**Managing Azure with PowerShell and Azure CLI.**

* PowerShell
  + Installing the required modules
  + Login to Azure Portal
  + Managing Resource Group
  + Managing App Service Plans and App Service Web Apps
  + Create and Configure a Storage Account
  + Managing Storage Accounts using PowerShell
* Azure CLI
  + Introduction and Setup
  + Creating Resources
  + Listing and Formatting output
  + Connect a Web App to SQL Database
  + Deleting Resources
  + Interactive Mode

Diagram

Description automatically generated

**Installing PowerShell Modules for Azure**

**Introduction:**

* **Azure PowerShell** is a set of **modules** that provide **cmdlets** to manage Azure with **PowerShell**. You can use the cmdlets to create, test, deploy, and manage solutions and services delivered through the Azure platform.
* Since **PowerShell 6.x+** can run on **Linux**, **macOS**, and **Windows**, Azure PowerShell is now available for all platforms.
* Typically, as a developer, you might want to automate some management tasks by creating reusable scripts, or combine management of Azure resources with management of other network and infrastructure services.
* Starting in December 2018, the **Azure PowerShell Az** module is in general release and now the intended PowerShell module for interacting with Azure.
* Az offers **shorter commands, improved stability, and cross-platform support**.
* Az uses the **.NET Standard library**, which means it runs on PowerShell 5.x and PowerShell 6.x.

Verify Version:

$PSVersionTable.PSVersion

1. Windows Search 🡪 **Powershell 7.X**

If not found, Install Powershell 7 (<https://github.com/PowerShell/PowerShell/releases/download/v7.3.6/PowerShell-7.3.6-win-x64.msi>)

1. **Install the Azure Resource Manager (ARM) modules from the PowerShell Gallery**

In **administrator** mode, Open Standard Windows PowerShell console,

Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser

Install-Module -Name Az -Repository PSGallery -Force

**If already existing, use following command to update**

Update-Module -Name Az

**Get help for cmdlets**

To get detailed help for any cmdlet that you see in this tutorial, use the Get-Help cmdlet.

**Get-Help** <cmdlet-name> -Detailed

For example, to get help for the Get-AzureRmResource cmdlet, type:

**Get-Help** Get-AzResource -Detailed

**Formatting Output:**

Get-Command -Verb Format | Format-Wide

Get-Process | Format-Table

Get-Process | Format-List

**Managing Azure Accounts and Subscriptions**

**Sign in to Azure**

**Connect-AzAccount -Tenant** 82d8af3b-d3f9-465c-b724-0fb186cc28c7

# To view all subscriptions for your account

**Get-AzSubscription | Format-List**

**Get-AzSubscription | Format-Table**

**Get-AzSubscription | Format-Wide**

# To select a default subscription for your current session

**Select-AzSubscription** -Subscription "<Your Subscription Name/ID>"

Or

**Set-AzContext** -Subscription "<Your Subscription Name/ID>"

# View your current Azure PowerShell session context

# This session state is only applicable to the current session and will not affect other sessions

**Get-AzContext**

**Get-AzSubscription**

**Managing a Resource Group**

1. In the Windows PowerShell ISE, execute the following lines:

$location = "East US"

$rgName = "Powershell-Demo-rg"

**New-AzResourceGroup** -Name $rgName -Location $location

**Delete a resource group**

Remove-AzResourceGroup -Name $rgName

**Get the list of all resource groups**

Get-AzResourceGroup

**List all resources in Resource group**

Get-AzResource -ResourceGroupName "Demo-rg" | Format-table

**List only virtual machines in a specific resource group**

Get-AzResource -ResourceGroupName Demo-rg -ResourceType "Microsoft.Compute/virtualMachines" | Format-table

**List resources in a specific resource group located in a specific Azure region**

Get-AzResource -ResourceGroupName $resourceGroupName -Location "East US"

**Get a reference to a particular resource in a resource group**

$resourceStorage=Get-AzResource -ResourceName dsdemostorage

$resourceStorage | Select-Object ResourceType

**Creating a VM using PowerShell**

This example creates two new subnet configurations using the New-AzVirtualSubnetConfig cmdlet, and then uses them to create a virtual network. The New-AzVirtualSubnetConfig template only creates an in-memory representation of the subnet. In this example, the frontendSubnet has CIDR 10.0.1.0/24 and references a network security group that allows RDP access. The backendSubnet has CIDR 10.0.2.0/24 and references the same network security group.

# Set variables

$location = "East US"

$rgName = "Powershell-Demo-rg"

$vnetName = "My-Demo-vnet"

$frontendSubnet = "Frontend-subnet"

$backendSubnet = "Backend-subnet"

$vnetAddressSpace = "10.0.0.0/16"

$frontendSubnetAddressPrefix = "10.0.1.0/24"

$backendSubnetAddressPrefix = "10.0.2.0/24"

$frontEndNSG = "Frontend-nsg"

$backEndNSG = "Backend-nsg"

# Create a resource group

**New-AzResourceGroup** -Name $rgName -Location $location

#Create NSG RDP Rule

$rdpRule = **New-AzNetworkSecurityRuleConfig** -Name rdp-rule -Description "Allow RDP" -Access Allow -Protocol Tcp -Direction Inbound -Priority **1000**  -SourceAddressPrefix Internet -SourcePortRange \* -DestinationAddressPrefix \* -DestinationPortRange **3389**

# Create NSG

$frontendNetworkSecurityGroup = **New**-**AzNetworkSecurityGroup** -ResourceGroupName $rgName -Location $location -Name $frontEndNSG -SecurityRules $rdpRule

$backendNetworkSecurityGroup = **New**-**AzNetworkSecurityGroup** -ResourceGroupName $rgName -Location $location -Name $backEndNSG -SecurityRules $rdpRule

# Create subnets

$frontendSubnet = **New-AzVirtualNetworkSubnetConfig** -Name $frontendSubnet -AddressPrefix $frontendSubnetAddressPrefix -NetworkSecurityGroup $frontendNetworkSecurityGroup

$backendSubnet = **New-AzVirtualNetworkSubnetConfig** -Name $backendSubnet -AddressPrefix $backendSubnetAddressPrefix -NetworkSecurityGroup $backendNetworkSecurityGroup

# Create a virtual network

$vNet = **New-AzVirtualNetwork** -Name $vNetName -ResourceGroupName $rgName  -Location $location -AddressPrefix $vnetAddressSpace -Subnet $frontendSubnet,$backendSubnet

**Use the following commands to create Network Interface Card along with Public IP Address**

$NICName="WebVM1-nic"

$publicIpName = "WebVM1-ip"

$dnsPrefix = "dss-webvm1-mar" # Must be Unique

# Create a public IP address object that can be assigned to the NIC (Network Interface Card)

$publicIP = **New-AzPublicIpAddress** -Name $publicIpName -ResourceGroupName $rgName -AllocationMethod Static -DomainNameLabel $dnsPrefix -Location $location

#(change DomainNameLabel when you practice)

#Create IPConfig for NIC

$IPconfig = **New-AzNetworkInterfaceIpConfig** -Name "IPConfig1" -PrivateIpAddressVersion IPv4 -PrivateIpAddress "10.0.1.10" -SubnetId $vNet.Subnets[0].Id -PublicIpAddressId $publicIP.Id

#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 -IpConfiguration $IPconfig

**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.

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.

$vmName = "Web1-vm"

$vmSize = "Standard\_B2s"

# Create the virtual machine configuration object and save a reference to it

$vmConfig = **New-AzVMConfig** -VMName $vmName -VMSize $vmSize

# 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"

$SecurePassword = **ConvertTo-SecureString** $SecurePassword -AsPlainText -Force

$cred = New-Object System.Management.Automation.**PSCredential** ($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 2022 R2 Datacenter image is specified in the configuration information.

$pubName = "MicrosoftWindowsServer"

$offerName = "WindowsServer"

$skuName = "2022-Datacenter"

# Assign the gallery image to the VM configuration

$vmConfig = **Set-AzVMSourceImage** -VM $vmConfig -PublisherName $pubName -Offer $offerName -Skus $skuName -Version "latest"

# 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 $rgName -Name $vmName

**Azure CLI**

* With Azure CLI, you can create, manage, and delete services on the command line via cmd.exe, bash or {your shell} on the operating system of your choice.
* You can use it in your browser with Azure Cloud Shell, or you can install it on macOS, Linux, and Windows and run it from the command line.
* Azure CLI 2.0 is optimized for managing and administering Azure resources from the command line, and for building automation scripts that work against the Azure Resource Manager.

**Install Azure CLI on Windows:**

1. **MSI Installer**: <https://learn.microsoft.com/en-us/cli/azure/install-azure-cli-windows?tabs=azure-cli#install-or-update>
2. **Powershell Installer:** Open PowerShell as Administrator and execute below command

$ProgressPreference = 'SilentlyContinue'; Invoke-WebRequest -Uri https://aka.ms/installazurecliwindows -OutFile .\AzureCLI.msi; Start-Process msiexec.exe -Wait -ArgumentList '/I AzureCLI.msi /quiet'; Remove-Item .\AzureCLI.msi

**Installing Azure CLI on Ubuntu:**

curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash

**To Set Tenant and Subscription in Current Context**

az login

az login --tenant <tentant id>

az account list

az account set --subscription <subscription id>

az account show

**To search for commands, use** [**az find**](https://docs.microsoft.com/cli/azure/reference-index#az-find)**.**

az find "az network"

**Use the --help argument to get a complete list of commands and subgroups of a group.**

az network nsg **--help**

**Creating Resources**

**#Create a Resource Group:**

az group create -n DemoRG -l southindia

Following are some popular Azure resource types and the corresponding Azure CLI create commands to create them:

|  |  |
| --- | --- |
| **Resource Type** | **Azure CLI create command** |
| Resource Group | az group list |
| Virtual Machine | az vm create |
| Virtual Network | az network vnet create |
| Load Balancer | az network lb create |
| Managed Disk | az disk create |
| Storage account | az storage account create |
| Virtual Machine Scale Set | az vmss create |
| Azure Container | az acs create |
| Web App | az webapp create |
| SQL Database Server | az sql server create |
| Document DB | az documentdb create |

**Note:**

If you do not need to wait on creation of a resource before continuing, you can use the **no-wait** option to start a create action in the background.

az webapp create -n DssMyWebApp2 -g DemoRG --plan MyAppServicePlan **--no-wait**

**Listing resources and formatting output**

To list all the items for a particular category:

az group **list**

**Output Formats:**

|  |  |
| --- | --- |
| **--output** | **Description** |
| json | json string. json is the default. |
| jsonc | colored json string. |
| table | table with column headings. |
| tsv | tab-separated values. |

**Example:**

* az group list --**output** **json**
* az group list --**output** **table**
* az group list --**output** **tsv**

**Query:**

az group list --query "[\*][name, location]" -o table

az group list --query "[].{**Name**:name, **Location**:location}" -o table

**Interactive Mode**

You can use Azure CLI 2.0 in interactive mode by running the az **interactive** command.

az **interactive**

Note: Interactive mode optionally displays command descriptions, parameter descriptions, and command examples. You can turn descriptions and examples on or off using F1.

You can turn the display of parameter defaults on or off using F2.

**Create Azure VM using Azure CLI**

**# Set variables**

rgName="AzureCli-rg"

location="eastus"

vmName="Demo-vm"

vnetName="Demo-vnet"

frontEndSubnetName="frontEnd-subnet"

backEndSubnetName="backEnd-subnet"

nicName="Demo-vm-nic"

publicIpName="myPublicIP"

nsgName="myNSG"

**# Create a resource group**

az group create --name $rgName --location $location

**# Create a virtual network with two subnets**

az network vnet create --name $vnetName --resource-group $rgName --location $location --address-prefixes "10.0.0.0/16" --subnet-name $frontEndSubnetName --subnet-prefix "10.0.1.0/24" --subnet-name $backEndSubnetName --subnet-prefix "10.0.2.0/24"

**# Create a public IP address**

az network public-ip create --name $publicIpName --resource-group $rgName --location $location --allocation-method Dynamic

**# Create a network security group**

az network nsg create --name $nsgName --resource-group $rgName --location $location

**# Create a network interface with two IP configurations (one for each subnet)**

az network nic create --name $nicName --resource-group $rgName --location $location --subnet $frontEndSubnetName --vnet-name $vnetName --public-ip-address $publicIpName --network-security-group $nsgName --subnet $backEndSubnetName --private-ip-address "10.0.2.4"

**# Create a virtual machine**

az vm create --name $vmName --resource-group $rgName --location $location --nics $nicName --image UbuntuLTS --admin-username azureuser --generate-ssh-keys