Planning and deploying Azure VMs

Compared to a traditional on-premises virtual machine, a cloud-based VM allows you to quickly scale up and down as requested; you don’t need to buy and maintain the physical hardware that runs the virtual machine. Another great advantage is that you only pay for what you use in the cloud. In practice you, still need to maintain the virtual machine by configuring, patching, and maintaining the software that runs on the VM.  However, its flexible pricing makes it more efficient in terms of time and costs.

Identifying the workloads

Before deploying a virtual machine in Azure, we will always start by identifying the workloads, whether the best deployment solution for the target workloads is on Azure VM or maybe on other Azure offerings.

In real life, organizations thinking about migrating their existing application to Azure quickly should not only take into account the technical concerns but also the financial aspects. Besides, certain types of workloads are a great fit for hosting in an Azure IaaS environment, for example, when you need a high flexibility to control your OS and don't mind higher administration efforts than other PaaS offerings in Azure.

However, not every application is always a suitable fit for the cloud, as the following case :

* Certain low volumes or limited growth workloads where it might be cheaper to run the service or applications on commodity hardware on-premises
* Certain regulated environment workloads where the type of data is more sensitive or credentials requested by an organization needs to be kept on-premises or using other private cloud platforms such as Open Stack or the extension of the public cloud such as Azure Stack or VMware cloud on AWS.

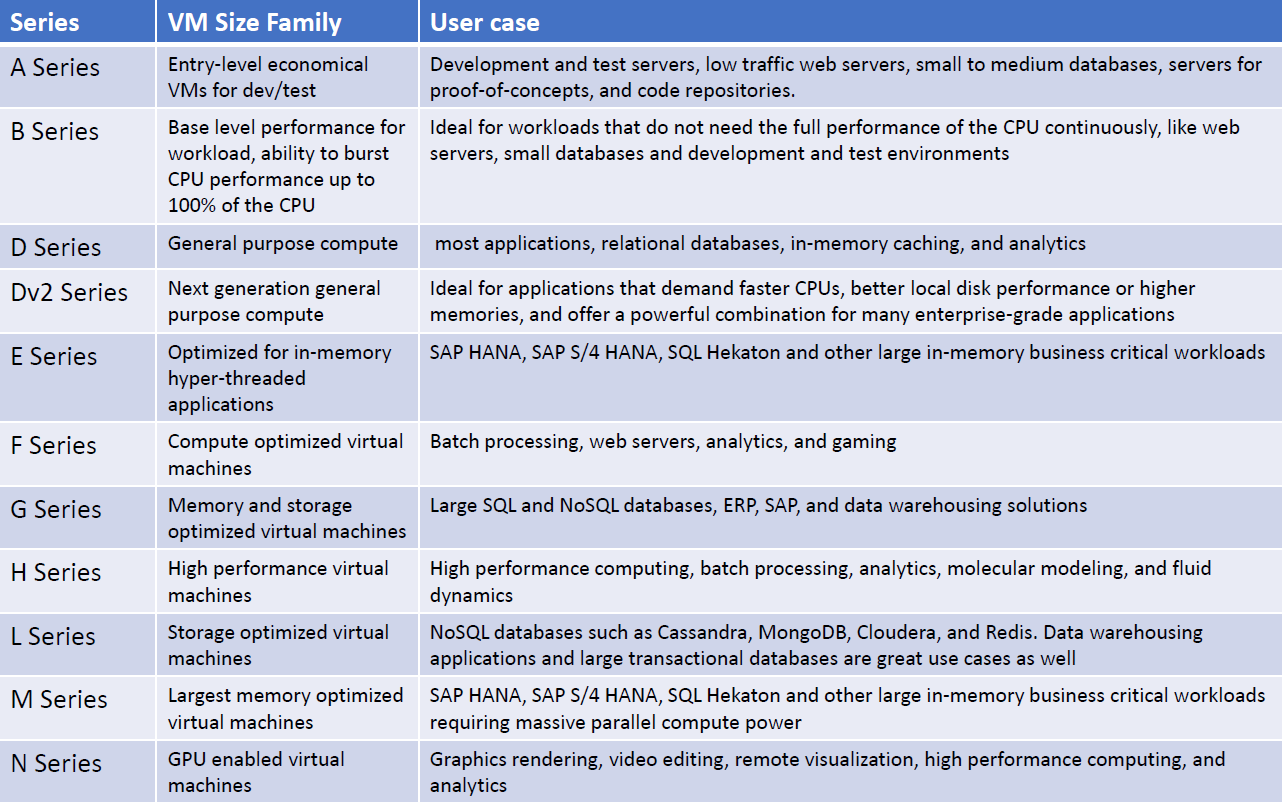
Choosing the appropriate Azure VM sizing

If you decide to go further with Azure VM, you should choose the SKU of the virtual machine or the size of virtual machine of the Azure virtual machine provided by Microsoft Azure. In Azure, the SKU or sizing is based on a variety of options for the number and speed of its processors (VCPU), amount of memory (RAM), the number of data disks you can attach to it, the maximum size of a temporary disk, IOPS, and the type of disks for the operating system. Generally, when the VM sizes support premium storage, which uses **solid-state drive** (**SSD**), the maximum aggregate disk I/O performance would be better than standard storage with a **hard disk drive** (**HDD**).

Virtual machines are available in several different sizes. When your requirements change, it is easy to resize the VM, which means you can use more advanced VM configurations, such as a more powerful CPU or larger RAM.

You can choose the appropriate size depending on your technical requirements. Try to balance the appropriate size of VMs and the number of VMs in your project. In real life, very often, the final decision on the size of VMs and number of instances for DevTest or the production environment would be made after a period of workload testing.

The following are the available categories of Azure virtual machines that are available so far:



VM size family and use cases

Azure VM storage options

Every Azure VM generally has two disks:

* An **operating system disk**, which is registered as a SATA drive and labeled as the C: drive for Windows machines and the repository /dev/sda for Linux machines, by default.
* A **temporary disk**, which is labeled as the D: drive for Windows that stores the pagefile.sys file. For a Linux machine, the repository /dev/sdb is the temporary disk.

Besides these two disks, Azure VMs can be attached to a number of **data disks**. The operating system disk is created from a VM image; the operating system disk and data disks are **virtual hard disks** (**VHDs**) stored in a page blob in an Azure storage account.

Managed disks versus unmanaged disks

Azure **unmanaged disks** are the disks created and managed by service administrators. Azure **Managed Disks** are the disks that allow Microsoft Azure to handle the disk management of the IaaS VMs.

Compared to unmanaged disks, Azure-managed disks provide better scalability while scaling the VMs with VMSS (scale sets) and breaks the limit of IOPS per storage account, that is, Azure has a limit of 20,000 IOPS per storage account which will impact the number of VMs that can be created per storage account. Azure Managed Disks are recommended by Microsoft for storing persistent storage of data while creating Azure VMs.

Both of them have standard and premium pricing tiers. A standard tier is based on HDD. A premium tier is based on high-performance SSD to support I/O intensive workloads.

Unmanaged disks are available for **locally redundant storage** (**LRS**), **zone-redundant storage** (**ZRS**), **geo-redundant storage**(**GRS**), and **read-access geo-redundant storage** (**RA-GRS**). At the time of writing this book managed disks are only available for LRS.

Azure reserved VM instances (RIs) versus pay–as–you–go instances

Compared to pay-as-you-go prices, Azure also provides a way to cut down VM costs in a significant way by purchasing the **reserved VM instances**(**RIs**), which link to a 1-year or 3-year engagement on both Windows and Linux VMs.

Purchasing Azure-reserved VM instances can help users save up to 72% VMs costs. Additionally, it accumulates the **Azure Hybrid Benefit**, which can help users save up to 82% VM costs.

Deploying an Azure VM

To facilitate the deployments of different workloads using an Azure virtual machine, Microsoft Azure offers different ways to release the deployment. Users can deploy their virtual machines via Azure Portal, Azure PowerShell, Azure CLI, Azure Cloud Shell, or ARM Template.

The deployment usually starts from choosing an OS image from the Azure Marketplace. The Azure Marketplace provides images of various Microsoft and Linux operating systems, such as CentOS, Debian, Ubuntu, and so on, and also provides preconfigured products with a **ready-to-use** image. Microsoft and Microsoft's third-party partner provides various popular image solutions, such as Windows Server 2016 Data Center, Red Hat Enterprise, SUSE Linux Enterprise, the Data Science Virtual Machine, and so on.

Creating Azure VMs

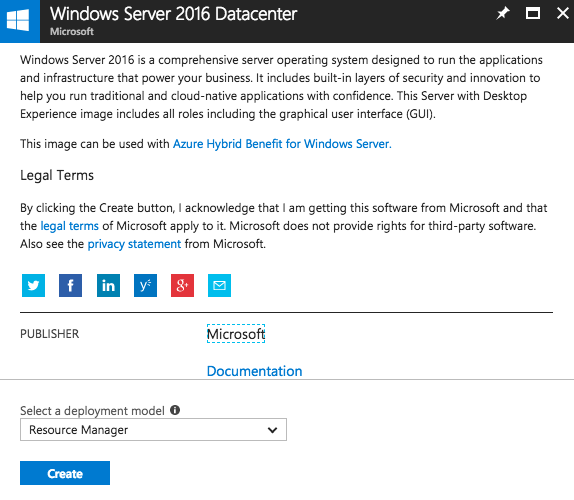
There are many ways to create an Azure VM, for example, via Azure Portal, Azure PowerShell, or Azure CLI.

Creating Azure VMs via the Azure Portal

Let’s take a look at how to create an Azure VM via the Azure Portal with a Windows image and a Linux image in the Azure Marketplace.

Creating an Azure VM with a Windows image

Via the Azure Portal, click on Create a resource and then on the Compute option, then choose the appropriate Windows image (**Windows Server 2016 Datacenter**, which is the latest one). The following screenshot shows the first page that will appear when you start to deploy an Azure Windows VM; be careful to choose the ARM as a deployment model, which will register your resource with Azure Resource Manager:

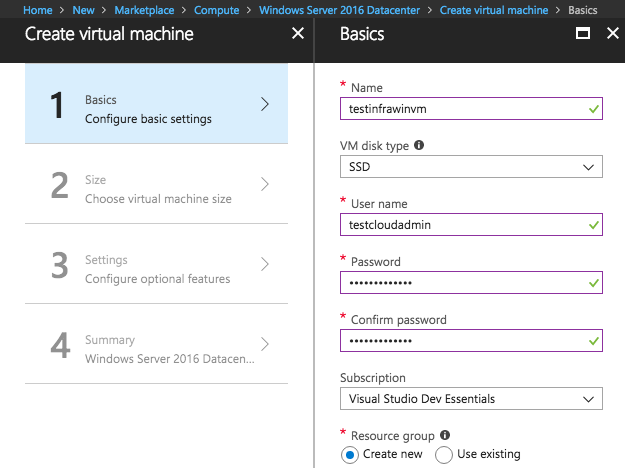


Choosing the appropriate deployment model to start deploying a windows VM

Then, you should fill in the necessary information in the **Basics** blade. The VM Disk type will affect the proposed pricing plan in the next step, which lets users choose the appropriate size of the VM. Azure provides the following types of disks:

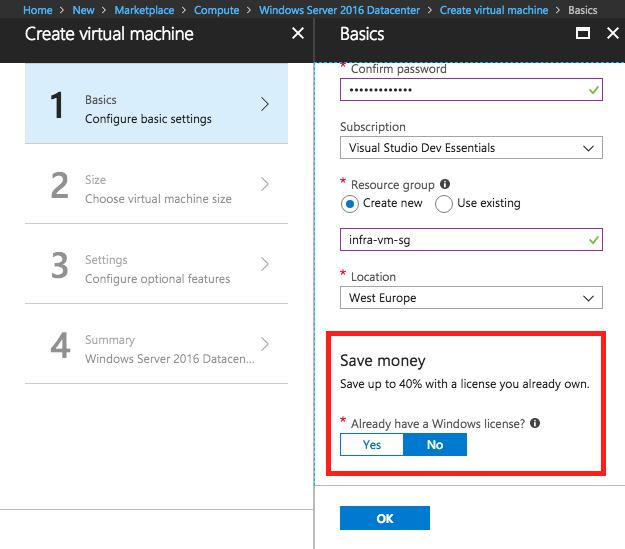
* The **Premium disks (SSD)** are backed by SSDs, provide consistent, low-latency performance, and are ideal for I/O-intensive applications and production workloads
* The **Standard disks (HDD)** are backed by magnetic drives and are designed for applications with infrequently accessed data

The **User name** and **Password** will be used to connect to the virtual machine. Usernames can be a maximum of 20-characters long, and the password must be at least 12-characters long and should have lower characters, upper characters, at least a digit, and a special character. While creating Azure VMs, users should pay attention to how to choose the most appropriate resource group, the subscription of the organization, and the location closest to your geography to reduce latency. The following screenshot is an example of the information in the **Basics** blade:



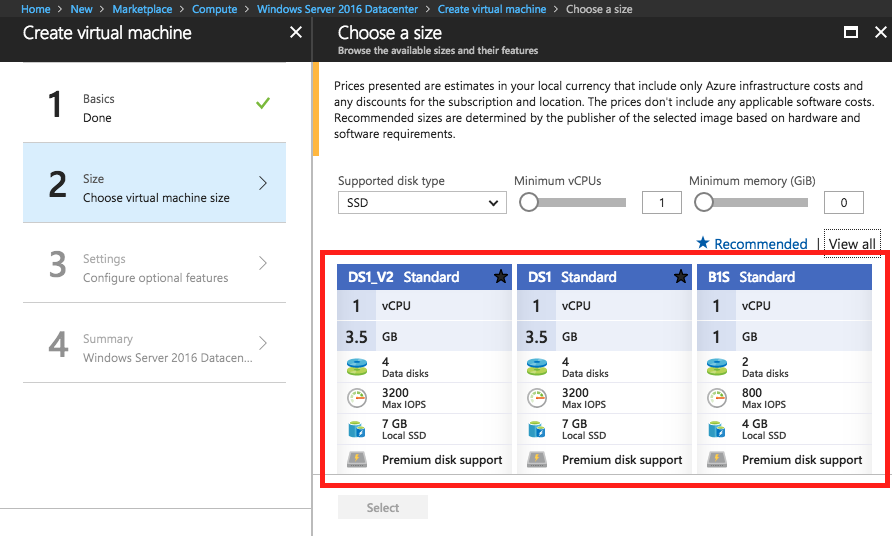
The information on the Basics blade

All Microsoft software installed in the Azure virtual machine environment must be licensed correctly. Microsoft Azure provides Azure Hybrid Benefit for the Windows server, which allows users to use on-premises Windows Server licenses and run Windows virtual machines on Azure at a reduced cost—this offer allows users to save up to 40% in costs. To obtain the benefits of Azure Hybrid, just confirm that you already have an on-premise license, as indicated in the following screenshot:



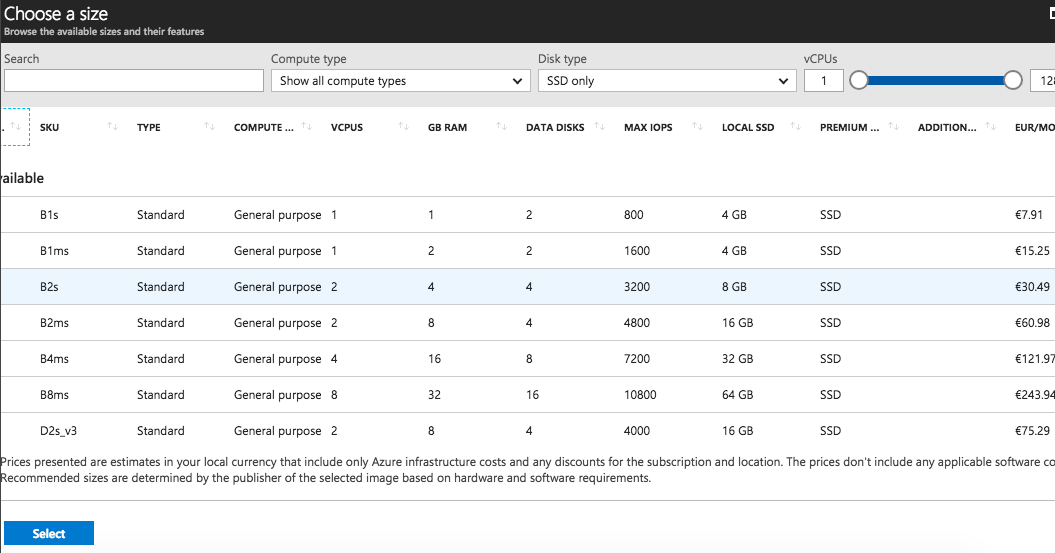
How to benefit from Azure Hybrid Benefit

Microsoft Azure provides different pricing solutions with a range of predefined configuration options that correspond to different VM sizes. The different VM sizes indicate the different numbers and speed of its processors, different amounts of memory, a maximum number of network adapters or data disks that users can attach to it, and the maximum size of a temporary disk. As shown in the following screenshot, users should choose an initial VM size while deploying a new VM in Azure:



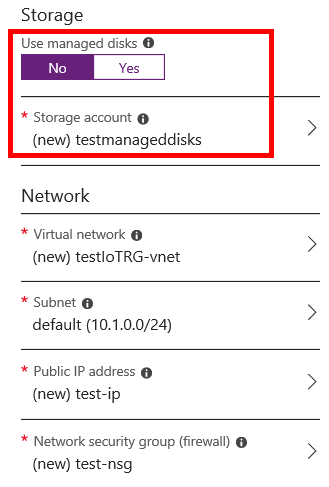
Choosing an Azure VM initial size

Here is a list view of the same:



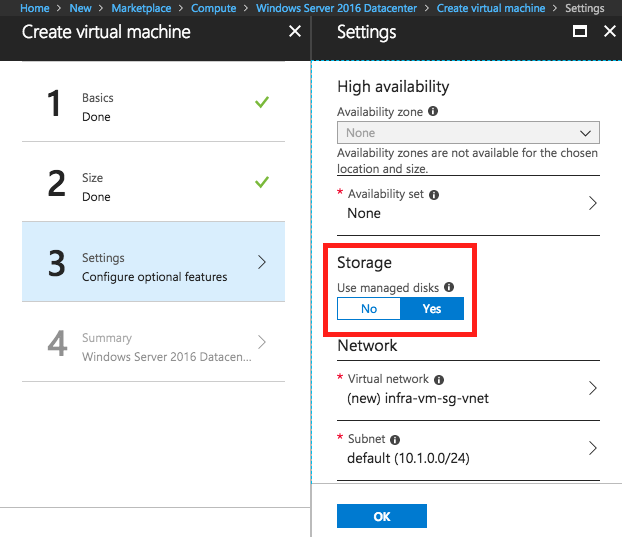
Choosing an Azure VM initial size (list view)

In the **settings** step, if you’re not going to use the managed disk, you should specify a storage account to store disks, as shown in the following screenshot:



Choosing storage account while using non-managed disks

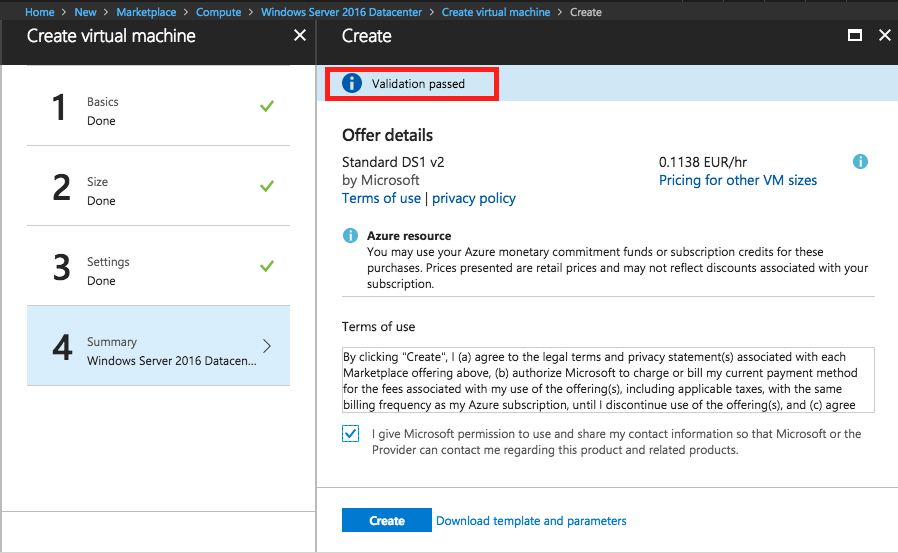
Select your storage option:



Choosing whether to use managed disks in Settings step

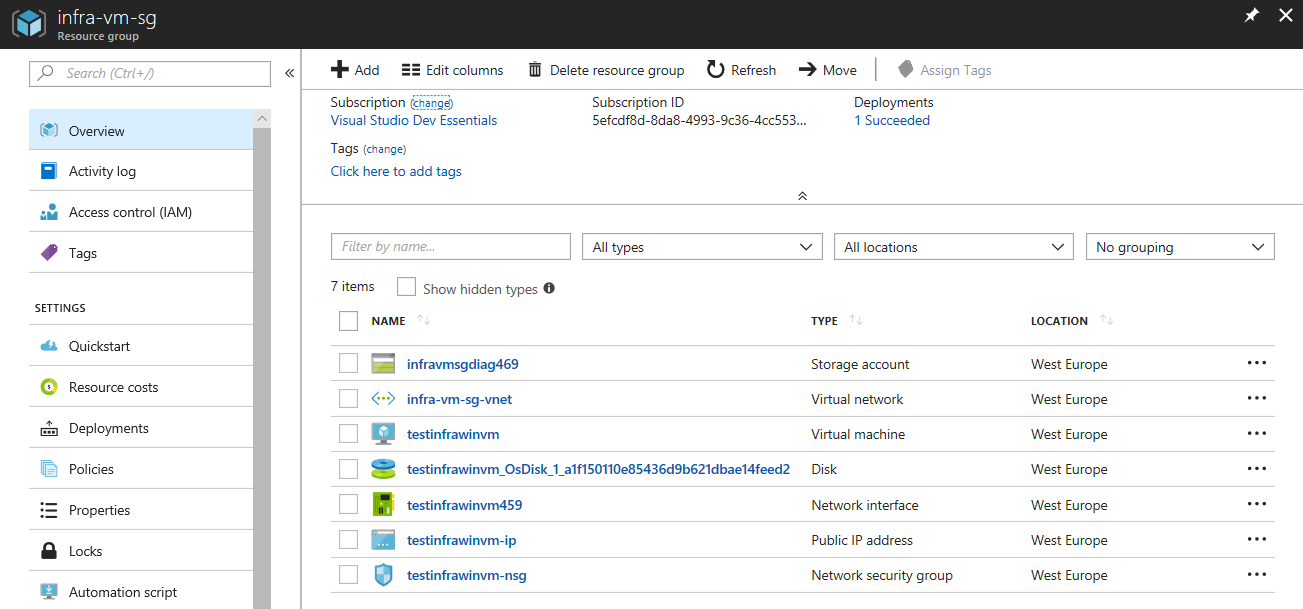
There are many options for creating a new Azure VM, such as **Virtual Network** (**VNet**), Subnet, and **Networks Security Group**(**NSG**). Microsoft Azure manages a default configuration for a predefined VM template that is ready to use. You can change these options while creating a new VM, depending on your intentions and requirements.

After filling in all the necessary information, Microsoft Azure will summarize and validate these details, as follows:



Azure VM deployment validation

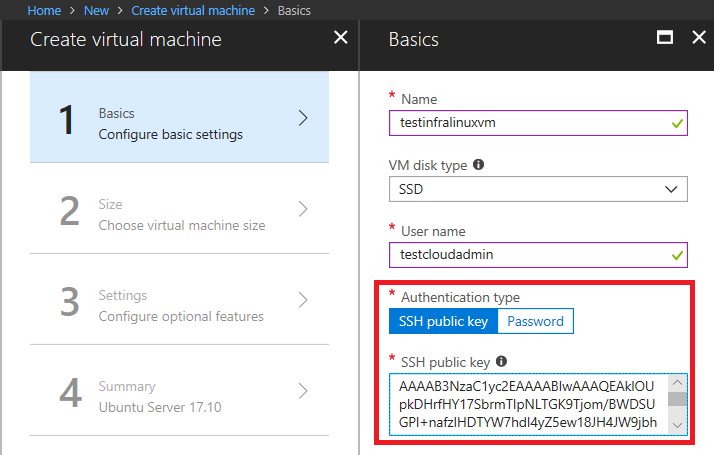
After deploying a Windows-based VM in Azure, the related resources such as vnet, nsg, and NIC are shown:



Related resources after a successful Azure VM deployment

Creating an Azure VM with a Linux distribution image

While creating a Linux-based VM, Microsoft provides almost the same options. A little different from the approach to create a Windows-based VM, a Linux-based VM is the authentication type, which means that Azure allows users to choose between the password-based and SSH public key–based authentication types while creating Linux-based Azure VMs, as follows:



Choose between the password-based and SSH public key–based authentication

Users should provide an RSA public key in the single line format (starting with ssh-rsa) or multiline PEM format (the multiline SSH key must begin with ---- BEGIN SSH2 PUBLIC KEY ---- and end with ---- END SSH2 PUBLIC KEY ----). You can generate SSH keys using ssh-keygen on Linux and macOS X, or PuTTYGen on Windows.

If you're using macOS or Linux, go to the following link to create your RSA key:

<https://docs.microsoft.com/en-us/azure/virtual-machines/linux/mac-create-ssh-keys>

If you're a Windows users, don't worry, go to the following link to get your RSA public key:

<https://docs.microsoft.com/en-us/azure/virtual-machines/linux/ssh-from-windows>

Creating Azure VMs via Azure PowerShell

To create an Azure VM using a Windows image, for example, Windows Server 2016 image via Azure PowerShell, we should configure the login and subscription that we will use in Azure. Note that, as we explained in [Chapter01](https://www.packtpub.com/mapt/book/virtualization_and_cloud/9781789137958/1), **Introduction to Cloud Computing**, there are two different deployment models in Azure: the classic deployment model, and the ARM model, for the Azure PowerShell. We will use the Azure Service Manager to deploy resources in Azure with the classic deployment model and the Azure Resource Manager to deploy resources in Azure with the ARM deployment model. That’s why all the PowerShell commands in the ARM model are prefixed with RM.

Before proceeding we need to import the subscription by using the command :

Import-PublishSettingsFile and install it in your local machine.

Let's begin now:

1. To sign in to Azure via Azure PowerShell or Windows PowerShell ISE, use the following command:

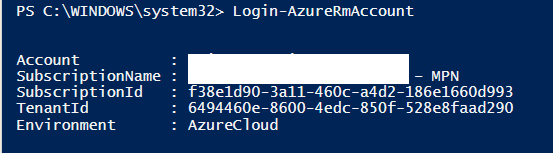
Copy

**Login-AzureRmAccount**

**Now Connect-AzureRMaccount**

Running the preceding command will display a popup to let you put in your Azure account name and password to pursue the authentication in Azure.

1. After logging into Azure with Azure PowerShell successfully, you’ll get the following output:



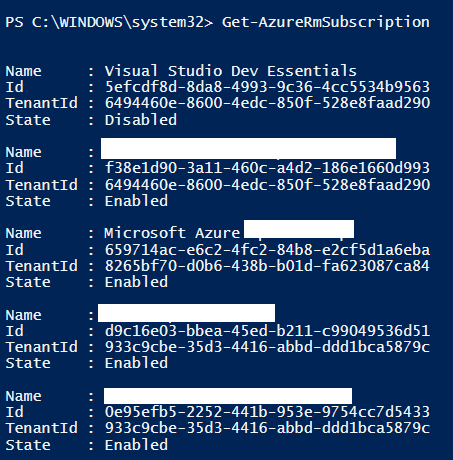
Output after login in Azure with Azure PowerShell successfully

1. Now, to get the list of Azure subscriptions associated with your account, use the following command:

Copy

**Get-AzureRmSubscription**

You will get the following output after executing the preceding command:



List of all subscriptions in the current tenant

1. If it is not the target subscription, use the following command to choose the target subscription:

Copy

**Select-AzureRmSubscription -SubscriptionName "<subscription name>"**

1. To set the subscription context, using the following command:

Copy

**Set-AzureRmContext -SubscriptionId "<subscription id>"**

1. To start the deployment of the Windows image, we should collect the related information regarding the deployment:
   1. The virtual network and its subnet
   2. Public IP address (optional)
   3. Network interface card (NIC)
   4. NSG with a rule allowing inbound RDP traffic (open inbound traffic for the port 3389 of your Azure VM)
   5. OS admin credentials (it is recommended to store them in a variable)
2. Then, create the Azure VM using the following command:

Copy

**New-AzureRmVm**

**-ResourceGroupName "testinfra70533rg"**

**-Name "testinfra"**

**-Location "WEST Europe"**

**-VirtualNetworkName "testinfra70533Vnet"**

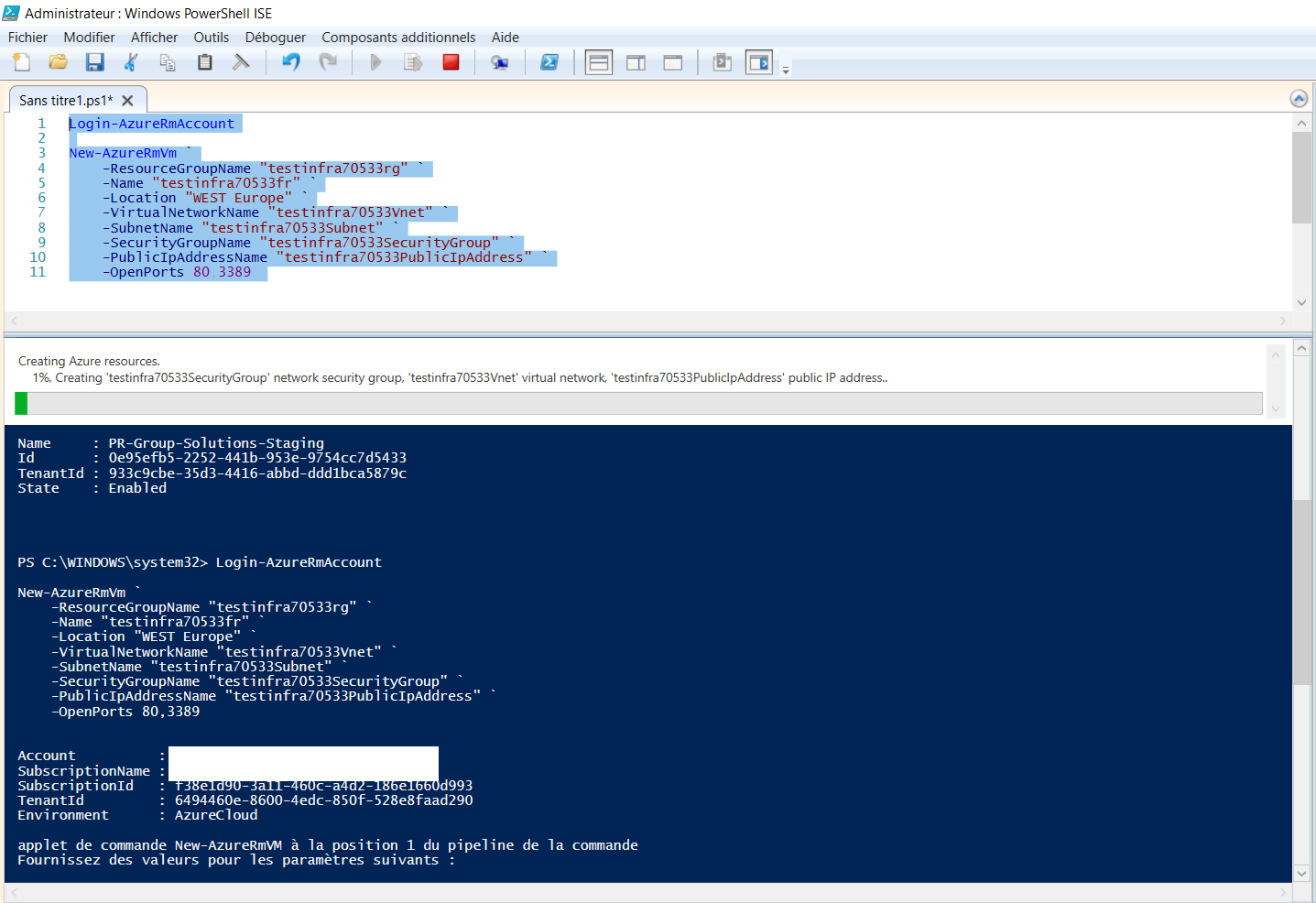
**-SubnetName "testinfra70533Subnet"**

**-SecurityGroupName "testinfra70533SecurityGroup"**

**-PublicIpAddressName "testinfra70533PublicIpAddress"**

**-OpenPorts 80,3389**

After authentication, it will start to create resources in Azure, as shown in the following screenshot:



Creating Azure VM via Azure PowerShell

Creating Azure VMs via an Azure CLI

To create an Azure VM using a Windows image, for example, a Windows Server 2016 image via the Azure CLI, similar to Azure PowerShell, we should configure the login and subscription that we will use in Azure. You can use the following command:

Copy

**az login**

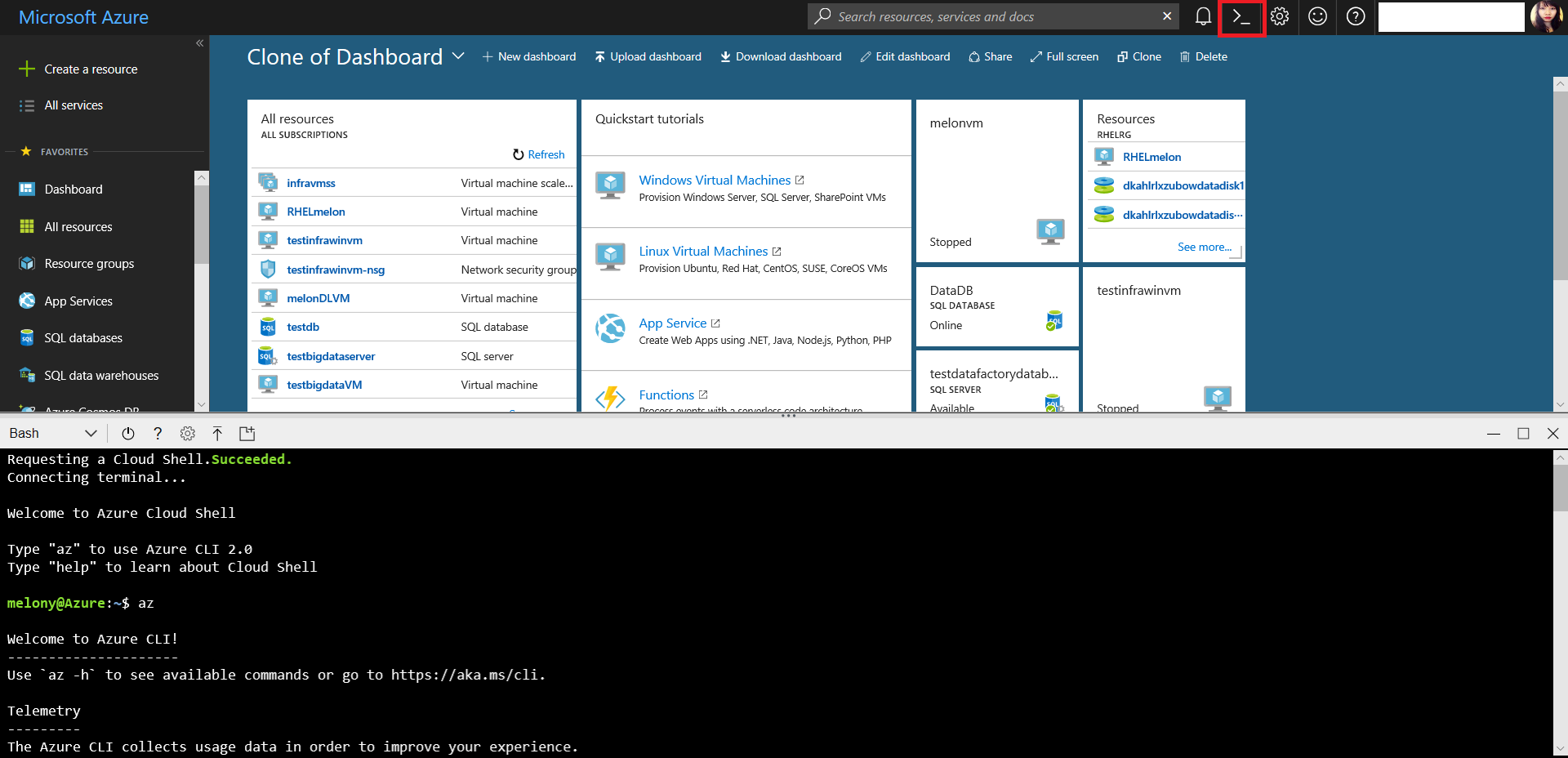
Use the following command to set your subscription, and where you want, to deploy your Azure VM:

Copy

**az account set –subscription <subscription name>**

You can use Azure CLI and follow the steps in [Chapter 02](https://www.packtpub.com/mapt/book/virtualization_and_cloud/9781789137958/2), **Overview of Microsoft Azure**, to install the Azure CLI on your PC. A great way to access Azure CLI is to use Azure Cloud Shell, which will let you always work with the latest Azure CLI commands without worrying about installing updates.

Launching the Cloud Shell via Azure Portal, as shown in the following screenshot:



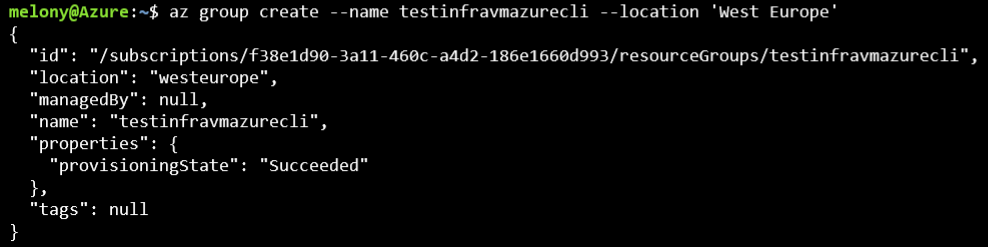
Launching a Cloud Shell via Azure Portal

After launching Cloud Shell successfully, create a resource group using the following command:

Copy

**az group create --name <resource group name> --location <Azure region>**

Specify the resource group that will host the Azure VM, its location following your intention. The example output will look like what's shown here in the following screenshot:



Creating resource group via Azure CLI

To create the Azure VM, use the following command:

Copy

**az vm create --resource-group <resource group name> --name <VM name> --image <Azure Marketplace image> --generate-ssh-keys**

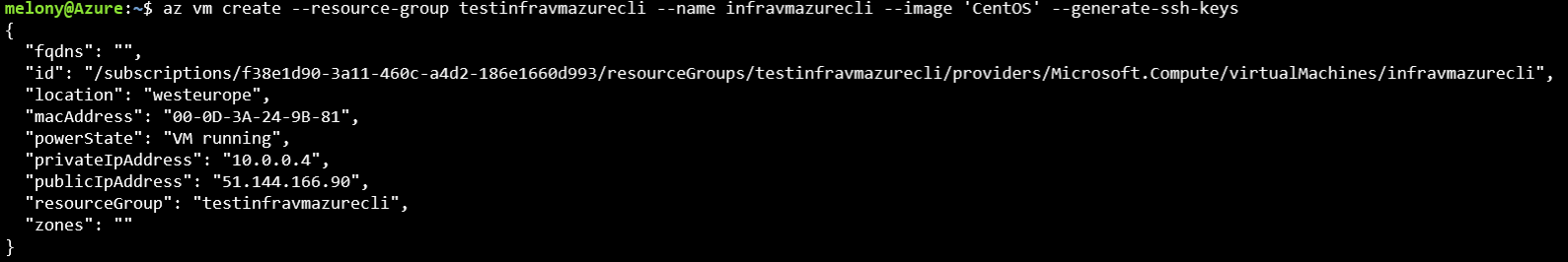
Currently, the valid images contain CentOS, CoreOS, Debian, openSUSE-Leap, RHEL, SLES, UbuntuLTS, Win2016Datacenter, Win2012R2Datacenter, Win2012Datacenter, and Win2008R2SP1.

The following is an example command I launched to create a Azure VM by using the Azure CLI:

Copy

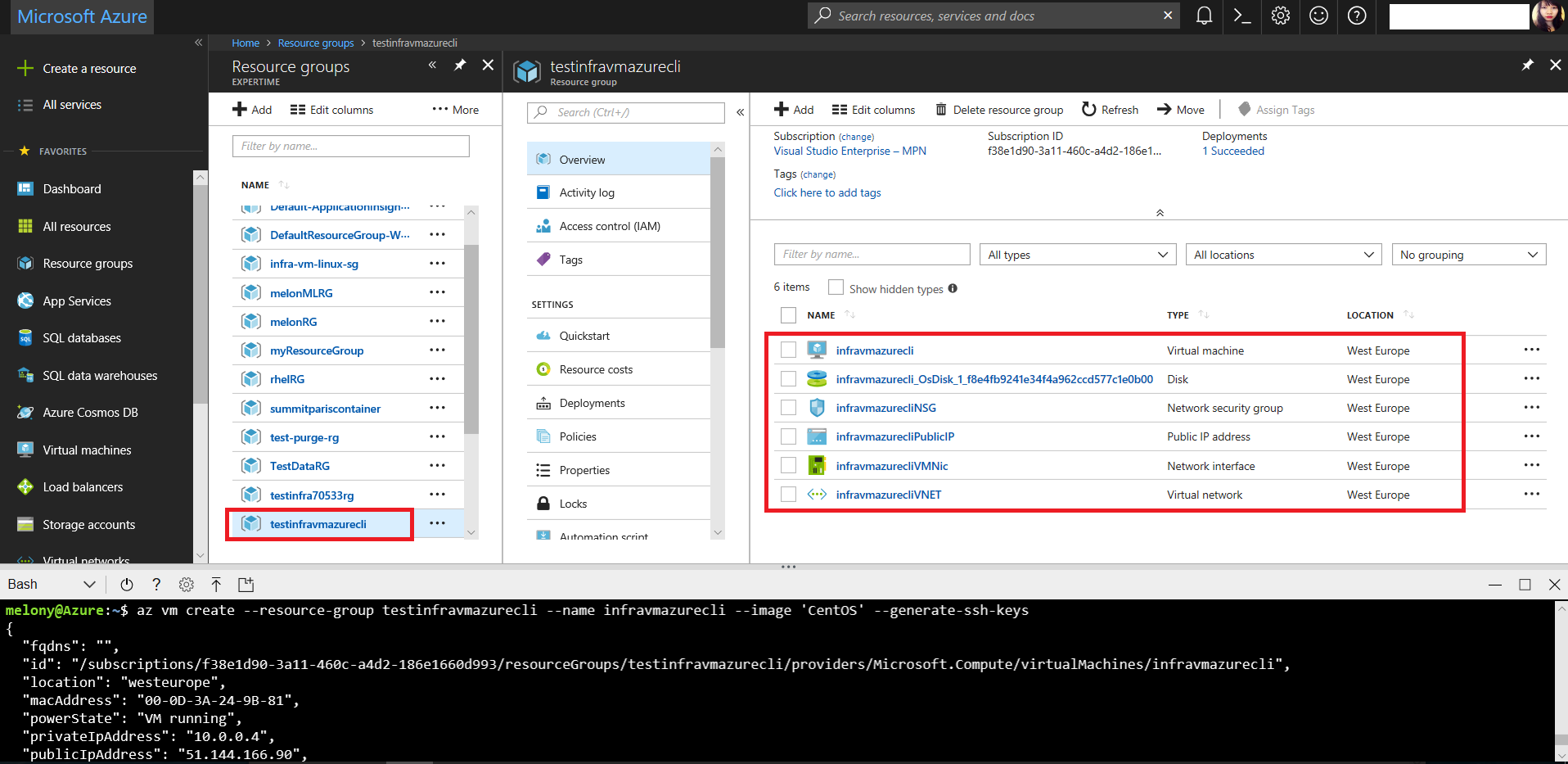
**az vm create --resource-group testinfravmazurecli --name infravmazurecli --image 'CentOS' --generate-ssh-keys**

The output of the preceding commands will be as follows, which means we have created a Linux VM via Azure CLI successfully:



The information from Azure while creating successfully

When returning to Azure Portal, we can find the resources that have been deployed in Azure successfully, as shown in the following screenshot:



Deploying Azure VM via Azure CLI successfully

Creating Azure VMs via the ARM template

You can also create VMs using Azure Resource Manager templates, which facilitate the deployment process. Microsoft Azure provides different ways to deploy ARM templates, such as Azure Portal, Visual Studio, and Visual Studio Code. This capability is provided by Azure Resource Manager, which makes it possible to use a formatted **JSON file** and include definitions of all the Azure Resource Manager resources that are part of the deployment.

An ARM Template contains $schema, contentVersion, parameters, variables, resources, and outputs. $schema is defined by Microsoft Azure which is mandatory; other elements, except contentVersion, and resources, are optional. The following is a sample template to create a Linux-based Azure VM using managed disks:

Copy

"resources": [

{

"apiVersion": "2018-06-01",

"type": "Microsoft.Network/publicIPAddresses",

"name": "myPublicIPAddress",

"location": "[resourceGroup().location]",

"properties": {

"publicIPAllocationMethod": "Dynamic",

"dnsSettings": {

"domainNameLabel": "testinfradns"

}

}

},

{

"apiVersion": "2018-04-01",

"type": "Microsoft.Network/virtualNetworks",

"name": "myVNet",

"location": "[resourceGroup().location]",

"properties": {

"addressSpace": { "addressPrefixes": [ "10.0.0.0/16" ] },

"subnets": [

{

"name": "mySubnet",

"properties": { "addressPrefix": "10.0.0.0/24" }

}

]

}

},

{

"apiVersion": "2018-04-01",

"type": "Microsoft.Network/networkInterfaces",

"name": "myNic",

"location": "[resourceGroup().location]",

"dependsOn": [

"[resourceId('Microsoft.Network/publicIPAddresses/', 'myPublicIPAddress')]",

"[resourceId('Microsoft.Network/virtualNetworks/', 'myVNet')]"

],

"properties": {

"ipConfigurations": [

{

"name": "ipconfig1",

"properties": {

"privateIPAllocationMethod": "Dynamic",

"publicIPAddress": { "id": "[resourceId('Microsoft.Network/publicIPAddresses','myPublicIPAddress')]" },

"subnet": { "id": "[variables('subnetRef')]" }

}

}

]

}

},

{

"apiVersion": "2018-04-01",

"type": "Microsoft.Compute/virtualMachines",

"name": "myVM",

"location": "[resourceGroup().location]",

"dependsOn": [

"[resourceId('Microsoft.Network/networkInterfaces/', 'myNic')]"

],

"properties": {

"hardwareProfile": { "vmSize": "Standard\_DS1" },

"osProfile": {

"computerName": "myVM",

"adminUsername": "[parameters('adminUsername')]",

"adminPassword": "[parameters('adminPassword')]"

},

"storageProfile": {

"imageReference": {

**"publisher": "OpenLogic",**

**"offer": "CentOS",**

**"sku": "7.4",**

**"version": "latest"**

},

"osDisk": {

"name": "myManagedOSDisk",

"caching": "ReadWrite",

"createOption": "FromImage"

}

},

"networkProfile": {

"networkInterfaces": [

{

"id": "[resourceId('Microsoft.Network/networkInterfaces','myNic')]"

}

]

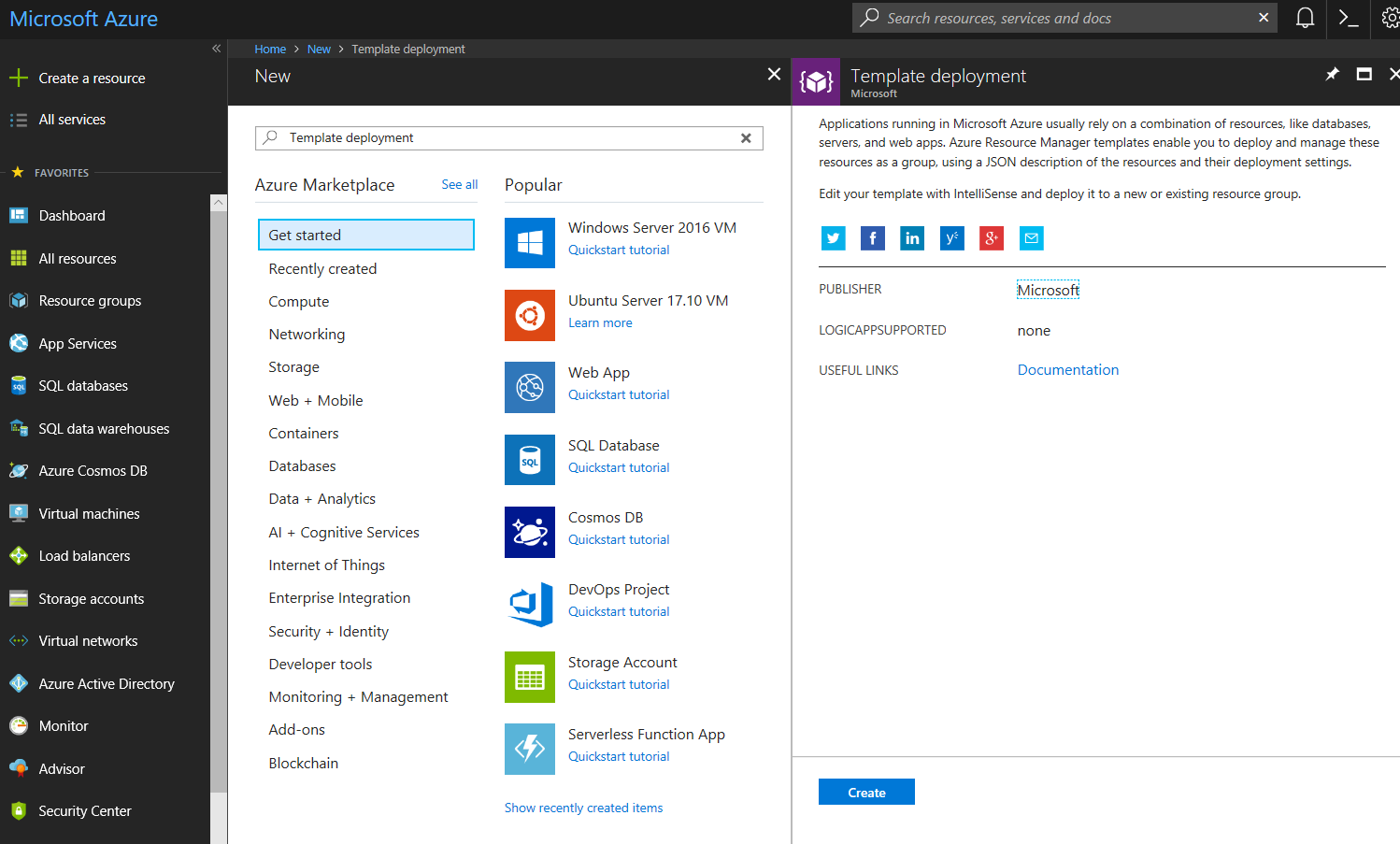
}

}

}

]

With the ARM Template, you can deploy it directly via Azure Portal. Start from **Create a resource** and then search **Template deployment** terms, and start to deploy ARM Template, as follows:



Template deployment via Azure Portal

To deploy an ARM Template via Azure PowerShell, use the following commands:

Copy

Login-AzureRmAccount

Select-AzureRmSubscription -SubscriptionName <yourSubscriptionName>

New-AzureRmResourceGroup -Name TestInfra70533RG -Location "West Europe"

New-AzureRmResourceGroupDeployment -Name TestInfra70533Deployment -ResourceGroupNameTestInfra70533RG

-TemplateFile c:\TestInfra70533Templates\storage.json -storageAccountType Standard\_GRS

To deploy an ARM Template via Azure CLI, use the following commands:

Copy

az login

az group create --name TestInfra70533RG --location "West Europe"

az group deployment create \

--name TestInfra70533Deployment \

--resource-group TestInfra70533RG \

--template-file storage.json \

--parameters storageAccountType=Standard\_GRS