Azure Resource Manager

- Azure Resource Manager (ARM) enables you to work with the different components (referred to as resources in Azure terminology) in your application solution as a group.
- This gives us the ability to deploy, update or delete all the resources for your application in a single operation if needed.
- Another very useful feature of ARM is that it allows us to apply tags to related resources. This in turn, allows you to better understand your organization's billing by being able to costs associated with a group of resources sharing the same tag.

In this article, we will discuss three of the most popular methods Azure administrators use to **interact with Azure Resource Manager**. We will also be including practical demonstrations of using these methods to step by step **create virtual machine in Azure Cloud** using each of the methods.

Different Azure Terminologies

Before we move on to the discussion about the different methods, there is some Azure related terminology that you should be familiar with.

- **Resource** - A manageable item that is available through Azure. Some common resources are a virtual machine, storage account, web app, database, and virtual network, but there are many more.
- **Resource group** - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization.
- **Resource provider** - A service that supplies the resources you can deploy and manage through Resource Manager. Each resource provider offers operations for working with the resources that are deployed. Some common resource providers are Microsoft.Compute, which supplies the virtual machine resource, Microsoft.Storage, which supplies the storage account resource, and Microsoft.Web, which supplies resources related to web apps.

Different methods to create VM in Azure

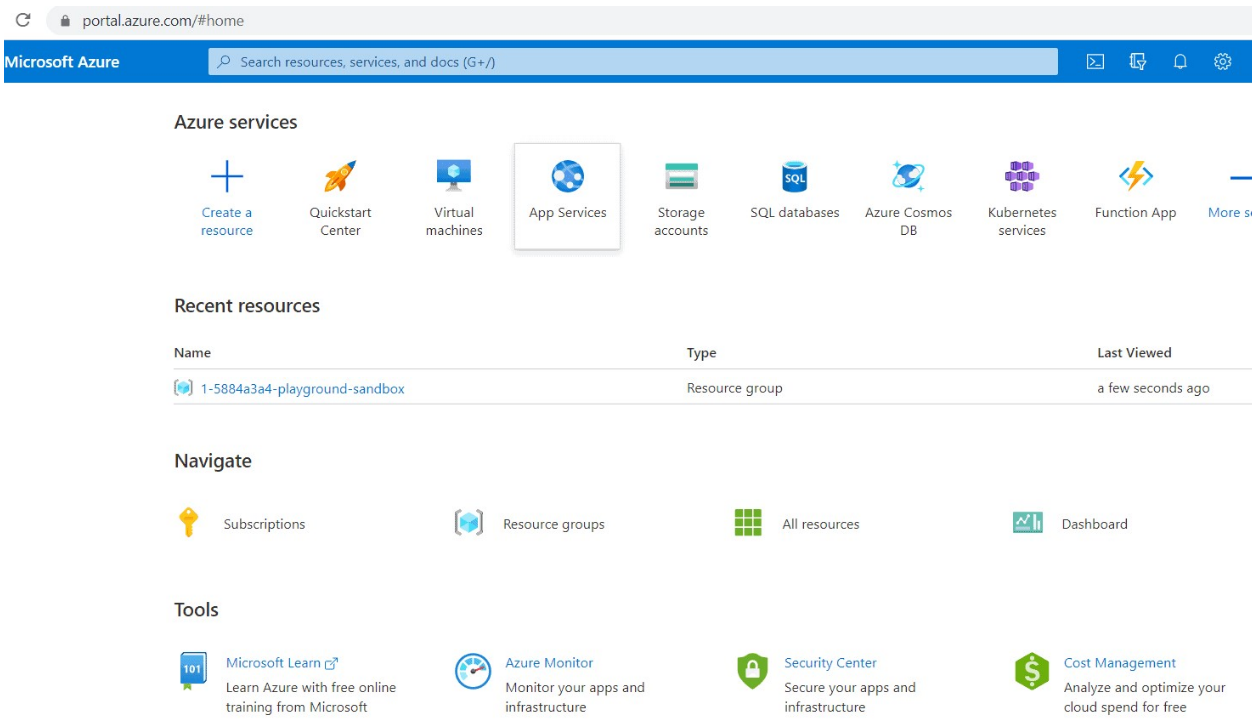
In this article we'll be exploring the following three methods to create VM in Azure

- The Azure Portal
- Azure CLI
- Azure PowerShell

Method 1: Create VM using the Azure portal

The Azure portal is our login portal to the Azure cloud and this is where we can create and modify our resources. To login to the Azure portal open the URL portal.azure.com and enter your credentials.

Once done, you would see the following page which is you home page.

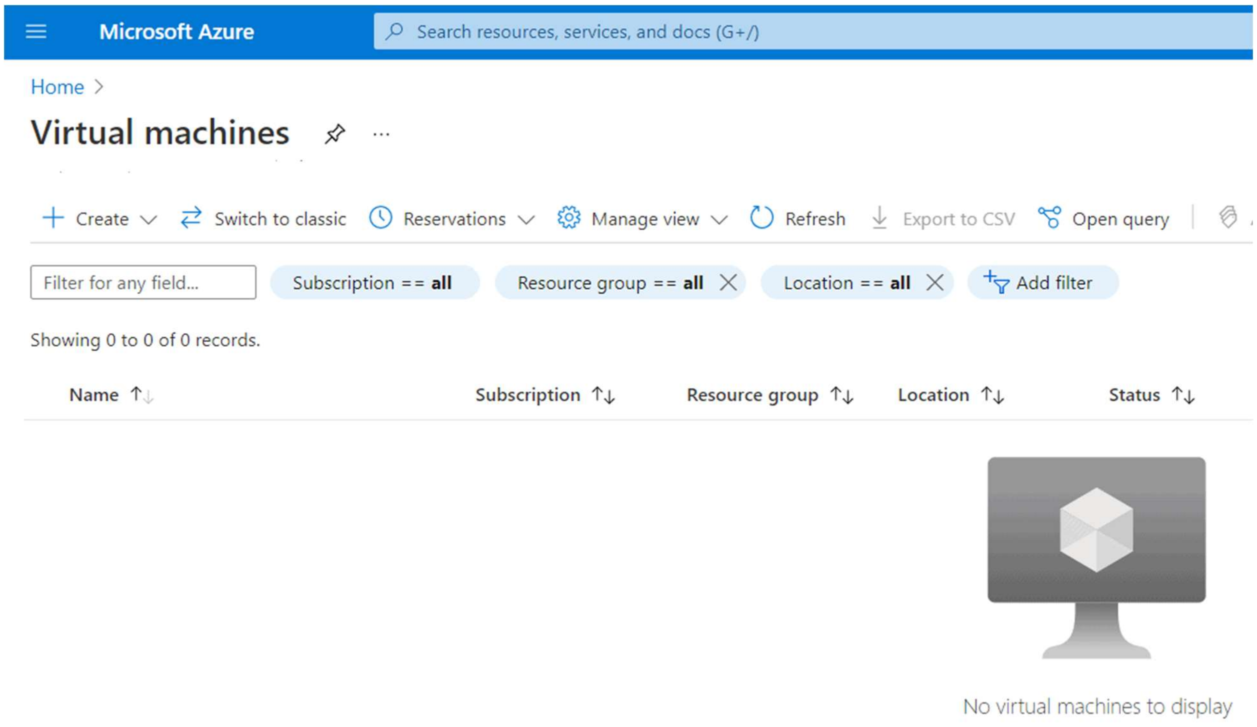


Azure Portal

Now that we've familiarized ourselves with the Azure portal, let's create a virtual machine using the portal.

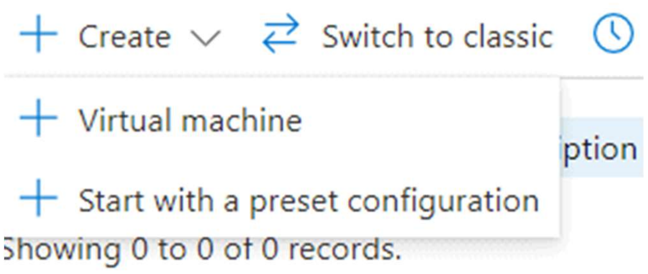
Step 1: Create new VM using the Azure portal

Click on **Virtual Machines** under the Azure Services section in Azure home page.



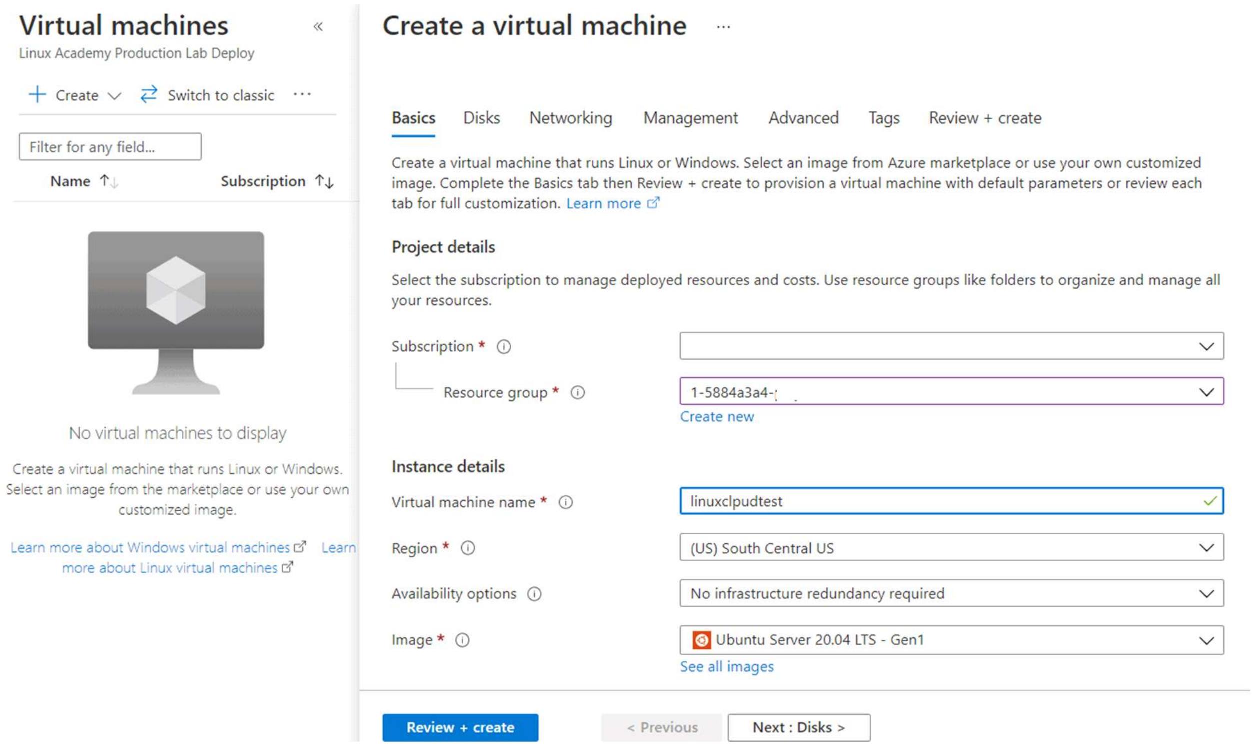
Create Virtual Machine using Azure Portal

From here click on create and click on **Virtual Machine**.


Create
Switch to classic
Virtual machine
Start with a preset configuration
Showing 0 to 0 of 0 records.

Step 2: Enter VM details

In the below menu. Keep all the defaults and type in a suitable name for your VM.



Create a virtual machine that runs Linux or Windows. Select an image from the marketplace or use your own customized image.

Create a virtual machine

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. Learn more

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *
Resource group * 1-5884a3a4-
Create new

Instance details

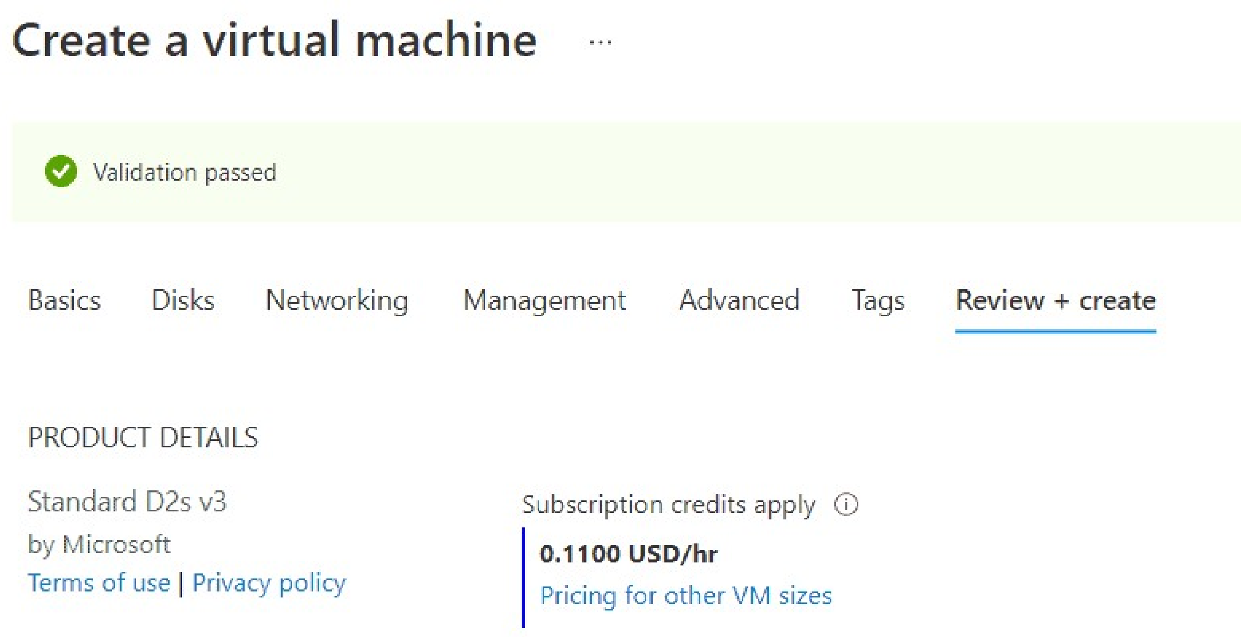
Virtual machine name * linuxclpudtest
Region * (US) South Central US
Availability options No infrastructure redundancy required
Image * Ubuntu Server 20.04 LTS - Gen1
See all images

Provide VM Details

We are using and Ubuntu image for the purpose of this demonstration. Since this is a basic VM creation demo, we’ll skip the advanced options. Now click on **Review + create**.

Step 3: Confirm VM details

In the next windows, the VM size and corresponding pricing details are displayed. Also, the subscribers’ information name, email and phone number fields are displayed.


Create a virtual machine

Validation passed

PRODUCT DETAILS

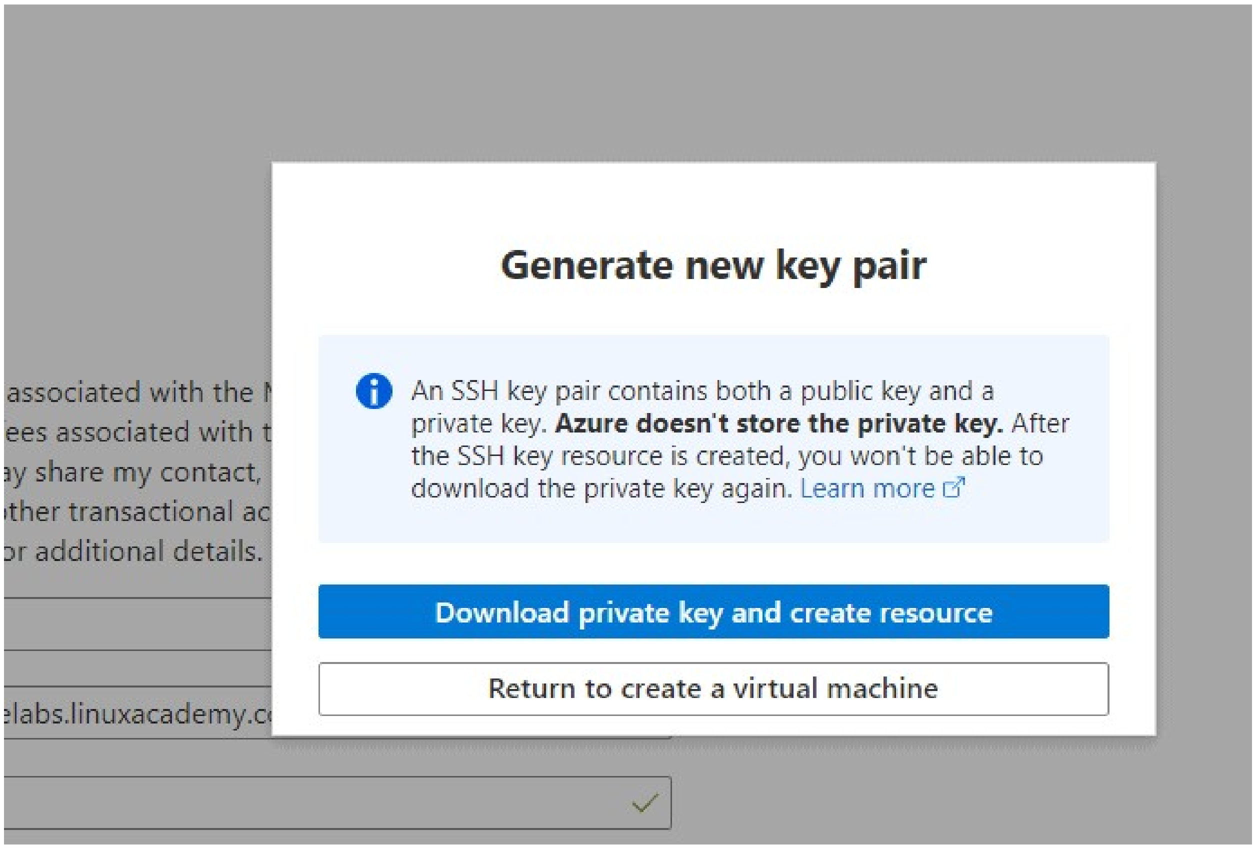
Standard D2s v3
by Microsoft
Terms of use | Privacy policy

Subscription credits apply

0.1100 USD/hr
Pricing for other VM sizes

Confirm VM Details

Towards the bottom left of the window, you'll see a create button. You can also click on the previous button next to the create button in case you'd like to modify some settings. In our case, we'll click on create. After clicking the create button, you'll be prompted to download a private key as shown below.



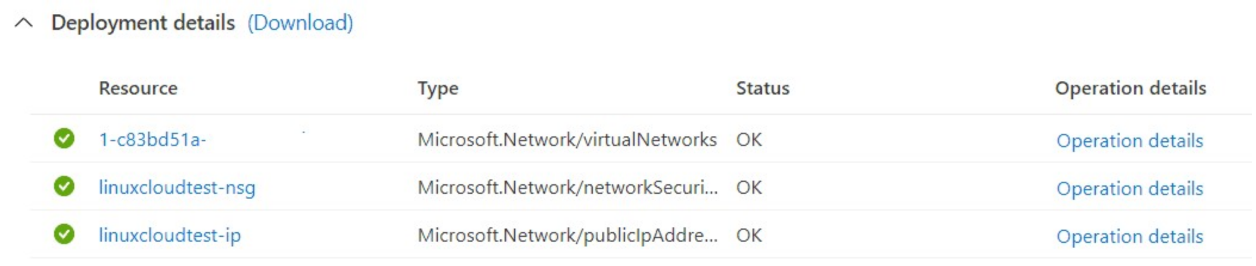
Generate new key pair

An SSH key pair contains both a public key and a private key. **Azure doesn't store the private key.** After the SSH key resource is created, you won't be able to download the private key again. Learn more

Download private key and create resource
Return to create a virtual machine

Generate KEY Pair

Click on this button to complete the creation of your virtual machine. The virtual machine creation will take a couple of seconds. While the VM is being deployed, you'll be able to see the status of the related resources as they get deployed as well.



Deployment details (Download)

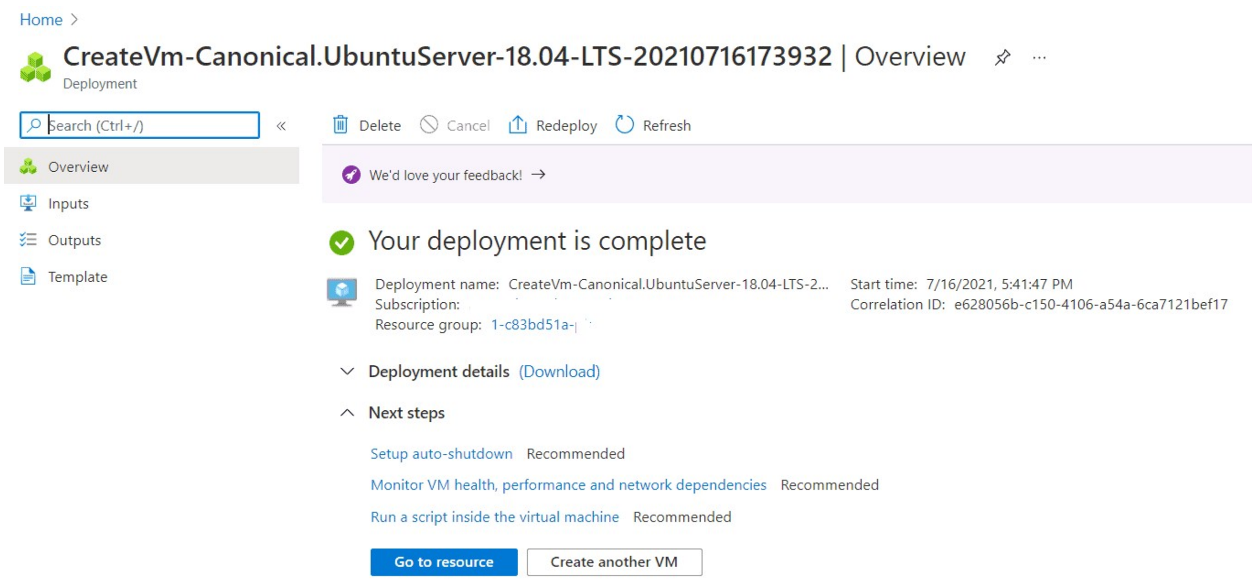
	Resource	Type	Status	Operation details
✓	1-c83bd51a-	Microsoft.Network/virtualNetworks	OK	Operation details
✓	linuxcloudtest-nsg	Microsoft.Network/networkSecuri...	OK	Operation details
✓	linuxcloudtest-ip	Microsoft.Network/publicIpAddre...	OK	Operation details

Deployment Status

This will also download a ssh public key to your downloads folder. You need to use this key to connect to your virtual machine once it boots up since password based authentication is not permitted by default in Azure VMs.

Step 4: Confirm VM deployment

Once the VM deployment completes successfully, we'll be shown the below message and a button towards the end named **Go to resource**. Clicking on this button will navigate you to the VM management window for the VM that we just created.


Home >

CreateVm-Canonical.UbuntuServer-18.04-LTS-20210716173932 | Overview

Deployment

Your deployment is complete

Deployment name: CreateVm-Canonical.UbuntuServer-18.04-LTS-2...
Subscription:
Resource group: 1-c83bd51a-

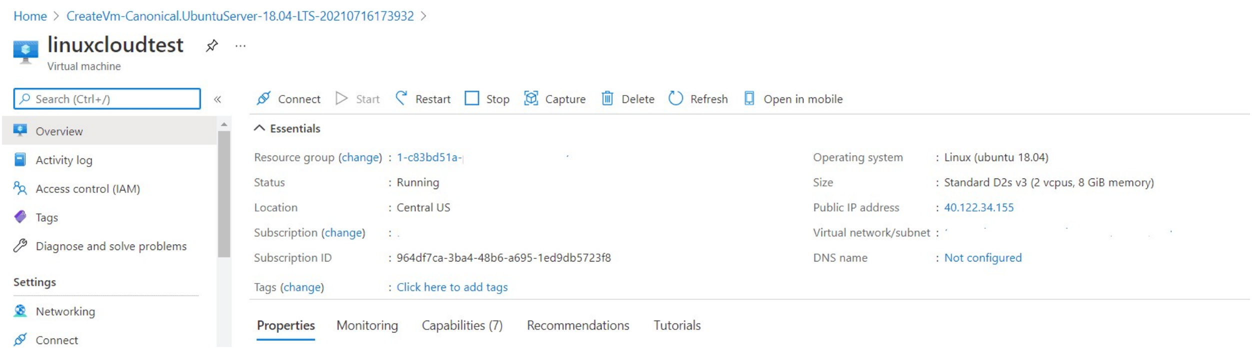
Start time: 7/16/2021, 5:41:47 PM
Correlation ID: e628056b-c150-4106-a54a-6ca7121bef17

Deployment details (Download)

Next steps

Setup auto-shutdown Recommended
Monitor VM health, performance and network dependencies Recommended
Run a script inside the virtual machine Recommended

The management window will display a breadth of information including the public IP address that we may use to connect to the VM via ssh.



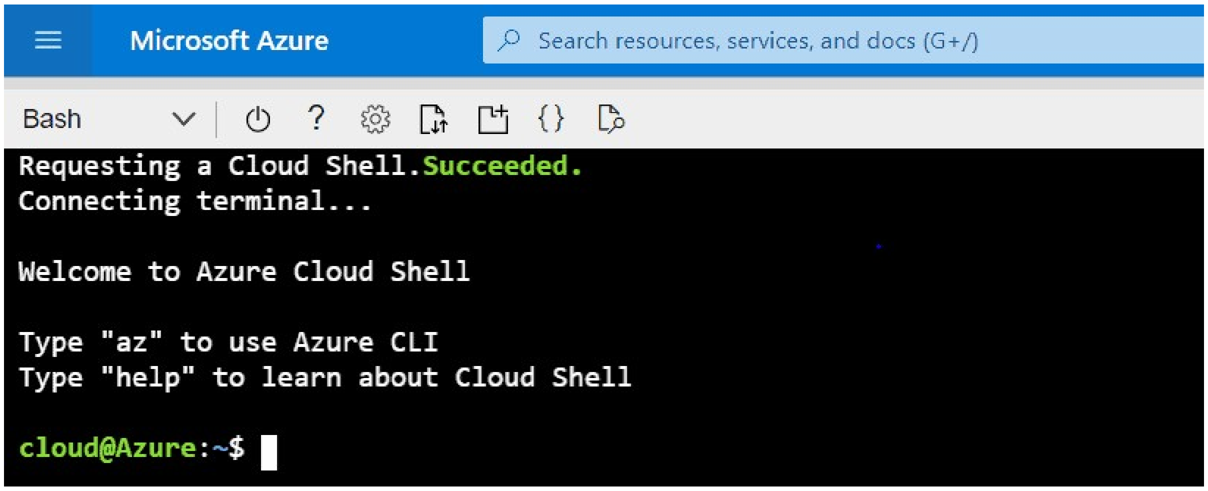
Essentials

Resource group (change)	: 1-c83bd51a-	Operating system	: Linux (ubuntu 18.04)
Status	: Running	Size	: Standard D2s v3 (2 vcpus, 8 GiB memory)
Location	: Central US	Public IP address	: 40.122.34.155
Subscription (change)	:	Virtual network/subnet	:
Subscription ID	: 964df7ca-3ba4-48b6-a695-1ed9db5723f8	DNS name	: Not configured
Tags (change)	: Click here to add tags		

Method 2: Create VM with Azure CLI (bash on cloud shell)

Earlier in the article, we explained how to start the Azure Cloud Shell command line interface and now we will use it to interact with Azure Resource Manager by creating a virtual machine.

We can use the Azure CLI to manage our Azure resources without having to log in to the Azure portal. For Linux system admins who are used to working with the command line, this tool would prove to be very helpful. Although, we can install an SDK for Azure CLI on our local system, we can also use it via the Azure cloud shell. For the purpose of our demonstration, we'll be using Azure CLI via Cloud Shell. When we select bash while launching Cloud shell, we are dropped into a command prompt as shown below.



```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

cloud@Azure:~$
```

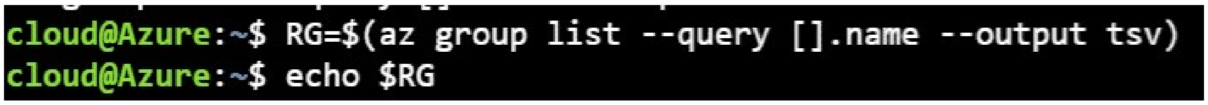
From within this terminal, we can now execute our Azure CLI commands.

Step 1: Determine Resource Group name

The resource group is a logical container where we group our Azure resources. All our Azure resources must belong to a resource group. Although we can type in the name of the resource group manually, in this example, we'll find it out programmatically using an Azure CLI command and assign the result to a variable. The below Azure CLI command list the resource group that we are currently using.

```
az group list --query [].name --output tsv
```

Sample Output:



```
cloud@Azure:~$ RG=$(az group list --query [].name --output tsv)
cloud@Azure:~$ echo $RG
```

The above command assigns the output of the Azure CLI command to a variable named RG. Here's a quick explanation of what the command is doing

- The "`az group list`" command will print information about the currently active resource group in JSON format.
- The `--query` sub-command can be used to filter out the result and `[].name` will print only the name of the resource group.
- Since this name is still in JSON format, we modify the output format to tsv so that we may remove the additional braces encompassing the initial JSON output and use the output directly inside a variable.

We can view the contents of the variable by just printing it out using `echo` command. Most commands found in a Linux distro will work in this interface.

Step 2: Create VM

To create a virtual machine, we will use the following command:

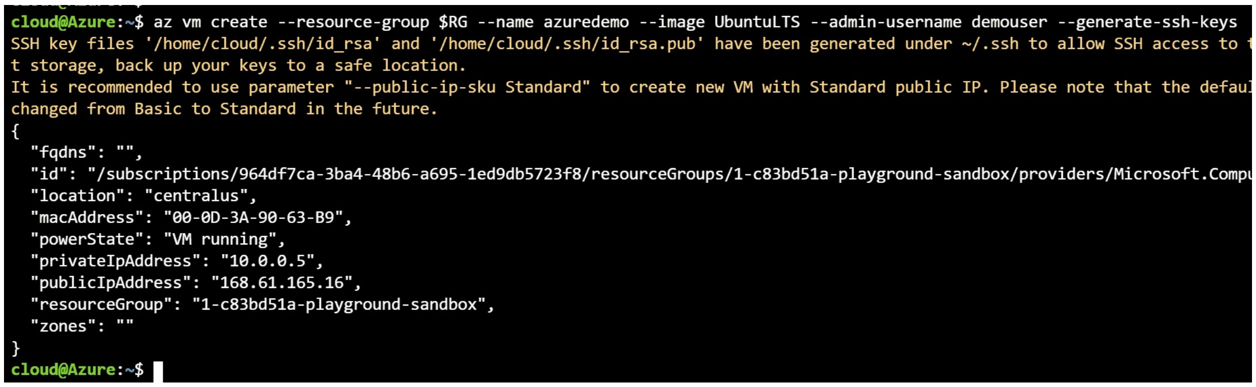
```
cloud@Azure:~$ az vm create --resource-group $RG --name azuredemo --image UbuntuLTS --
admin-username demouser --generate-ssh-keys
```

SSH key files '/home/cloud/.ssh/id_rsa' and '/home/cloud/.ssh/id_rsa.pub' have been generated under ~/.ssh to allow SSH access to the VM. If using machines without permanent storage, back up your keys to a safe location.

It is recommended to use parameter "--public-ip-sku Standard" to create new VM with Standard public IP. Please note that the default public IP used for VM creation will be
changed from Basic to Standard in the future.

```
{

  "fqdns": "",

  "id": "/subscriptions/964df7ca-3ba4-48b6-a695-1ed9db5723f8/resourceGroups/1-c83bd51a-playground-
sandbox/providers/Microsoft.Compute/virtualMachines/azuredemo",

  "location": "centralus",

  "macAddress": "00-0D-3A-90-63-B9",

  "powerState": "VM running",

  "privateIpAddress": "10.0.0.5",

  "publicIpAddress": "168.61.165.16",

  "resourceGroup": "1-c83bd51a-playground-sandbox",

  "zones": ""

}
```

Sample Output:



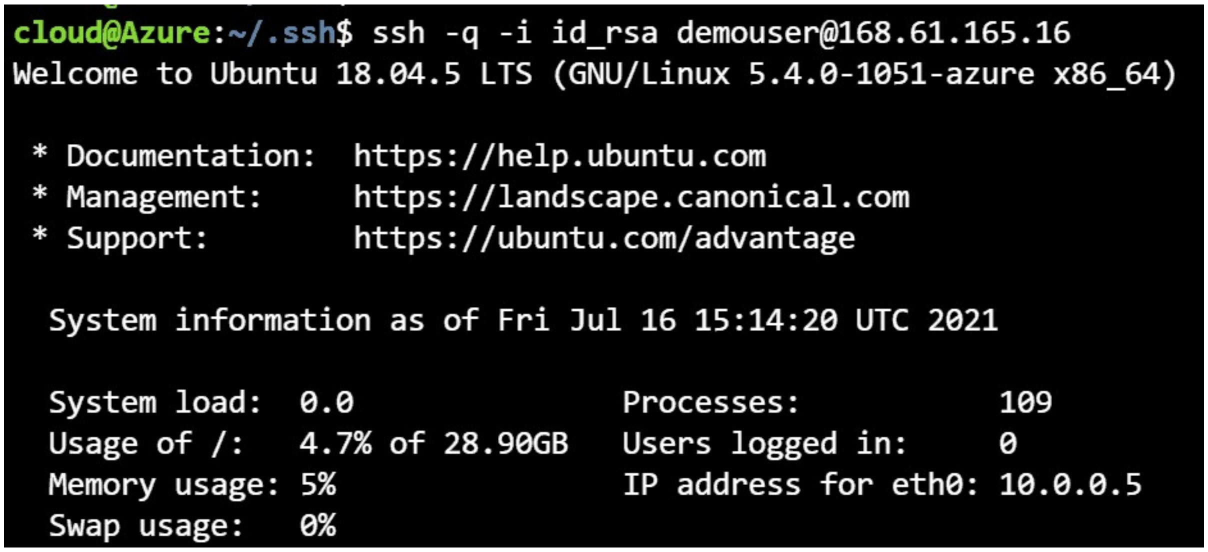
```
cloud@Azure:~$ az vm create --resource-group $RG --name azuredemo --image UbuntuLTS --admin-username demouser --generate-ssh-keys
SSH key files '/home/cloud/.ssh/id_rsa' and '/home/cloud/.ssh/id_rsa.pub' have been generated under ~/.ssh to allow SSH access to
t storage, back up your keys to a safe location.
It is recommended to use parameter "--public-ip-sku Standard" to create new VM with Standard public IP. Please note that the defaul
changed from Basic to Standard in the future.
{
  "fqdns": "",
  "id": "/subscriptions/964df7ca-3ba4-48b6-a695-1ed9db5723f8/resourceGroups/1-c83bd51a-playground-sandbox/providers/Microsoft.Compu
  "location": "centralus",
  "macAddress": "00-0D-3A-90-63-B9",
  "powerState": "VM running",
  "privateIpAddress": "10.0.0.5",
  "publicIpAddress": "168.61.165.16",
  "resourceGroup": "1-c83bd51a-playground-sandbox",
  "zones": ""
}
cloud@Azure:~$
```

Here is a quick description of the arguments we used in the above command while creating the VM

- **--resource-group** argument specifies the resource group within which this VM will reside. In our case, we provided the variable name that holds the name of the resource group.
- **--name** argument denotes the name of the virtual machine.
- **--image** argument specifies the OS image that we'd like to use. You can browse the Azure portal for a list of available images.
- **--admin-username** argument will create a user inside the VM that will have super user privileges since it's a common security measure in many environments to prevent direct root login.
- **--generate-ssh-keys** will generate a pair of public-private ssh keys and save them in the home directory of the currently logged in user. These keys are important since they will be used for authentication.

Step 3: Verify connectivity to the VM

Once, our VM is created we can connect to it via ssh from within cloud shell itself as shown below



```
cloud@Azure:~/.ssh$ ssh -q -i id_rsa demouser@168.61.165.16
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1051-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Jul 16 15:14:20 UTC 2021

  System load:  0.0               Processes:           109
  Usage of /:   4.7% of 28.90GB   Users logged in:     0
  Memory usage: 5%                IP address for eth0: 10.0.0.5
  Swap usage:   0%
```

In case you do not remember the public IP address of your VM, you can always query it form within the Azure CLI using the below command:

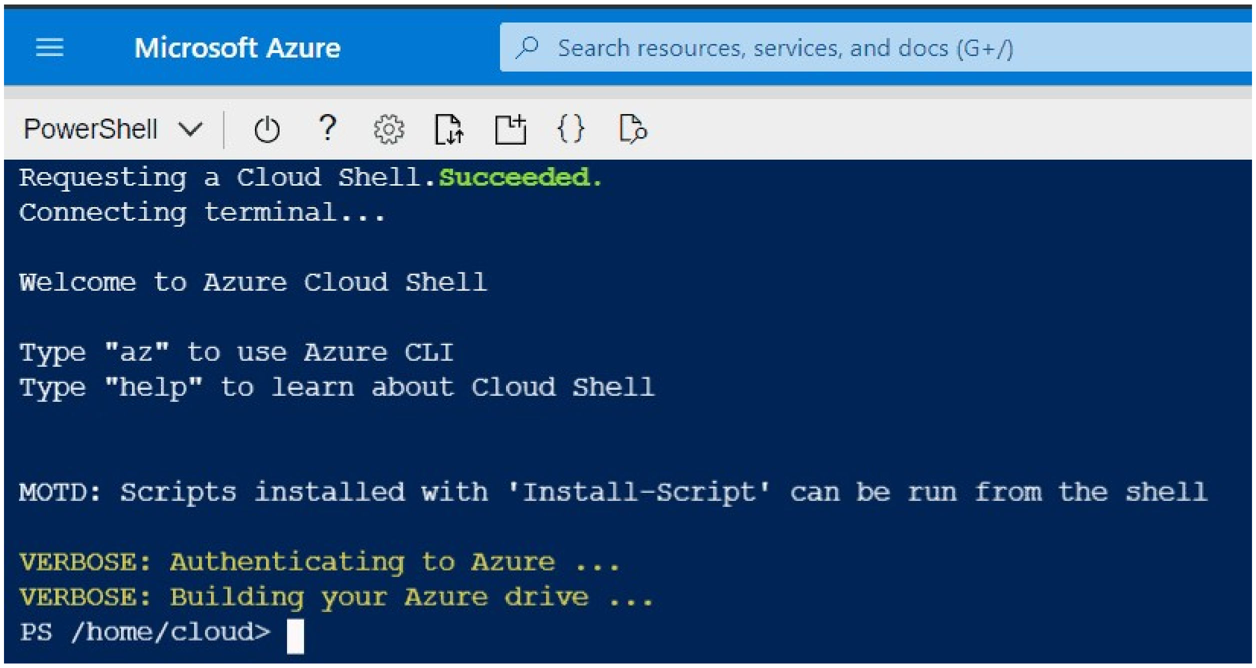
```
cloud@Azure:~$ az vm show -d --resource-group $RG --name azuredemo --query "publicIps" -o tsv


168.61.165.16
```

Method 3: Create VM Using Azure PowerShell

The third and final method to create VM is using Azure PowerShell.

This is essentially a set of PowerShell cmdlets that we use to interact with the Azure Resource Manager. Like the Azure CLI we can install Azure PowerShell cmdlets using an SDK or use it via the Azure Cloud shell. In our case, we'll be using it via the Cloud shell. To switch between bash and PowerShell we just need to click the button on the top left of the Azure Cloud shell window that says bash and select PowerShell. This will drop us into an Azure PowerShell prompt. Or you can also select PowerShell at the prompt when you initially launch Azure Cloud Shell.

The Azure PowerShell prompt will appear as follows:



```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell


MOTD: Scripts installed with 'Install-Script' can be run from the shell

VERBOSE: Authenticating to Azure ...
VERBOSE: Building your Azure drive ...
PS /home/cloud>
```

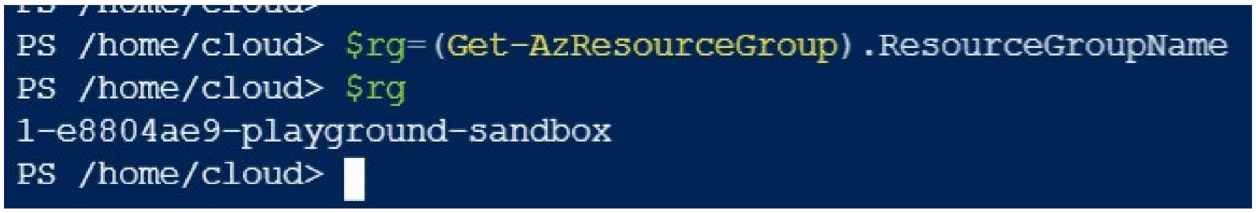
Let us now create a VM using Azure PowerShell

Step 1: Determine Resource Group

Like our last example with Azure CLI, we'll not use the resource group name directly and will instead determine it programmatically and store its value in a variable using the following command:

```
$rg=(Get-AzResourceGroup).ResourceGroupName
```

Sample Output:



```
PS /home/cloud> $rg=(Get-AzResourceGroup).ResourceGroupName
PS /home/cloud> $rg
1-e8804ae9-playground-sandbox
PS /home/cloud>
```

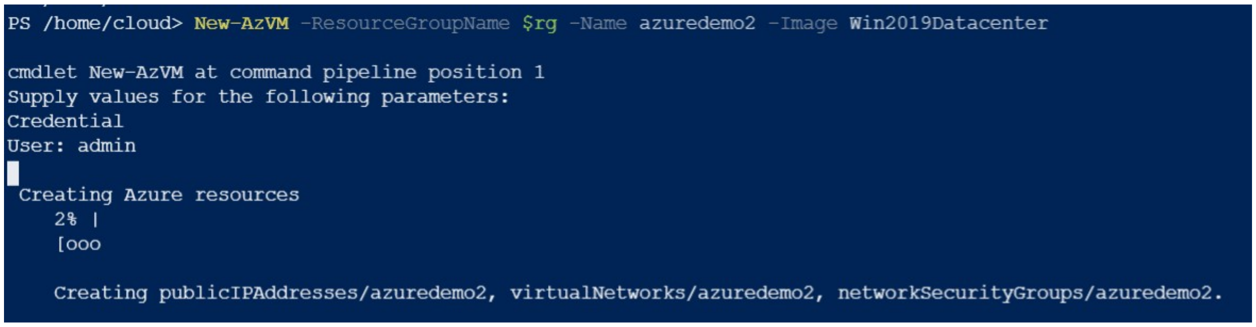
A great thing about Azure PowerShell is that it allows tab completion which means you don't need to memorize lengthy commands.

Step 2: Create virtual machine

To create a virtual machine using Azure PowerShell we will use the New-AzVM cmdlet. We will provide the resource group name, the virtual machine name and the name of the image to use as arguments to the New-AzVM cmdlet as shown in the below command.

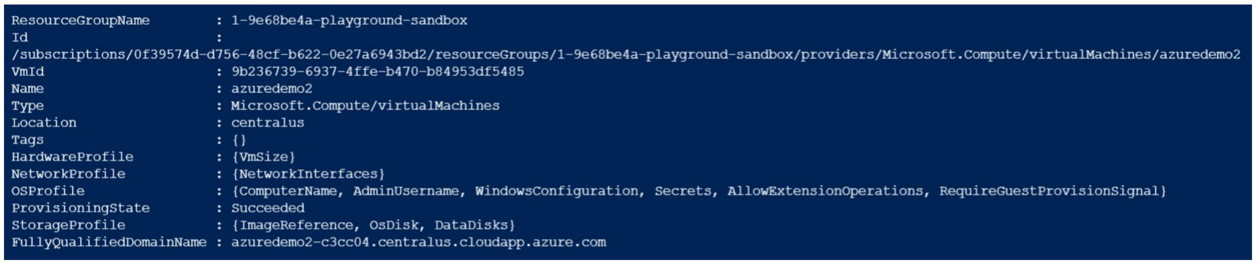
```
New-AzVM -ResourceGroupName $rg -Name azuredemo2 -Image Win2019Datacenter
```

Sample Output:



```
PS /home/cloud> New-AzVM -ResourceGroupName $rg -Name azuredemo2 -Image Win2019Datacenter

cmdlet New-AzVM at command pipeline position 1
Supply values for the following parameters:
Credential
User: admin

 Creating Azure resources
    2% |
    [ooo

    Creating publicIPAddresses/azuredemo2, virtualNetworks/azuredemo2, networkSecurityGroups/azuredemo2.
```

Here we've used the Windows 2019 Datacenter edition as our VM image and named it azuredemo2. Once the command executes, we'll be prompted to enter credentials for our admin user. After the VM build completes, the VM details will be displayed in JSON format as shown below.



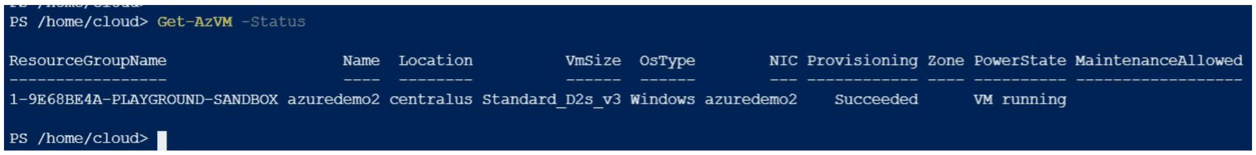
```
ResourceGroupName        : 1-9e68be4a-playground-sandbox
Id                       :
/subscriptions/0f39574d-d756-48cf-b622-0e27a6943bd2/resourceGroups/1-9e68be4a-playground-sandbox/providers/Microsoft.Compute/virtualMachines/azuredemo2
VmId                     : 9b236739-6937-4ffe-b470-b84953df5485
Name                     : azuredemo2
Type                     : Microsoft.Compute/virtualMachines
Location                 : centralus
Tags                     : {}
HardwareProfile          : {VmSize}
NetworkProfile           : {NetworkInterfaces}
OSProfile                : {ComputerName, AdminUsername, WindowsConfiguration, Secrets, AllowExtensionOperations, RequireGuestProvisionSignal}
ProvisioningState        : Succeeded
StorageProfile           : {ImageReference, OsDisk, DataDisks}
FullyQualifiedDomainName : azuredemo2-c3cc04.centralus.cloudapp.azure.com
```

Step 3: Verify VM build

To check the status of the virtual machine, we can use the Get-AzVM -status command.

```
Get-AzVM -status
```

Sample Output:



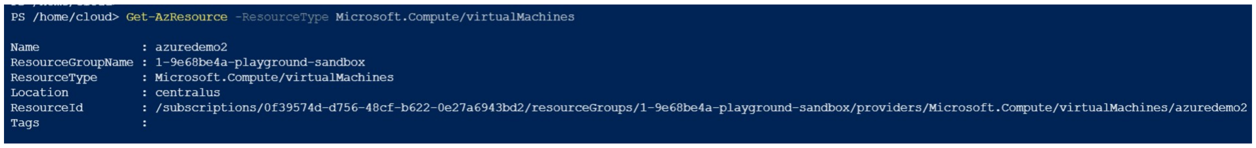
```
PS /home/cloud> Get-AzVM -Status

ResourceGroupName                  Name  Location          VmSize  OsType        NIC Provisioning Zone PowerState MaintenanceAllowed
-----------------                  ----  --------          ------  ------        --- ------------ ---- ---------- ------------------
1-9E68BE4A-PLAYGROUND-SANDBOX azuredemo2 centralus Standard_D2s_v3 Windows azuredemo2    Succeeded      VM running

PS /home/cloud>
```

Advertisement

We can also use the Get-AzResource cmdlet to display information about VMs running in Azure.

```
Get-AzResource -ResourceType Microsoft.Compute/virtualMachines
```

Sample Output:

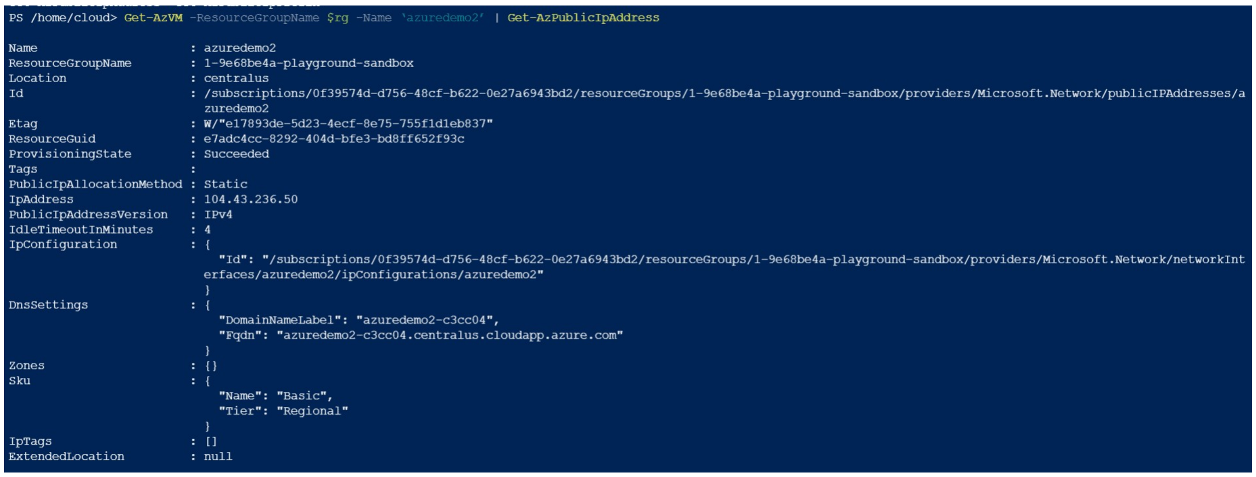


```
PS /home/cloud> Get-AzResource -ResourceType Microsoft.Compute/virtualMachines

Name              : azuredemo2
ResourceGroupName : 1-9e68be4a-playground-sandbox
ResourceType      : Microsoft.Compute/virtualMachines
Location          : centralus
ResourceId        : /subscriptions/0f39574d-d756-48cf-b622-0e27a6943bd2/resourceGroups/1-9e68be4a-playground-sandbox/providers/Microsoft.Compute/virtualMachines/azuredemo2
Tags              :
```

To get the public IP information for the VM, we can issue the following command

```
Get-AzVM -ResourceGroupName $rg -Name ‘azuredemo2’ | Get-AzPublicIpAddress
```

Sample Output:



```
PS /home/cloud> Get-AzVM -ResourceGroupName $rg -Name 'azuredemo2' | Get-AzPublicIpAddress

Name                     : azuredemo2
ResourceGroupName        : 1-9e68be4a-playground-sandbox
Location                 : centralus
Id                       : /subscriptions/0f39574d-d756-48cf-b622-0e27a6943bd2/resourceGroups/1-9e68be4a-playground-sandbox/providers/Microsoft.Network/publicIPAddresses/a
                           zuredemo2
Etag                     : W/"e17893de-5d23-4ecf-8e75-755f1d1eb837"
ResourceGuid             : e7adc4cc-8292-404d-bfe3-bd8ff652f93c
ProvisioningState        : Succeeded
Tags                     :
PublicIpAllocationMethod : Static
IpAddress                : 104.43.236.50
PublicIpAddressVersion   : IPv4
IdleTimeoutInMinutes     : 4
IpConfiguration          : {
                             "Id": "/subscriptions/0f39574d-d756-48cf-b622-0e27a6943bd2/resourceGroups/1-9e68be4a-playground-sandbox/providers/Microsoft.Network/networkInt
                           erfaces/azuredemo2/ipConfigurations/azuredemo2"
                           }
DnsSettings              : {
                             "DomainNameLabel": "azuredemo2-c3cc04",
                             "Fqdn": "azuredemo2-c3cc04.centralus.cloudapp.azure.com"
                           }
Zones                    : {}
Sku                      : {
                             "Name": "Basic",
                             "Tier": "Regional"
                           }
IpTags                   : []
ExtendedLocation         : null
```

To connect to the VM, we’d need to go to the Azure portal and download the corresponding RDP file for the VM and then use it to connect. This concludes our demonstration on using Azure PowerShell for interacting with the Azure Resource Manager. The PowerShell cmdlets being used might seem like difficult to remember but since PowerShell allows tab completion, using cmdlets becomes fairly easy leaving not much to memorize.