Excercises

Contents

[1. Prerequisites 3](#_Toc517204969)

[2. Clone Azure Workshop Git Repository on Github 3](#_Toc517204970)

[3. Create an Azure Resource Group for the VNET 4](#_Toc517204971)

[4. Create an Azure VNET 4](#_Toc517204972)

[5. Create an Azure Resource Group for the Solution 4](#_Toc517204973)

[6. Create an Azure Storage Account 5](#_Toc517204974)

[7. Create Containers in Azure Storage Account 5](#_Toc517204975)

[8. Upload DSC ZIP Files to Azure Storage Containers 5](#_Toc517204976)

[9. Create an Azure Keyvault 6](#_Toc517204977)

[10. Add Default Windows Admin Password to Keyvault 6](#_Toc517204978)

[11. Get the Keyvault Resource ID 7](#_Toc517204979)

[12. Create Log Analytics Workspace 7](#_Toc517204980)

[13. Update Parameter File Domain Controller Deployment 7](#_Toc517204981)

[14. Deploy Domain Controller Template 9](#_Toc517204982)

[15. Update DNS Settings VNET 10](#_Toc517204983)

[16. Update Parameter File Windows Admin Center Deployment 10](#_Toc517204984)

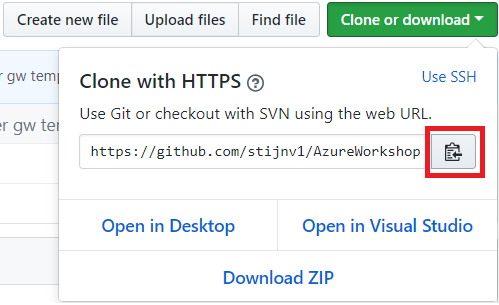
[17. Deploy Windows Admin Center Template 10](#_Toc517204985)



# Prerequisites

* Install Git for Windows: <https://git-scm.com/download/win>
* Install VS Code: <https://code.visualstudio.com/>
* Install VS Code Extensions :
  + Azure Account
  + Azure Resource Manager Tools
  + Powershell
* Install latest AzureRM Powershell module. If you have Azure SDK installed (Visual Studio 201\*), please uninstall the msi package of AzureRM Powershell first.  
  Execute following command to install the latest version of the modules:  
  Install-Module AzureRM

# Clone Azure Workshop Git Repository on Github

* Go to following URL: <https://github.com/stijnv1/AzureWorkshop>
* Click ‘Clone or download’ in the repository and copy the URL by pressing the highlighted button in the screenshot below:  
    
  
* On your laptop/desktop, create following directory in the root C:\ :  
  AzureWorkshopRepo
* Open command prompt/powershell and navigate to the following directory:   
  cd C:\AzureWorkshopRepo
* Execute following command to clone the AzureWorkshop Github Repo:  
  git clone <URL you copy pasted from the Github repo site>

# Create an Azure Resource Group for the VNET

* Create an Azure Resource Group for your VNET. You can do this via the Azure Portal, or use AzureRM Powershell.  
  Following steps describe how to create the resource group via AzureRM Powershell:  
    
  New-AzureRMResourceGroup –Name RG-<your username>-Networking –Location westeurope

# Create an Azure VNET

* Create an Azure VNET using the Azure Portal or using the provided ARM template from the cloned AzureWorkshop Github repo. Create the VNET in the Azure Resource Group you created in step 3.  
    
  Use following settings for the VNET:
  + VNET Name: VNET-Workshop
  + VNET Address space: 10.0.0.0/16
  + Subnets:
    - Subnetname 1:
      * Name: SNET-Infra
      * CIDR: 10.0.1.0/24
    - Subnetname 2:
      * Name: SNET-DMZ
      * CIDR: 10.0.2.0/24
    - Subnetname 3:
      * Name: SNET-WebServers
      * CIDR: 10.0.3.0/24

Following steps describe how to deploy the VNET using the provided template:

New-AzureRMResourceGroupDeployment –Name vnetDeployment –ResourceGroupName <Name of the resourcegroup you created in step 3> -Mode Incremental -TemplateFile <path to the following file in the cloned github repo: ‘vnet-deploy.json’> -TemplateParameterFile <path to the following file in the cloned github repo: ‘vnet-deploy.vnet1.parameters.json’> -Verbose

# Create an Azure Resource Group for the Solution

* Create an Azure Resource Group for the solution. You can do this via the Azure Portal, or use AzureRM Powershell.  
  Following steps describe how to create the resource group via AzureRM Powershell:  
    
  New-AzureRMResourceGroup –Name RG-<your username>-IAAS –Location westeurope

# Create an Azure Storage Account

* This storage account is needed for the DSC scripts which are used by the ARM templates for the domain controller and windows IIS deployments.  
  You can create this storage account using the portal or by using following AzureRM Powershell command:  
    
  New-AzureRMStorageAccount –ResourceGroupName <resourcegroup name you created in step 5) –Name <a unique name in Azure for this account> -SkuName Standard\_LRS –Location westeurope –Kind Storage

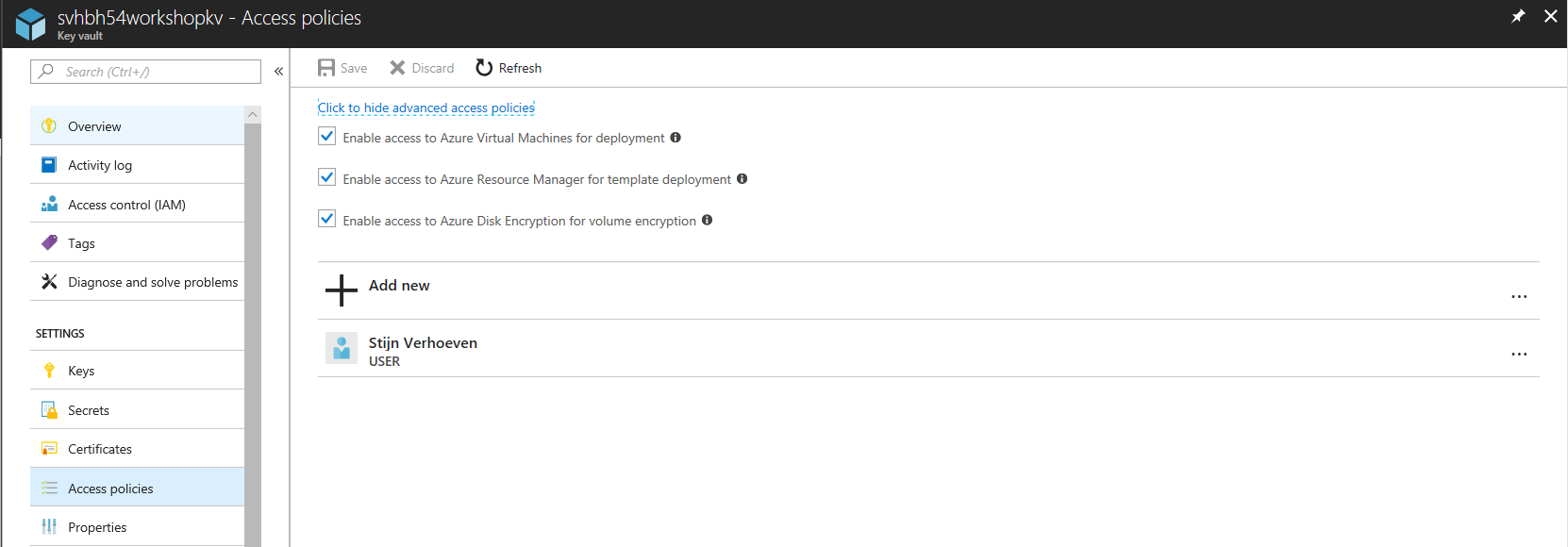
# Create Containers in Azure Storage Account

* Create following containers in the created storage account using the Storage Explorer available in the Azure Portal or using AzureRM Powershell.  
    
  $storageaccount = Get-AzureRMStorageAccount –ResourceGroupName < resourcegroup name you created in step 5) –Name <name of storageaccount created in step 6>  
  $ctx = $storageaccount.Context  
  New-AzureStorageContainer –Name dsc –Context $ctx  
  New-AzureStorageContainer –Name cse –Context $ctx

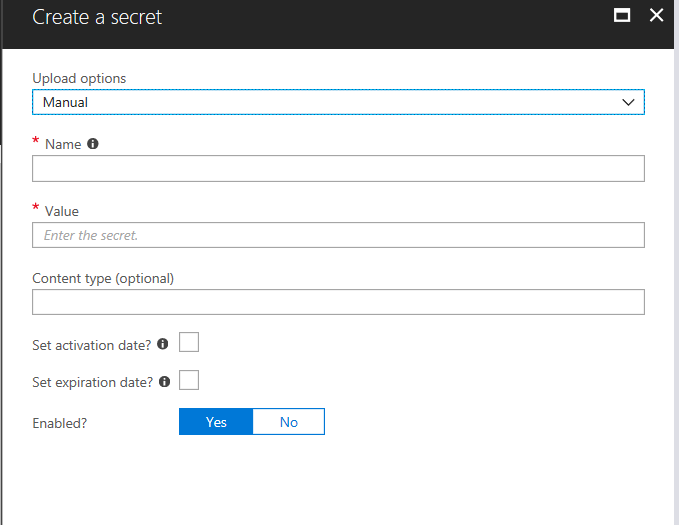
# Upload DSC ZIP Files to Azure Storage Containers

* Upload following ZIP file to the dsc container created in step 7:  
  C:\AzureWorkshopRepo\AzureWorkshop\ARM\_Templates\DomainController\DSC\CreateNewAD.ps1.zip
* Upload following powershell script to the cse container created in step 7:  
  C:\AzureWorkshopRepo\AzureWorkshop\ARM\_Templates\ WindowsAdminCenterGWServer \scripts\installWindowsAdminCenter.ps1
* You can use following powershell commands to perform the upload or use the Storage Explorer in the Azure Portal:  
    
  $storageaccount = Get-AzureRMStorageAccount –ResourceGroupName < resourcegroup name you created in step 5) –Name <name of storageaccount created in step 6>  
  $ctx = $storageaccount.Context  
  Set-AzureStorageBlobContent –File “C:\AzureWorkshopRepo\AzureWorkshop\ARM\_Templates\DomainController\DSC\CreateNewAD.ps1.zip” –Container “dsc” –Blob “CreateNewAD.ps1.zip” –Context $ctx  
  Set-AzureStorageBlobContent –File “C:\AzureWorkshopRepo\AzureWorkshop\ARM\_Templates\ WindowsAdminCenterGWServer \scripts\installWindowsAdminCenter.ps1” –Container “cse” –Blob “installWindowsAdminCenter.ps1” –Context $ctx

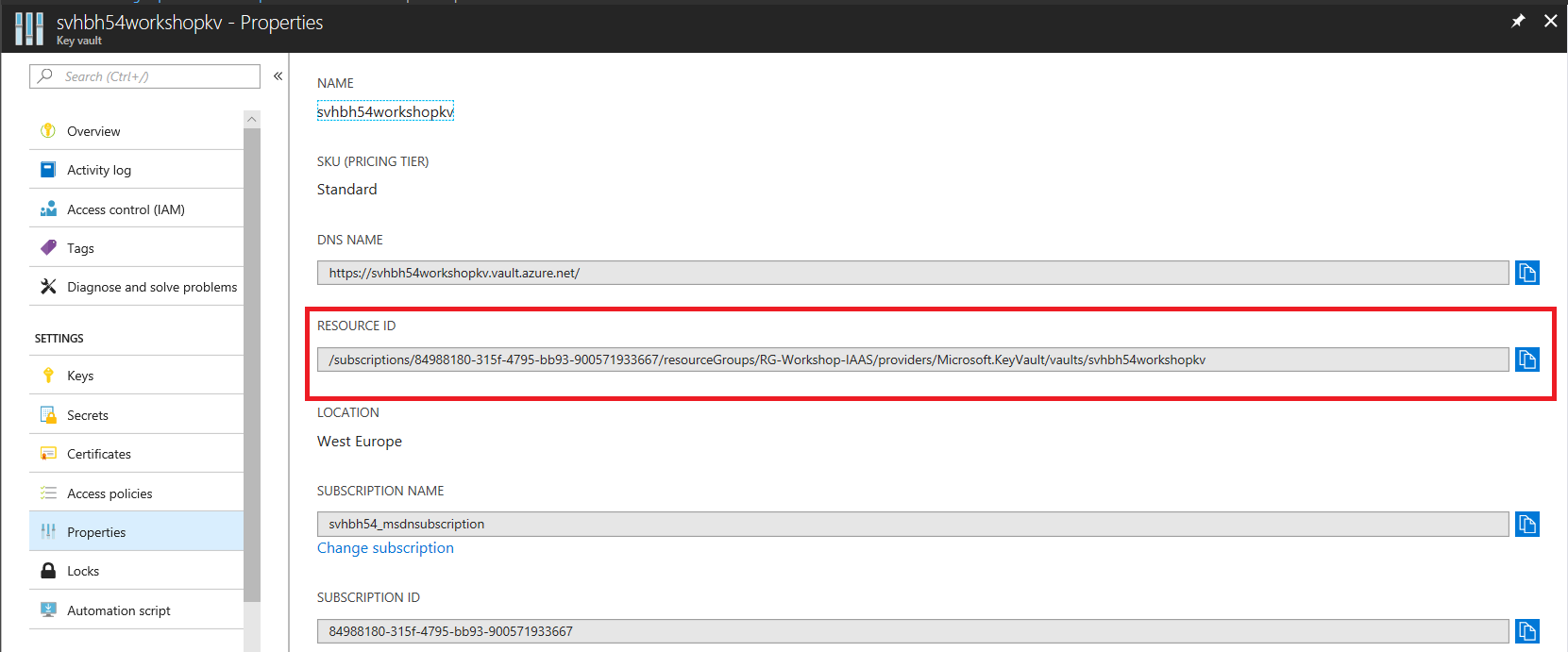
# Create an Azure Keyvault

* Create an Azure Keyvault using the portal or using following AzureRM Powershell command. If you go via the portal, make sure you enable following settings after the keyvault has been created:  
    
    
    
  New-AzureRmKeyVault -Name <unique name for keyvault> -ResourceGroupName <resource group you created in step 5> -Location westeurope -EnabledForDeployment -EnabledForTemplateDeployment -EnabledForDiskEncryption

# Add Default Windows Admin Password to Keyvault

* Open the keyvault you created in step 9 and add a secret which will contain the default administrator password for the VMs we will deploy in the next steps:  
    
  Go to ‘Secrets’ | click ‘Generate/Import’:  
    
    
    
  Give the secret a name (it is not required that this name is the same as the username you will define for the default windows administrator in the next steps)  
    
  In the Value field, enter the password you want to use.  
    
  Click Create

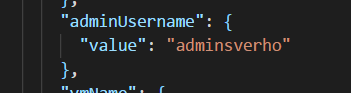
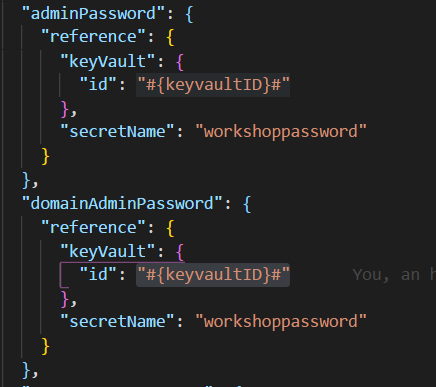
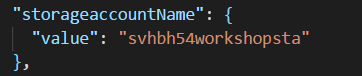
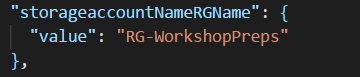
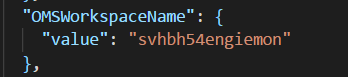
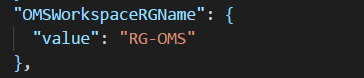
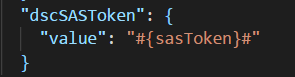
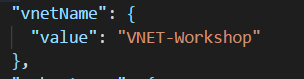
# Get the Keyvault Resource ID

* In the portal, go to the created Keyvault and go to ‘Properties’. Copy paste the following value in a notepad so you can quickly access it in the next steps:  
    
  

# Create Log Analytics Workspace

* Create a Log Analytics workspace using the portal or using following AzureRM Powershell command:  
    
  New-AzureRmOperationalInsightsWorkspace -Location westeurope -Name <unique name for the workspace> -Sku free -ResourceGroupName <resource group you created in step 5>

# Update Parameter File Domain Controller Deployment

* Go to following directory:  
  cd C:\AzureWorkshopRepo\AzureWorkshop\ARM\_Templates\DomainController\parameters
* Open the following file in this directory in VS Code:  
  domaincontroller-azuredeploy.parameters.json
* Update following parameters in this json file with the correct values:
  + adminUsername: give a username you want to use. This will be the default administrator username on the deployed Domain Controller:  
      
    
  + adminPassword and domainAdminPassword: copy paste the copied resource id of the keyvault (step 11) in the ‘id’ field (replace #{keyvaultID}# with the resource id):  
      
    
  + storageaccountName: give the name of the storage account you created in step 6  
      
    
  + storageaccountRGName: give the name of the resource group you created in step 5 (= the resource group in which you created the storage account)  
      
    
  + OMSWorkspaceName: give the name of the workspace you created in step 12  
      
    
  + OMSWorkspaceRGName: give the resource group in which you created the workspace  
      
    
  + dscSASToken: create a SAS token using the Azure Portal or use following powershell commands to generate one. Copy paste the generated token:  
      
    $storageaccount = Get-AzureRMStorageAccount –ResourceGroupName < resourcegroup name you created in step 5) –Name <name of storageaccount created in step 6>  
    $ctx = $storageaccount.Context  
    New-AzureStorageContainerSASToken –Container dsc –Permissions r –Context $ctx  
      
    
  + vnetName: change the value to the name of the VNET you created in step 4  
      
    
  + vnetRGName: change the value to the name of the resource group you created in step 3 (=the resource group which contains the VNET)

# Deploy Domain Controller Template

* Use following AzureRM powershell command to deploy the Domain Controller:  
    
  New-AzureRMResourceGroupDeployment –Name deployDC –ResourceGroupName <Name of the resourcegroup you created in step 5> -Mode Incremental -TemplateFile <path to the following file in the cloned github repo: ‘domaincontroller-azuredeploy.json’> -TemplateParameterFile <path to the following file in the cloned github repo: ‘domaincontroller-azuredeploy.parameters.json’> -Verbose
* After the deployment is finished, you can RDP into the created Domain Controller using the Public IP which is deployed and attached to the deployed DC Azure VM

# Update DNS Settings VNET

* In the portal , go to the VNET you created and change the DNS Servers settings to ‘custom’. Add following IP address:
  + 10.0.1.10

As an alternative, you can rerun the ARM template of the Azure VNET, but using another parameter file:  
  
New-AzureRMResourceGroupDeployment –Name vnetDeployment –ResourceGroupName <Name of the resourcegroup you created in step 3> -Mode Incremental -TemplateFile <path to the following file in the cloned github repo: ‘vnet-deploy.json’> -TemplateParameterFile <path to the following file in the cloned github repo: ‘vnet-deploy.vnet2.parameters.json’> -Verbose

# Update Parameter File Windows Admin Center Deployment

* Update the following parameters with the same values you used to deploy the domain controller VM:
  + adminUsername
  + vnetRGName
  + vnetName
  + storageaccountName
  + storageaccountRGName
  + OMSWorkspaceRGName
  + OMSWorkspaceName
* Change following parameter value to a unique value (for example, append your username to the dns label):
  + dnsLabelPrefix: example: srvwinadmincenter<your username>

# Deploy Windows Admin Center Template

* Use following AzureRM powershell command to deploy Windows Admin Center Azure VM incl Azure External Loadbalancer:  
    
  New-AzureRMResourceGroupDeployment –Name deployWinAdminCenter –ResourceGroupName <Name of the resourcegroup you created in step 5> -Mode Incremental -TemplateFile <path to the following file in the cloned github repo: ‘windowsadminCenterGW-deploy.json’> -TemplateParameterFile <path to the following file in the cloned github repo: ‘windowsadminCenterGW-deploy.parameters.json’> -Verbose
* After the deployment has finished, go to following URL: https://<dnsLabelPrefix value you used in step 15>.westeurope.cloudapp.azure.com
* When prompted for a user account, use following username and the password you inserted as a secret in the created keyvault in step 10:
  + workshop\<adminUsername parameter value you used in the parameter file of the domain controller deployment>