**Q2 - SCENARIO**

Macro Life, a healthcare company has recently setup the entire Network and Infrastructure on Azure.

The infrastructure has different components such as Virtual N/W, Subnets, NIC, IPs, NSG etc.

The IT team currently has developed PowerShell scripts to deploy each component where all the properties of each resource is set using PowerShell commands.

The business has realized that the PowerShell scripts are growing over period of time and difficult to handover when new admin onboards in the IT.

The IT team has now decided to move to ARM based deployment of all resources to Azure.

All the passwords are stored in a Azure Service known as key Vault. The deployments needs to be automated using Azure DevOps using IaC(Infrastructure as Code).

*1) What are different artifacts you need to create - name of the artifacts and its purpose*

*2) List the tools you will to create and store the ARM templates.*

*3) Explain the process and steps to create automated deployment pipeline.*

*4) Create a sample ARM template you will use to deploy a Windows VM of any size*

*5) Explain how will you access the password stored in Key Vault and use it as Admin Password in the VM ARM template.*

Answer :

1. The different artifacts which will be required for the deployment of all the resources are :-
2. **Azuredeploy.json** –

This is the configuration file which have all the specification of the resource deployment which acts as the Infrastructure as code which defines the infra which needs to be deployed .This is a template is in JSON format which defines the infrastructure and configuration to be provisioned .

The template has following details:

**Parameters** – Provide the value during deployment that allow same template to be used multiple time

**Variables** – We can define the values which can be reused in our template

**User defined function** – Customized function can be created for the template

**Resources** – we can define the resources which needs to be deployed

**Output** – Return the values from the deployed resource.

**b) Azuredeploy.parameter.json:**

We can provide the parameter different values enhance the reusability of the of template for different environment.

1. **Keyvault.json :**

Which allows to domain join credential and local admin credential where the configuration is set for template deployment and the values of tenant ID ,Object ID for the user ,Service principal for the Azure AD for the vault .Properties are enabled for the deployment ,template deployment and all the access policies are stated .

1. **Keyvaultparameter.json**

Where all the parameters and the secret objects are passed for the Key vault .

1. **The tools to create and store the ARM templates**

**Visual Studio Code** with the Resource Manager Tools as Extension can be used for creation of the arm template.

For storing the arm templates we can Sign in to the **Azure DevOps** and create New Project and specify the name :

* We need to configure the Azure repos which will have the JSON codes for the pipeline and we can import the JSON code from the github repository .
* In azure devops we can import the github account in Azure Repos for storing all the templates post authorization to our source code repository .
* The process and steps to create automated deployment pipeline.

**In Azure DevOps we create the build pipeline**

Select the Azure Repos Git as repository and select the corresponding Repository and Branch

Select the Empty Job

New Build pipeline and Agent will be created and we can use the task for Azure Resource group deployment

* Input the information for the task

Name

Azure Subscription

Action – Create the Resource Group

Resource Group Creation

Location for the resource deployment

Template – Select the JSON template code for the deployment and also the parameters file to be used for the resource deployment.

Once all the tasks are defined and saved, we can run the build pipeline for provisioning the resources in Azure Platform.

**4. A sample ARM template to deploy a Windows VM**

{

"$schema": "https://schema.management.azure.com/deploymentTemplate.json#",

"contentVersion": "1.0..0",

"parameters": {

"adminUsername": {

"type": "string",

"metadata": {

"description": "Username of Virtual Machine."

}

},

"adminPassword": {

"type": "securestring",

"minLength": 7,

"metadata": {

"description": "Password Virtual Machine."

}

},

"dnsLabelPrefix": {

"type": "string",

"defaultValue": "[toLower(concat(parameters('vmName'),'-', uniqueString(resourceGroup().id, parameters('vmName'))))]",

"metadata": {

"description": "Unique DNS Name for the Public IP used to access the Virtual Machine."

}

},

"publicIpName": {

"type": "string",

"defaultValue": "myPublicIP",

"metadata": {

"description": "Name for the Public IP used to access the Virtual Machine."

}

},

"publicIPAllocationMethod": {

"type": "string",

"defaultValue": "Dynamic",

"allowedValues": [

"Dynamic",

"Static"

],

"metadata": {

"description": "Allocation method for the Public IP used to access the Virtual Machine."

}

},

"publicIpSku": {

"type": "string",

"defaultValue": "Basic",

"allowedValues": [

"Basic",

"Standard"

],

"metadata": {

"description": "SKU for the Public IP used to access the Virtual Machine."

}

},

"OSVersion": {

"type": "string",

"defaultValue": "2019-Datacenter",

],

"metadata": {

"description": " Windows version"

}

},

"vmSize": {

"type": "string",

"defaultValue": "Standard\_D2\_v3",

"metadata": {

"description": "Size virtual machine."

}

},

"location": {

"type": "string",

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

"metadata": {

"description": "Location for all resources."

}

},

"vmName": {

"type": "string",

"defaultValue": "simple-vm",

"metadata": {

"description": "Name of the virtual machine."

}

}

},

"resources": [

{

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

"apiVersion": "2020-06-01",

"name": "[parameters('vmName')]",

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

"dependsOn": [

"[resourceId('Microsoft.Storage/storageAccounts', variables('storageAccountName'))]",

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

],

"properties": {

"hardwareProfile": {

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

},

"osProfile": {

"computerName": "[parameters('vmName')]",

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

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

},

"storageProfile": {

"imageReference": {

"publisher": "MicrosoftWindowsServer",

"offer": "WindowsServer",

"sku": "[parameters('OSVersion')]",

"version": "latest"

},

"osDisk": {

"createOption": "FromImage",

"managedDisk": {

"storageAccountType": "StandardSSD\_LRS"

}

},

"dataDisks": [

{

"diskSizeGB": 1023,

"lun": 0,

"createOption": "Empty"

}

]

},

"networkProfile": {

"networkInterfaces": [

{

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

}

]

},

"diagnosticsProfile": {

"bootDiagnostics": {

"enabled": true,

"storageUri": "[reference(resourceId('Microsoft.Storage/storageAccounts', variables('storageAccountName'))).primaryEndpoints.blob]"

}

}

}

}

],

"outputs": {

"hostname": {

"type": "string",

"value": "[reference(parameters('publicIPName')).dnsSettings.fqdn]"

}

}

}

Paramaeter.json

{

"$schema": "https://schema.management.azure.com/schemas/deploymentParameters.json#",

"contentVersion": "1.0.0.0",

"parameters": {

"adminUsername": {

"value": "GEN-UNIQUE"

},

"adminPassword": {

"value": "GEN-PASSWORD"

},

"dnsLabelPrefix": {

"value": "GEN-UNIQUE"

}

}

}

5. **To access the password stored in Key Vault and use it as Admin Password in the VM ARM template.**

Once the key vault and the secret is added. We set the access policy to provide access to the user under the context that the template will be deployed and the user id is being added .

In advanced access policy we need to allow the access that the key vault can be accessed from ARM template deployment

So one of the parameter file in **deployment.json** file is Admin password and we need to use the key vault to pass this value.

In parameter file we need to pass the reference of the key vault value :

**Key value Id (Resource ID )**

**Name of the secret in Azure Key vault**