS -Location "East US"
```

Setting a Default Storage Account

A subscription can have multiple storage accounts. One can be chosen as the default. The default storage account is used as the default for all commands in the same PowerShell session

- PowerShell command to set a default storage account
  `Set-AzureRMCurrentStorageAccount`
- Command parameters required

- **-Resource Group** 
- **-Name** 

Creating a Storage Account Context

An Azure storage context is an object in PowerShell to encapsulate the storage account credentials. To create a storage account context, you must obtain the account keys

- PowerShell command for retrieving the storage account keys

```
$storageAccountKey = Get-AzureRmStorageAccountKey
-ResourceGroupName $resourceGroup -Name $storageaccount
```

- PowerShell command for creating storage account context

```
$ctx = New-AzureStorageContext -StorageAccountName $storageaccount
-StorageAccountKey $storageAccountkey.value[0]
```

Creating a Container

To store blobs/files in Azure storage, you must first create an Azure storage container to store the blobs

To create a storage container with PowerShell you must first do one of the following:

- Choose a desired storage account and set as the default storage account
- Create an Azure storage context for the required storage account

Creating a Container

- PowerShell command for creating a storage container

```
New-AzureStorageContainer
```

- Command parameters required

Default Storage Account

`-Name` 

OR

Storage Account Context

`-Context` 

`-Name` 

Uploading VHDs to Azure Storage

- Uploading a VHD to Azure Storage requires you to know the URI of the storage container where the VHD will be stored.
- To gather the URI: Login to the Azure portal and copy the Uri of the storage account and container or use the URI syntax below.

Container
Uri Syntax: http://<storage account>.blob.core.windows.net/<container>
Example: http://ContosoStorage1.blob.core.windows.net/VHDUploads

Blob
Uri Syntax: http://<storage account>.blob.core.windows.net/<container>/<blob>
Example: http://ContosoStorage1.blob.core.windows.net/VHDUploads/disk1.vhd

Uploading VHDs to Azure Storage

- PowerShell command for uploading VHDs to storage
  Add-AzureRMVhd
- Command parameters required

- Resource Group 

- LocalFilePath 

- Destination 

Downloading VHDs from Azure Storage

- PowerShell command for downloading VHD from storage
  Save-AzureRMVhd
- Command parameters required

- Resource Group 

- LocalFilePath 

- SourceUri 

Removing Azure Storage Accounts

- PowerShell Command for removing a storage Account
  Remove-AzureRMStorageAccount
- Command parameters required
  - **Resource Group**
  - **Name**

Creating Azure Virtual Machine

- ✓ Create Resource Group
- ✓ Create Storage Account
- ✓ Create Network Security Group
- ✓ Create Public IP Address
- ✓ Create Virtual Network
- ✓ Create Network Interface
- ✓ Create Virtual Machine Configuration and Virtual Machine

Resource Group

- Storage Account
- Network Security Group
- Public IP Address
- Virtual Network
- Network Interface
- Virtual Machine

Creating Azure Virtual Machines

- Step 1: Create Azure Resource Group

```
New-AzureRmResourceGroup -Name -Location
```

- Step 2: Create Azure Storage Account

```
New-AzureRmStorageAccount -ResourceGroupName -Name -SkuName -Kind
 -Location
```

Creating Azure Virtual Machines

- Step 3: Create Azure Network Security Group

Create network security rule configuration that will be used to create the network security group

```
New-AzureRmNetworkSecurityRuleConfig -Name –Description -Access
-Protocol -SourcePortRange -SourceAddressPrefix
-DestinationPortRange -DestinationAddressPrefix -Direction
-Priority
```

Create network security group using the network security rule configuration

```
New-AzureRmNetworkSecurityGroup -Name -ResourceGroupName
-Location -SecurityRules
```

Creating Azure Virtual Machines

- Step 4: Create Azure Virtual Network

Create virtual network subnet virtual configuration that will be used to create the virtual network

```
New-AzureRmVirtualNetworkSubnetConfig -Name -AddressPrefix
```

Create Azure Virtual network using the virtual network subnet configuration

```
New-AzureRmVirtualNetwork -Name -ResourceGroupName -Location
-AddressPrefix -Subnet
```

Creating Azure Virtual Machines

- Step 6: Create Azure Public Address

```
New-AzureRmPublicIpAddress –Name -ResourceGroupName -Location
-AllocationMethod
```

- Step 7: Create Network Interface Card

```
New-AzureRmNetworkInterface -Name –ResourceGroupName
-Location -SubnetId -PublicIpAddressId
```

Creating Azure Virtual Machines

- Step 8: Get Virtual Machine publisher, Image Offer, Sku and Image

```
Get-AzureRMVMImagePublisher -location
```

```
Get-AzureRMVMImageOffer –location -publisher
```

```
Get-AzureRmVMImageSku –location –publisher -offer
```

```
Get-AzureRmVMImage –location –publisher –offer -sku
```

Creating Azure Virtual Machines

- Step 9: Create an virtual machine configuration file

This command creates a configurable local virtual machine object for Azure. Store the results as a variable.

```
$VM = New-AzureRmVMConfig -VMName -VMSize
```

Other cmdlets can be used to configure a virtual machine object

- Set the operating system properties - **Set-AzureRmVMOperatingSystem**
- Set the platform image for virtual machine - **Set-AzureRmVMSourceImage**
- Add virtual network interface - **Add-AzureRmVMNetworkInterface**
- Set the operating system disk properties - **Add-AzureRmVMNetworkInterface**

Creating Azure Virtual Machines

Example: Create an VM configuration file and configure VM object

```
$VM = New-AzureRmVMConfig –VMName $vmname –VMSize “Standard_A1”
$VM = Set-AzureRmVMOperatingSystem -VM $vm -Windows
-ComputerName $compName -Credential $cred -ProvisionVMAgent
-EnableAutoUpdate
$VM = Set-AzureRmVMSourceImage -VM $vm -PublisherName $publisher
-Offer $offer -Skus $sku -Version $image
$VM = Add-AzureRmVMNetworkInterface -VM $vm -Id $nic.Id
$VM = Set-AzureRmVMOSDisk -VM $vm -Name $diskName –VhdUri
$osDiskUri -CreateOption fromImage
```

Creating Azure Virtual Machines

- Step 10: Create Azure Virtual Machine

This cmdlet creates the virtual machine once all the previous requirements are complete. The –VM parameter accepts the virtual machine configuration that is stored in a variable

```
New-AzureRmVM -ResourceGroupName -Location –VM
```

Example: Creating an Azure Virtual Machine

```
New-AzureRmVM –ResourceGroupName ContosoRG1 –Location “East US”
–VM $VM
```

Creating Azure Virtual Machines with Generalized Images

- Creating a virtual machine from a generalized image is similar to creating a virtual machine using an Azure image
- The generalized image must be uploaded to an Azure storage account
- No requirement to gather the publisher, offer and sku for virtual machine creation when using a generalized image
- The VM configuration file should point to the generalized Image Uri in storage and not to an Azure VM image

Creating Azure Virtual Machines with Generalized Images

- Step 1: Upload the generalized image to Azure Storage

```
Add-AzureRMVhd –ResourceGroup –Destination -LocalFilePath
```

Creating Azure Virtual Machines with Generalized Images

- Step 2: Create Azure Resource Group

```
New-AzureRmResourceGroup -Name -Location
```

- Step 3: Create Azure Storage Account

```
New-AzureRmStorageAccount -ResourceGroupName -Name -SkuName -Kind
 -Location
```

Creating Azure Virtual Machines with Generalized Images

- Step 4: Create Azure Network Security Group

Create network security rule configuration

```
New-AzureRmNetworkSecurityRuleConfig -Name –Description -Access
-Protocol -SourcePortRange -SourceAddressPrefix
-DestinationPortRange -DestinationAddressPrefix -Direction
-Priority
```

Create network security group using the network security rule configuration

```
New-AzureRmNetworkSecurityGroup -Name -ResourceGroupName
-Location -SecurityRules
```

Creating Azure Virtual Machines with Generalized Images

- Step 5: Create Azure Virtual Network

Create virtual network subnet virtual configuration

```
New-AzureRmVirtualNetworkSubnetConfig -Name -AddressPrefix
```

Create Azure Virtual network using the virtual network subnet configuration

```
New-AzureRmVirtualNetwork -Name -ResourceGroupName -Location
-AddressPrefix -Subnet
```

Creating Azure Virtual Machines with Generalized Images

- Step 6: Create Azure Public Address

```
New-AzureRmPublicIpAddress –Name -ResourceGroupName -Location
-AllocationMethod
```

- Step 7: Create Network Interface Card

```
New-AzureRmNetworkInterface -Name –ResourceGroupName
-Location -SubnetId -PublicIpAddressId
```

Creating Azure Virtual Machines with Generalized Images

- Step 8: Create an virtual machine configuration file

```
$VM = New-AzureRmVMConfig -VMName –VMSize
```

Other cmdlets can be used to configure a virtual machine object

- Set the operating system properties - **Set-AzureRmVMOperatingSystem**
- Set the platform image for virtual machine - **Set-AzureRmVMSourceImage**
- Add virtual network interface - **Add-AzureRmVMNetworkInterface**
- Set the operating system disk properties - **Add-AzureRmVMNetworkInterface**

Creating Azure Virtual Machines with Generalized Images

Important: To Configure the OS disk to be created from the existing VHD image, use the **Set-AzureRMVMOSDIsk** cmdlet

The **-CreateOption** parameter should be set to **fromImage** and the **-SourceImageUri** should point to the Uri of the VHD image

Example:

```
$vm = Set-AzureRmVMOSDisk -VM $vm -Name $osDiskName -VhdUri
$osDiskUri -CreateOption fromImage -SourceImageUri $imageURI
-Windows
```

Creating Azure Virtual Machines with Generalized Images

- Step 9: Create Azure Virtual Machine

This cmdlet creates the virtual machine once all the previous requirements are complete. The –VM parameter accepts the virtual machine configuration that is stored in a variable

```
New-AzureRmVM -ResourceGroupName -Location -VM
```

Stopping a Virtual Machine

- PowerShell Command for stopping a virtual machine
  Stop-AzureRMVM

Use the –StayProvisioned optional parameter to keep VM resources provisioned

- Command parameters required

-Resource Group 

-Name 

Starting a Virtual Machine

- PowerShell Command for starting a virtual machine
  `Start-AzureRMVM`
- Command parameters required
  - `Resource Group` 
  - `Name` 

Restarting a Virtual Machine

- PowerShell Command for restarting a virtual machine
  `Restart-AzureRMVM`
- Command parameters required
  - `Resource Group` 
  - `Name` 

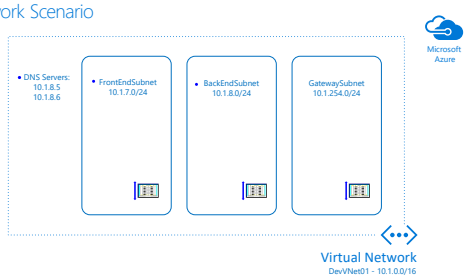
Deleting a Virtual Machine

- PowerShell Command for deleting a virtual machine
  `Remove-AzureRMVM`
- Command parameters required
  - `Resource Group` 
  - `Name` 

Virtual Networks benefits

- Isolation
- Access to the public Internet
- Access to Virtual Machines within the Virtual Network
- Security
- Connectivity

Network Scenario



Choose names and locations

- To choose names for Azure objects follow by Naming Rules and Restrictions article
- Run `Get-AzureRMLocation | Select DisplayName` to get a list of available locations.

Declare variables

- Resource Group name:
  `$RG1 = "TestRG01"`
- Location:
  `$Location = "East US"`
- Virtual Network Name:
  `$VNetName1 = "DevVNet01"`
- Subnets:
  `$SubnetName1 = "FrontEndSubnet"`
  `$SubnetName2 = "BackEndSubnet"`
  `$SubnetName3 = "GatewaySubnet"`

Declare variables

- Virtual Network Prefix:
  `$VNetAddressPrefix = "10.1.0.0/16"`
- Subnet Prefixes:
  `$SubnetAddressPrefix1 = "10.1.7.0/24"`
  `$SubnetAddressPrefix2 = "10.1.8.0/24"`
  `$SubnetAddressPrefix3 = "10.1.254.0/24"`
- DNS Servers:
  `$DnsServer = @("10.1.8.5","10.1.8.6")`

How to create Virtual Networks

- If necessary, create a Resource Group:

```
New-AzureRmResourceGroup -Name $RG1 -Location $Location `
-Tag @{Dept="IT"; Environment="TestDev"}
```

- Create a new VNet:

```
$vnet01 = New-AzureRmVirtualNetwork -ResourceGroupName $RG1 `
-Name $VNetName1 -AddressPrefix $VNetAddressPrefix `
-Location $location -Tag @{Dept="IT"; Environment="TestDev"}
```

How to create Virtual Networks

- Add two subnets to the $vnet01 variable:

```
Add-AzureRmVirtualNetworkSubnetConfig -Name FrontEndSubnet `
-VirtualNetwork $vnet01 -AddressPrefix $SubnetAddressPrefix1

Add-AzureRmVirtualNetworkSubnetConfig -Name BackEndSubnet `
-VirtualNetwork $vnet01 -AddressPrefix $SubnetAddressPrefix2
```

How to create Virtual Networks

- To save the changes to Azure, run:

```
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet01
```

- In an output double check ProvisioningState
  `"ProvisioningState": "Succeeded"`
- To control created VNet:

```
Get-AzureRmVirtualNetwork -ResourceGroupName $RG1 -Name $VNetName1
```

How to modify VNet configuration

- a) `$vnet01 = Get-…` b) `Add-…` c) `Set-…`
- We will change VNet configuration by adding:
  - Another one subnet object – "GatewaySubnet"
  - Array of DNS servers. Must be an array of up to 10 DNS servers, by IP address

How to modify VNet configuration

- Read VNet configuration into a variable $vnet01:

```
$vnet01 = Get-AzureRmVirtualNetwork -ResourceGroupName $RG1 `
-Name $VNetName1
```

- Add a new subnet to the $vnet01 variable:

```
Add-AzureRmVirtualNetworkSubnetConfig -Name $SubnetName3 `
-VirtualNetwork $vnet01 -AddressPrefix $SubnetAddressPrefix3
```

How to modify VNet configuration

- And array of DNS servers to the $vnet01 variable:

```
$vnet01.DhcpOptions.DnsServers = $DnsServer
```

- Save the changes to Azure:

```
Set-AzureRmVirtualNetwork -VirtualNetwork $vnet01
```

How to delete a Virtual Network

- In order to delete Virtual Network use `Remove-AzureRmVirtualNetwork` cmdlet. By default, the cmdlet prompts you for confirmation. To suppress the prompt, use the `Force` parameter:

```
Remove-AzureRmVirtualNetwork -Name $VNetName1 `
-ResourceGroupName $RG1 -Force
```

- Subnet objects are going to be deleted automatically
- Note, you can delete Resource Group with a VNet object in one cmdlet:

```
Remove-AzureRmResourceGroup -Name $RG1 -Force -Verbose
```

Demo: Create a Resource Group & Storage Account

Microsoft

Azure Resource Health

Microsoft Services

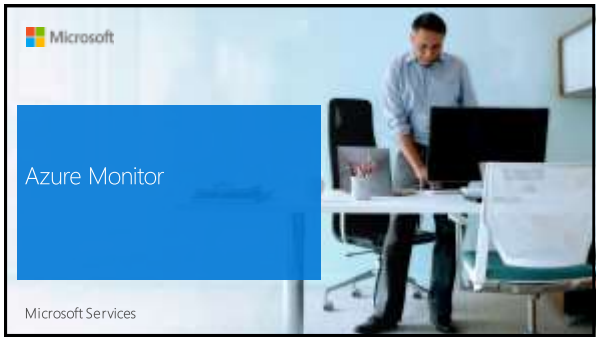
Azure Resource Health

- Azure Resource health helps you diagnose and get support when an Azure issue impacts your resources.
- Provides you with a personalized dashboard of the health of your resources, as opposed to Azure Status which informs you about the global health status of Azure services.
- Shows you all the times your resources were unavailable in the past due to Azure service issues, making it simple for you to understand if an SLA was violated.
- Is a free service.

Azure Resource Health Statuses

- Available: The service has not detected any events impacting the health of the resource.
- Unavailable: The service has detected an ongoing platform or non-platform event impacting the health of the resource.
  - Platform events: These events are triggered by multiple components of the Azure infrastructure.
  - Non-Platform events: These events are triggered by actions taken by users.
- Access up to 14 days of historical health data in the Resource health blade.

Demo: Azure Resource Health



Azure Monitor

- Azure Monitor is a service that provides a single source for monitoring your Azure resources.
- Visualize, query, route, archive, and take action on the metrics and logs coming from resources in Azure.
- Accessed via Azure portal, PowerShell, Cross-Platform CLI or Azure Monitor REST APIs.

Azure Monitor Categories

- Activity log: Describes all operations performed on resources in your subscription e.g. who created or deleted a VM.
- Metrics: Provides a single view of all metrics so you can easily understand how your resources are performing.
- Diagnostic logs: Provide data about the operation of a particular resource e.g. NSG Rule Counters.

Demo: Azure Monitor

Microsoft

Lab: Automating VM Deployment with PowerShell

Microsoft Services

Microsoft