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
- Visual Studio 2017 Community or greater (Professional or Enterprise)

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

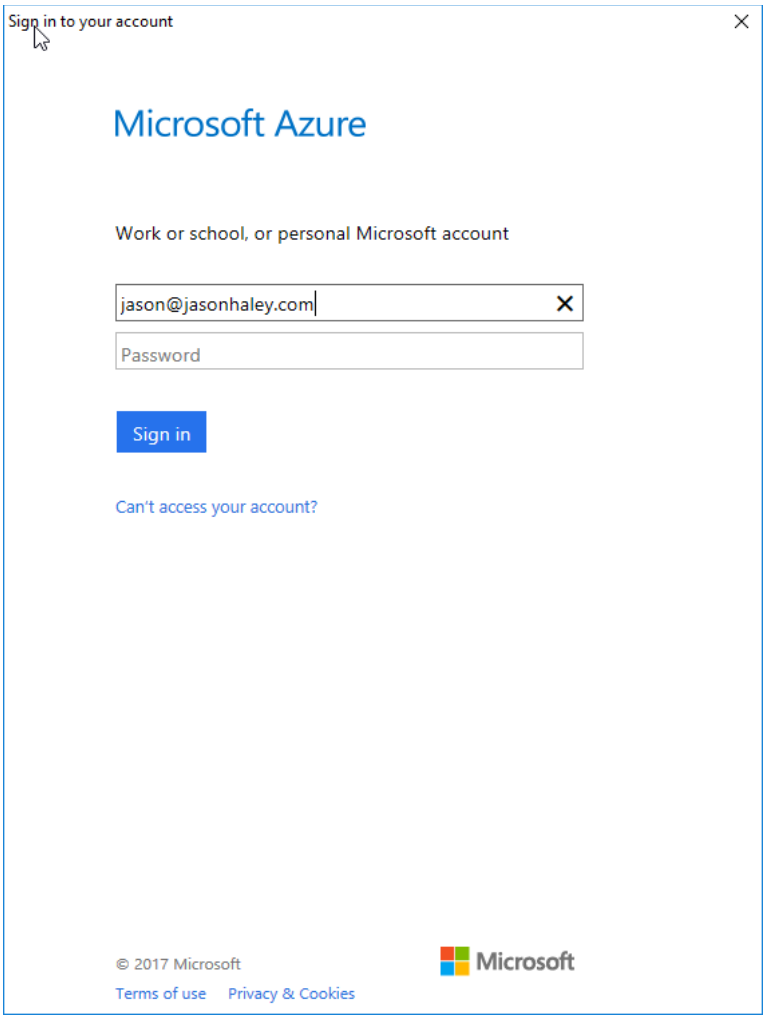
Estimated time to complete this lab: 30 - 45 minutes

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 – update the $firstname variable to be yours

```
#variable for VM creation
$firstname = "jason"
$resourceGroup = "GABPSLab"
$vmName = "GABPSLabVM" + $firstname
$location = "eastus"
```

4. Create the resource group to connect everything

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

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

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

# Create a virtual network
$vnet = New-AzureRmVirtualNetwork -ResourceGroupName $resourceGroup `
-Location $location -Name GABVMVNet -AddressPrefix 192.168.0.0/16 `
-Subnet $subnetConfig
```

```
# Create a public IP address and specify a DNS name
$pip = New-AzureRmPublicIpAddress -ResourceGroupName $resourceGroup `
-Location $location -AllocationMethod Static -IdleTimeoutInMinutes 4 `
-Name "GABPSLabIP" + $firstname
```

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 GABVMLabNetworkSecurityGroupRuleRDP  -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
$nsgRuleWeb = New-AzureRmNetworkSecurityRuleConfig `
-Name GABVMLabNetworkSecurityGroupRuleWWW  -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 GABVMLabNetworkSecurityGroupRuleWebDeploy  -Protocol Tcp `
-Direction Inbound -Priority 1020 -SourceAddressPrefix * -SourcePortRange * `
-DestinationAddressPrefix * -DestinationPortRange 8172 -Access Allow

# Create a network security group
$nsg = New-AzureRmNetworkSecurityGroup -ResourceGroupName $resourceGroup `
-Location eastus -Name GABVMLabNSG -SecurityRules  `
$nsgRuleRDP,$nsgRuleWeb,$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 GABPSNic -ResourceGroupName $resourceGroup `
-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

```
# 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 $resourceGroup -Location $location -VM $vmConfig
```

10. Configure the DSC extension to add IIS and WebDeploy.

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

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

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

```
# set variable for the location of your Visual Studio Solution file
$solutionPath = "C:\_junk\GABWebApp\GABWebApp.sln"
$solutionName = "WebApplication2.sln"
$user = $vmName + "\" + $cred.UserName
$password = $cred.Password
$msbuild = "C:\Windows\Microsoft.NET\Framework64\v4.0.30319\MSBuild.exe"
```

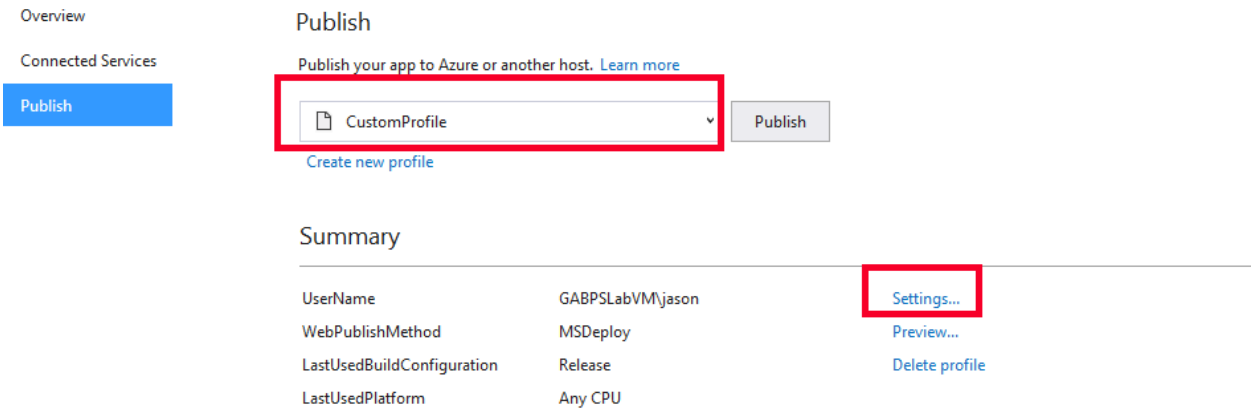
12. Get the public IP of your new virtual machine

```
Get-AzureRmPublicIpAddress -ResourceGroupName $resourceGroup | Select `
-ExpandProperty IpAddress
```

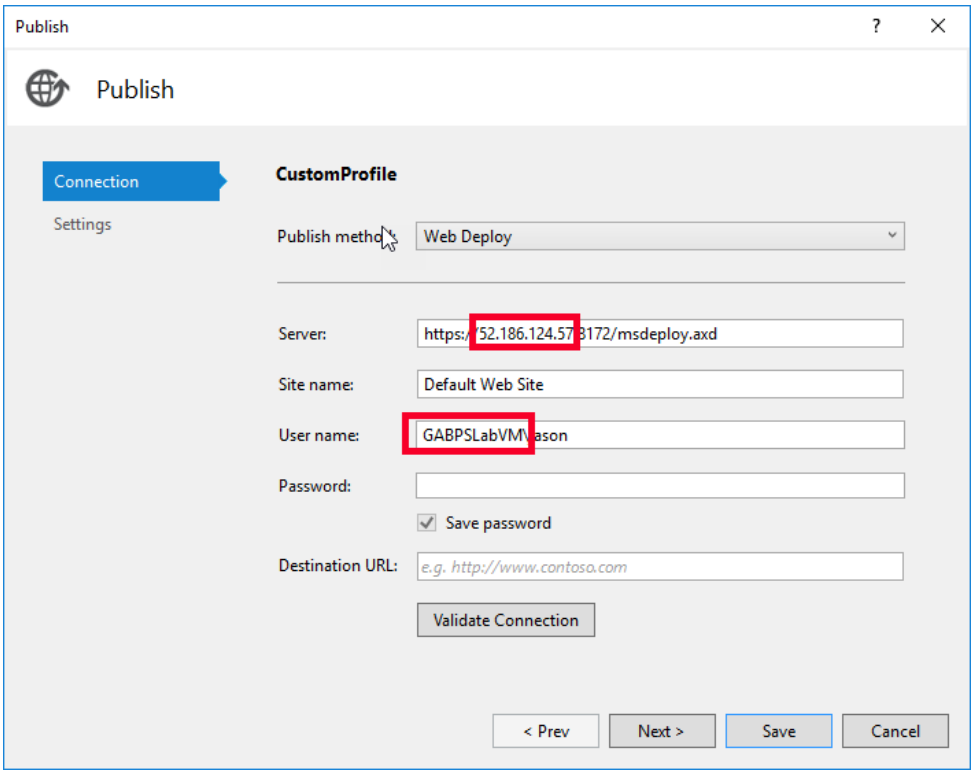
Modify the publish profile CustomProfile to point at your new VM's public IP.

13. In Visual Studio, in the **Solution Explorer**, **right click on the web project** and select **Publish**

This will show the publish settings



14. Make sure the **CustomProfile** is selected (this was created in the virtual machine hands on lab earlier) and select the **Settings…** link.



15. Change the Server ip address to match your new virtual machine and the username domain to be the name of your new virtual machine.
16. Back in your PowerShell ISE or PowerShell Console, use msbuild to build and deploy your web application to the virtual machine

```
# build and publish to site
& $msbuild $solutionPath /p:DeployOnBuild=true /p:PublishProfile=CustomProfile
/p:AllowUntrustedCertificate=True /p:User=$user /p:Password=$password
```

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

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 – update the $firstname to be yours.

```
#variable for WebApp creation
$firstname = "jason"
$resourceGroup = "GABPSLab"
$location = "eastus"
```

```
$webappname="gabpswebapp" + $firstname
```

4. Create an App Service for the web application

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

5. Create a Web App

```
# Create a web app.
New-AzureRmWebApp -Name $webappname -Location $location `
-AppServicePlan $webappname -ResourceGroupName $resourceGroup
```

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

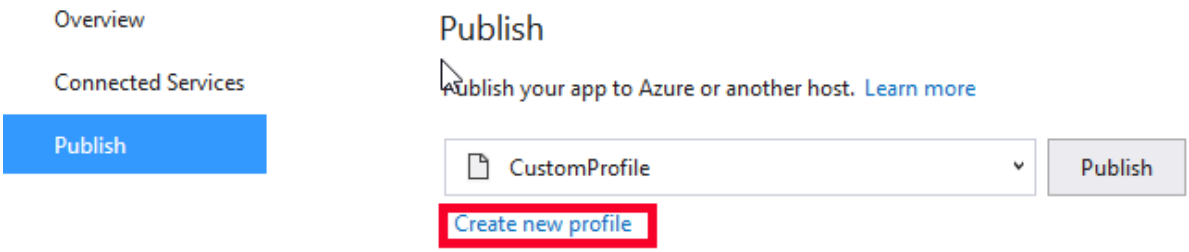
```
# Get publishing profile for the web app
$xml = [xml](Get-AzureRmWebAppPublishingProfile -Name $webappname `
-ResourceGroupName $resourceGroup `
-OutputFile null)

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

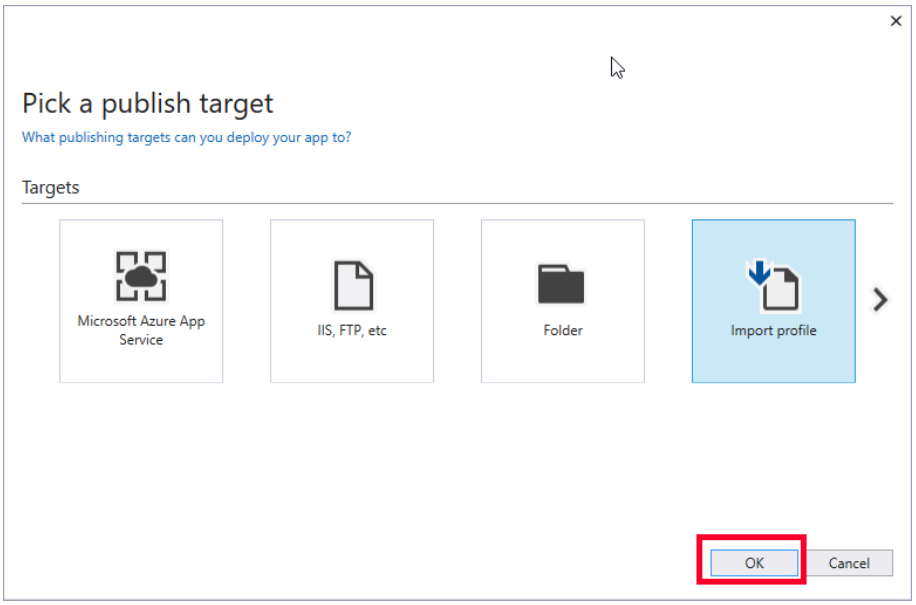
In Visual Studio, you also need to import the new publish profile in order to use the msbuild to deploy.

7. In the Azure Portal, find the new Web application created in step 4 above and download the publish profile for it
8. In Visual Studio, in the **Solution Explorer**, **right click on the web project** and select **Publish**

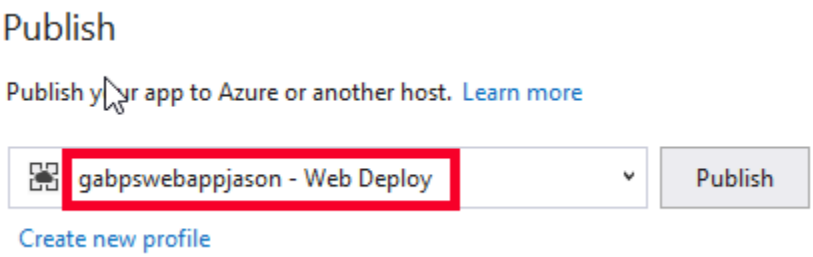
This will show the publish settings



9. Click the Create new profile link
10. Select the Import profile button and click OK



11. Find your Publish Settings file in the downloads folder and click Open



12. Make sure the name of the profile matches the expected name in $profileName variable
13. Back in your PowerShell ISE or PowerShell Console, use msbuild to deploy the application to the Web App

```
# Use msbuild to build and deploy the solution to the web app
& $msbuild $solutionPath /p:DeployOnBuild=true /p:AllowUntrustedCertificate=True
/p:PublishProfile=$profileName /p:User=$username /p:Password=$password
```

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

Exercise 3: Create an Azure SQL database

In this exercise, you will create an Azure SQL database 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 db creation – update the $firstname to be yours.

```
#variable for Db creation
$firstname = "jason"
$resourceGroup = "GABPSLab"
$location = "eastus"
$servername = "gabdbpsserver" + $firstname
$databasename = "gabdbps" + $firstname
$adminlogin = "ServerAdmin"
$password = "ChangeYourAdminPassword1"
```

4. 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  " ;|; ",";"
```

5. Create the server

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

6. Create a firewall rule

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

7. Create a new database

```
# Create a new database
New-AzureRmSqlDatabase  -ResourceGroupName $resourceGroup `
-ServerName $servername `
-DatabaseName $databasename `
-RequestedServiceObjectiveName "S0"
```

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

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

[System.Reflection.Assembly]::LoadWithPartialName('Microsoft.SqlServer.SMO') | Out-
Null
$SqlServer = New-Object Microsoft.SqlServer.Management.Smo.Server($mySrvConn)

# 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")
```

9. In Visual Studio, in your Solution Explorer, locate the web.config file and open it
10. Find the <connectionStrings> element and change the connectionString content to use the new database server, database and user you just created.
11. Run your web application locally and you should now be connecting to the Azure SQL database you created using PowerShell

Exercise 4: Delete all resources in the Resource Group

In this exercise, you will remove all the resources created in this lab.

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 removing the resource group.

```
#variable for resource group
$resourceGroup = "GABPSLab"
```

4. Remove all resources and resource group

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