**Exercise - Create an Azure Resource using scripts in Azure PowerShell**

* 15 minutes

This module requires a sandbox to complete. A [**sandbox**](https://docs.microsoft.com/en-us/learn/support/faq?pivots=sandbox) gives you access to Azure resources. Your Azure subscription will not be charged. The sandbox may only be used to complete training on Microsoft Learn. Use for any other reason is prohibited, and may result in permanent loss of access to the sandbox.

[Sign in to activate sandbox](https://docs.microsoft.com/en-us/learn/modules/automate-azure-tasks-with-powershell/6-exercise-create-resource-interactively?activate-azure-sandbox=true)

Recall our original scenario - creating VMs to test our CRM software. When a new build is available, we want to spin up a new VM so we can test the full install experience from a clean image. Then when we are finished, we want to delete the VM.

Let's try the commands you would use to create a VM.

**Create a Linux VM with Azure PowerShell**

Since we are using the Azure sandbox, you won't have to create a Resource Group. Instead, use the Resource Group **[sandbox resource group name]**. In addition, be aware of the location restrictions.

Let's create a new Azure VM with PowerShell.

1. Use the New-AzVm cmdlet to create a VM.
   * Use the Resource Group **[sandbox resource group name]**.
   * Give the VM a name - typically you want to use something meaningful that identifies the purposes of the VM, location, and (if there is more than one) instance number. We'll use "testvm-eus-01" for "Test VM in East US, instance 1". Come up with your own name based on where you place the VM.
   * Select a location close to you from the following list available in the Azure sandbox. Make sure to change the value in the below example command if you are using copy and paste.
     + westus2
     + southcentralus
     + centralus
     + eastus
     + westeurope
     + southeastasia
     + japaneast
     + brazilsouth
     + australiasoutheast
     + centralindia
   * Use "UbuntuLTS" for the image - this is Ubuntu Linux.
   * Use the Get-Credential cmdlet and feed the results into the Credential parameter.

**Important**

Please see the [**Linux VM FAQ**](https://docs.microsoft.com/en-us/azure/virtual-machines/linux/faq#what-are-the-username-requirements-when-creating-a-vm) for username and password limitations. Passwords must be 12 - 123 characters in length and meet 3 out of the following 4 complexity requirements:

* + - Have lower characters
    - Have upper characters
    - Have a digit
    - Have a special character (Regex match [\W\_])
  + Add the -OpenPorts parameter and pass "22" as the port - this will let us SSH into the machine.

PowerShellCopy

New-AzVm -ResourceGroupName [sandbox resource group name] -Name "testvm-eus-01" -Credential (Get-Credential) -Location "East US" -Image UbuntuLTS -OpenPorts 22

**Tip**

You can use the **Copy** button to copy commands to the clipboard. To paste, right-click on a new line in the Cloud Shell window and select **Paste** or use the Shift+Insert keyboard shortcut (⌘+V on macOS).

1. This will take a few minutes to complete. Once it does, you can query it and assign the VM object to a variable ($vm).

PowerShellCopy

$vm = Get-AzVM -Name "testvm-eus-01" -ResourceGroupName [sandbox resource group name]

1. Then query the value to dump out the information about the VM:

PowerShellCopy

$vm

You should see something like:

PowerShellCopy

ResourceGroupName : [sandbox resource group name]

Id : /subscriptions/xxxxxxxx-xxxx-aaaa-bbbb-cccccccccccc/resourceGroups/[sandbox resource group name]/providers/Microsoft.Compute/virtualMachines/testvm-eus-01

VmId : xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

Name : testvm-eus-01

Type : Microsoft.Compute/virtualMachines

Location : eastus

Tags : {}

HardwareProfile : {VmSize}

NetworkProfile : {NetworkInterfaces}

OSProfile : {ComputerName, AdminUsername, LinuxConfiguration, Secrets}

ProvisioningState : Succeeded

StorageProfile : {ImageReference, OsDisk, DataDisks}

1. You can reach into complex objects through a dot (".") syntax. For example, to see the properties in the VMSize object associated with the HardwareProfile section you can type:

PowerShellCopy

$vm.HardwareProfile

1. Or to get information on one of the disks:

PowerShellCopy

$vm.StorageProfile.OsDisk

1. You can even pass the VM object into other cmdlets. For example, this will retrieve the public IP address of your VM:

PowerShellCopy

$vm | Get-AzPublicIpAddress

1. With the IP address, you can connect to the VM with SSH. For example, if you used the username "bob", and the IP address is "205.22.16.5", then this command would connect to the Linux machine:

PowerShellCopy

ssh bob@205.22.16.5

Go ahead and log out by typing exit.

**Delete a VM**

Just to try out some more commands, let's delete the VM. We'll shut it down first.

PowerShellCopy

Stop-AzVM -Name $vm.Name -ResourceGroup $vm.ResourceGroupName

Now, let's delete the VM with the Remove-AzVM cmdlet:

PowerShellCopy

Remove-AzVM -Name $vm.Name -ResourceGroup $vm.ResourceGroupName

Try this command to list all the resources in your resource group:

PowerShellCopy

Get-AzResource -ResourceGroupName $vm.ResourceGroupName | ft

You should see a bunch of resources (disks, virtual networks, etc.) that all still exist.

OutputCopy

Microsoft.Compute/disks

Microsoft.Network/networkInterfaces

Microsoft.Network/networkSecurityGroups

Microsoft.Network/publicIPAddresses

Microsoft.Network/virtualNetworks

This is because the Remove-AzVM command *just deletes the VM*. It doesn't cleanup any of the other resources! At this point, we'd likely just delete the Resource Group itself and be done with it. However, let's just run through the exercise to clean it up manually. You should see a pattern in the commands.

1. Delete the Network Interface.

PowerShellCopy

$vm | Remove-AzNetworkInterface –Force

1. Delete the managed OS disks and storage account

PowerShellCopy

Get-AzDisk -ResourceGroupName $vm.ResourceGroupName -DiskName $vm.StorageProfile.OSDisk.Name | Remove-AzDisk -Force

1. Next, delete the virtual network.

PowerShellCopy

Get-AzVirtualNetwork -ResourceGroup $vm.ResourceGroupName | Remove-AzVirtualNetwork -Force

1. Delete the network security group.

PowerShellCopy

Get-AzNetworkSecurityGroup -ResourceGroup $vm.ResourceGroupName | Remove-AzNetworkSecurityGroup -Force

1. And finally, the public IP address.

PowerShellCopy

Get-AzPublicIpAddress -ResourceGroup $vm.ResourceGroupName | Remove-AzPublicIpAddress -Force

We should have caught all the created resources; check the resource group just to be sure. We did a lot of manual commands here but a better approach would have been to write a *script* so we could reuse this logic later to create or delete a VM. Let's look at scripting with PowerShell.