Azure PowerShell

Objectives

In this hands-on lab, you will learn how to:

- Create a virtual machine with PowerShell
- Create a web app with PowerShell
- Create an Azure Database with PowerShell

Prerequisites

The following are required to complete this hands-on lab:

- An active Microsoft Azure subscription
- Exercise 1 & 2: Web Deploy - https://docs.microsoft.com/en-us/iis/install/installing-publishing-technologies/installing-and-configuring-web-deploy-on-iis-80-or-later
- SQL Server Management Tools

Exercises

This hands-on lab includes the following exercises:

- Exercise 1: Create a virtual machine
- Exercise 2: Create a web app
- Exercise 3: Create an Azure SQL database

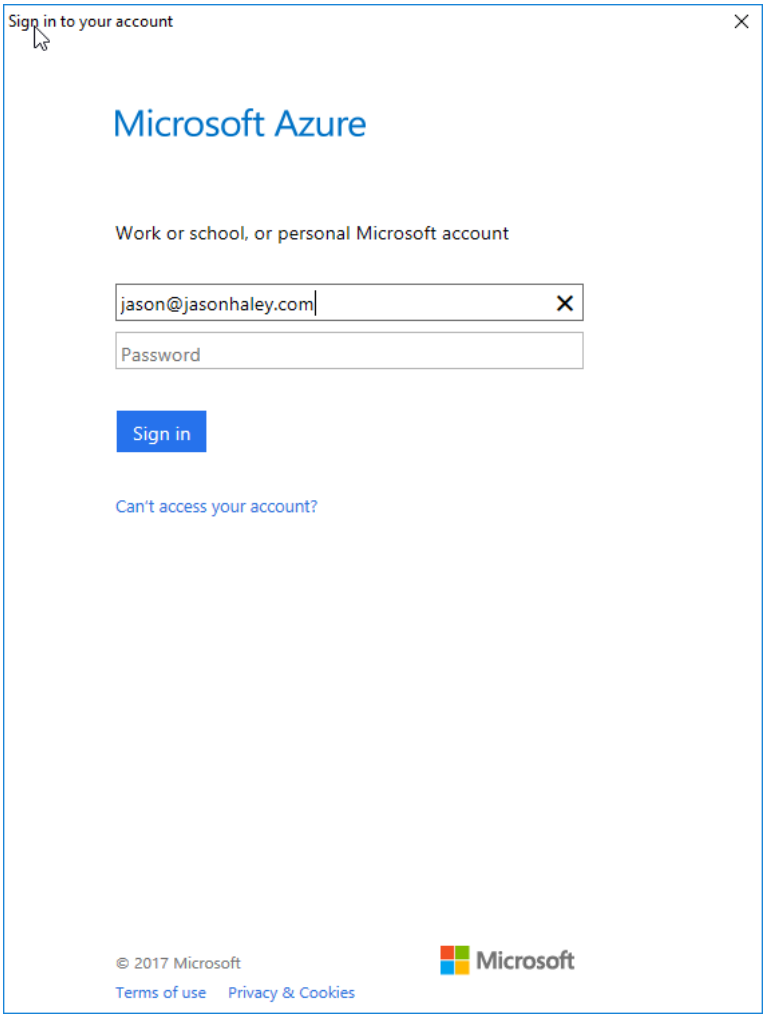
Estimated time to complete this lab: 30 - 45 minutes

Exercise 1: Create a virtual machine

In this exercise, you will create a virtual machine and all the related resources.

1. Open the Windows PowerShell ISE or PowerShell console
2. Log in to your Azure subscription with the Login-AzureRmAccount command and use the login dialogs that show

```
#login to your azure account
Login-AzureRmAccount
```



3. Setup your variables

```
#variable for VM creation
$id = [Guid]::NewGuid().ToString("n").SubString(0,8)
$resourceGroupName = $id + "-rg"
$location = "eastus"
$vmName = $id + "-vm1"
$subnetName = "web"
$vnetName = $id + "-vnet"
$pipName = $id + "-pip"
$rdpRuleName = $id + "-rdp-allow"
$httpRuleName = $id + "-http-allow"
$webDeployRuleName = $id + "-webdeploy-allow"
$nsgName = $id + "-nsg"
$nicName = $id + "-nic"
```

4. Create the resource group to connect everything

```
#create a new resource group to use
New-AzureRmResourceGroup -Name $resourceGroupName -Location $location
```

5. Create a virtual network, subnet and public IP address

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

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

```
-Location $location -Name $vnetName -AddressPrefix 192.168.0.0/16 `
-Subnet $subnetConfig

# Create a public IP address and specify a DNS name
$pip = New-AzureRmPublicIpAddress -ResourceGroupName $resourceGroupName `
-Location $location -AllocationMethod Static -IdleTimeoutInMinutes 4 -Name $pipName
```

6. Create a network security group and rules for RDP, HTTP and WebDeploy

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

# Create an inbound network security group rule for port 80 for HTTP
$nsgRuleHttp = New-AzureRmNetworkSecurityRuleConfig -Name $httpRuleName `
-Protocol Tcp -Direction Inbound -Priority 1010 -SourceAddressPrefix * `
-SourcePortRange * -DestinationAddressPrefix * -DestinationPortRange 80 `
-Access Allow

# Create an inbound network security group rule for port 8172 for WebDeploy
$nsgRuleWebDeploy = New-AzureRmNetworkSecurityRuleConfig -Name $webDeployRuleName `
-Protocol Tcp -Direction Inbound -Priority 1020 -SourceAddressPrefix * `
-SourcePortRange * -DestinationAddressPrefix * -DestinationPortRange 8172 `
-Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $resourceGroupName `
-Location eastus -Name $nsgName `
-SecurityRules $nsgRuleRDP,$nsgRuleHttp,$nsgRuleWebDeploy
```

7. Create a network card for the virtual machine

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

8. Capture the username and password for the virtual machine

```
# Define a credential object
$cred = Get-Credential
```

9. Create the VM config and the virtual machine (this will take around 5 minutes)

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

New-AzureRmVM -ResourceGroupName $resourceGroupName -Location $location -VM $vmConfig
```

10. Configure the DSC extension to add IIS and WebDeploy (this will take around 10 minutes)

```
# Install IIS and WebDeploy
$PublicSettings =
'{"ModulesURL":"https://github.com/JasonHaley/GlobalAzureBootcamp/raw/master/ConfigureWebServer.ps1
.zip", "configurationFunction": "ConfigureWebServer.ps1\\Main", "Properties": {"nodeName": "' +
$vmName + '"} }'

Set-AzureRmVMExtension -ExtensionName "DSC" -ResourceGroupName $resourceGroupName `
-VMName $vmName -Publisher "Microsoft.Powershell" -ExtensionType "DSC" `
-TypeHandlerVersion 2.24 -SettingString $PublicSettings -Location $location
```

11. Get the public IP of your new virtual machine

```
$ipaddress = Get-AzureRmPublicIpAddress -ResourceGroupName $resourceGroupName `
| Select -ExpandProperty IpAddress
```

12. Set your variables for your Sample Web app Visual Studio solution

```
# set variable for the location of your site package
$packagePath = "<your location of lab files>\WebApp.zip"
$user = $vmName + "\" + $cred.UserName
$password = $cred.GetNetworkCredential().Password

# requires Web Deploy to be installed!
$msdeploy = "C:\Program Files (x86)\IIS\Microsoft Web Deploy V3\msdeploy.exe"
$computerName = "https://" + $ipaddress + ":8172/msdeploy.axd?site=Default%20Web%20Site"

$nameParam = "IIS Web Application Name"
$nameValue = "Default Web Site"
$setParam = '-setParam:name="IIS', 'Web', 'Application', 'Name",value="Default', 'Web', 'Site"'
```

13. Run WebDeploy to deploy the sample package site to the virtual machine

```
& $msdeploy -source:package=$packagePath -verb=sync -
dest:auto,computerName=$computerName,userName=$user,password=$password,authType=Basic -
allowUntrusted=true $setParam
```

You should now be able to go to the public IP address of the new VM and see the deployed web application.

14. Remove all resources and resource group

```
# remove all resources in resource group
Remove-AzureRmResourceGroup -ResourceGroupName $resourceGroupName
```

Exercise 2: Create a web app

In this exercise, you will create a web app and its related resources

1. Open the Windows PowerShell ISE or PowerShell console
2. Log in to your Azure subscription with the Login-AzureRmAccount command and use the login dialogs that show

```
#login to your azure account
Login-AzureRmAccount
```

3. Setup your variables for the web app creation

```
#variable for WebApp creation
$id = [Guid]::NewGuid().ToString("n").SubString(0,8)
$resourceGroupName = $id + "-rg"
$location = "eastus"
$webAppName = $id + "-web"
$appServicePlanName = $id + "-plan"
```

4. Create a Resource Group

```
New-AzureRmResourceGroup -Name $resourceGroupName -Location $location
```

5. Create an App Service for the web application

```
# Create an App Service plan in Standard tier.
New-AzureRmAppServicePlan -Name $appServicePlanName -Location $location `
-ResourceGroupName $resourceGroupName -Tier Standard
```

6. Create a Web App

```
# Create a web app.
New-AzureRmWebApp -Name $webAppName -Location $location `
-AppServicePlan $appServicePlanName -ResourceGroupName $resourceGroupName
```

7. Download the publishing profile and grab the username and password out of it

```
# Get publishing profile for the web app
$xml = [xml](Get-AzureRmWebAppPublishingProfile -Name $webAppName `
-ResourceGroupName $resourceGroupName `
-OutputFile null)

# Extract connection information from publishing profile
$username = $xml.SelectNodes("//publishProfile[@publishMethod=`"MSDeploy`"]/@userName").value
$password = $xml.SelectNodes("//publishProfile[@publishMethod=`"MSDeploy`"]/@userPWD").value
$publishUrl = $xml.SelectNodes("//publishProfile[@publishMethod=`"MSDeploy`"]/@publishUrl").value
```

8. Set your variables for your Sample Web app Visual Studio solution

```
# requires Web Deploy to be installed!
$packagePath = "C:\Users\Jason\Desktop\PSInAction\WebApp.zip"
$msdeploy = "C:\Program Files (x86)\IIS\Microsoft Web Deploy V3\msdeploy.exe"
$computerName = "https://" + $publishUrl + "/msdeploy.axd?site=" $webAppName

$setParam = '-setParam:name="IIS', 'Web', 'Application', 'Name"'
$setParamValue += ',value="' + $webAppName + '"'
$setParamAndValue = "$setParam${setParamValue}"
```

9. Run WebDeploy to deploy the sample package site to the virtual machine

```
    & $msdeploy -source:package=$packagePath -verb=sync -
    dest:auto,computerName=$computerName,userName=$username,password=$password,authType=Basic
    $setParamAndValue -verbose
```

You should now be able to go to the web app and see the deployed web application.

10. Remove all resources and resource group

```
# remove all resources in resource group
Remove-AzureRmResourceGroup -ResourceGroupName $resourceGroupName
```

Exercise 3: Create an Azure SQL database

In this exercise, you will create an Azure SQL database and its related resources

11. Open the Windows PowerShell ISE or PowerShell console
12. Log in to your Azure subscription with the Login-AzureRmAccount command and use the login dialogs that show

```
#login to your azure account
Login-AzureRmAccount
```

13. Setup your variables for the db creation

```
#variable for Db creation
$id = [Guid]::NewGuid().ToString("n").SubString(0,8)
$resourceGroupName = $id + "-rg"
$location = "eastus"
$serverName = "dbserver" + $id
$databaseName = "db" + $id
$adminlogin = "ServerAdmin"
$password = "ChangeYourAdminPassword1"
```

14. Get the current public IP address of your local machine to create a firewall rull

```
#Get your client ip
$externalIp = Invoke-WebRequest ifconfig.me/ip | Select -ExpandProperty Content
$externalIp = $externalIp -replace "`t|`n|`r",""
$externalIp =  $externalIp -replace  " ;|; ",";"
```

15. Create the resource group

```
#create a new resource group to use
New-AzureRmResourceGroup -Name $resourceGroupName -Location $location
```

16. Create the server

```
# Create the server
New-AzureRmSqlServer -ResourceGroupName $resourceGroupName `
    -ServerName $serverName `
    -Location $location `
    -SqlAdministratorCredentials $(New-Object -TypeName System.Management.Automation.PSCredential -
ArgumentList $adminlogin, $(ConvertTo-SecureString -String $password -AsPlainText -Force))
```

17. Create a firewall rule

```
# Add a firewall rule
New-AzureRmSqlServerFirewallRule -ResourceGroupName $resourceGroupName `
    -ServerName $serverName `
    -FirewallRuleName "AllowSome" -StartIpAddress $externalIp -EndIpAddress $externalIp
```

18. Create a new database

```
# Create a new database
New-AzureRmSqlDatabase  -ResourceGroupName $resourceGroupName `
    -ServerName $serverName `
    -DatabaseName $databaseName `
    -RequestedServiceObjectiveName "S0"
```

19. Add a new login to the server and database for using with the application

```
# Use PowerShell to create a login for the server
$serverConnection = new-object Microsoft.SqlServer.Management.Common.ServerConnection
$serverConnection.ServerInstance=$serverName + '.database.windows.net'
$serverConnection.LoginSecure = $false
$serverConnection.Login = $adminlogin
$serverConnection.Password = $password
```

```
# get the serer object
[System.Reflection.Assembly]::LoadWithPartialName('Microsoft.SqlServer.SMO') | Out-Null
$SqlServer = New-Object 'Microsoft.SqlServer.Management.Smo.Server' ($serverName +
'.database.windows.net')


# get all of the current logins and their types
$SqlServer.Logins | Select-Object Name, LoginType, Parent

# create a new login by prompting for new credentials
$NewLoginCredentials = Get-Credential -Message "Enter credentials for the new login"
$NewLogin = New-Object Microsoft.SqlServer.Management.Smo.Login($SqlServer,
$NewLoginCredentials.UserName)
$NewLogin.LoginType = [Microsoft.SqlServer.Management.Smo.LoginType]::SqlLogin
$NewLogin.Create($NewLoginCredentials.Password)

# create a new database user for the newly created login
$NewUser = New-Object Microsoft.SqlServer.Management.Smo.User($SqlServer.Databases[$databaseName],
$NewLoginCredentials.UserName)
$NewUser.Login = $NewLoginCredentials.UserName
$NewUser.Create()
$NewUser.AddToRole("db_datareader")
$NewUser.AddToRole("db_datawriter")
$NewUser.AddToRole("db_ddladmin")
```

20. You should now be able to connect to the new SQL Database with either SSMS or Visual Studio

21. Remove all resources and resource group

```
# remove all resources in resource group
Remove-AzureRmResourceGroup -ResourceGroupName $resourceGroupName
```