**Microsoft Certified Associate: Azure Architect Technologies AZ-303**

Course-End Project

For Caltech/Simpilearn

Michael Whitehead

May 25, 2021

**Course-End Project: Implement Azure IaaS**

**Project objective:**

The Rand Enterprises Corporation wants to test ARM template deployment to bring infrastructure as code into practice. They have decided to work on project RandEnt to verify the functionality.

The operations team at Rand decides to define the entire networking architecture using the ARM template, once it is in place they intend to create the storage account along with virtual machines housing their application.

**Background of the problem statement:**

Rand Enterprises works extensively on delivering Image-based content for their global audience in a secure way by avoiding Azure Storage account access to the public internet. The communication from the application in the VM to the Azure Storage account must take place via the internal network of Azure.

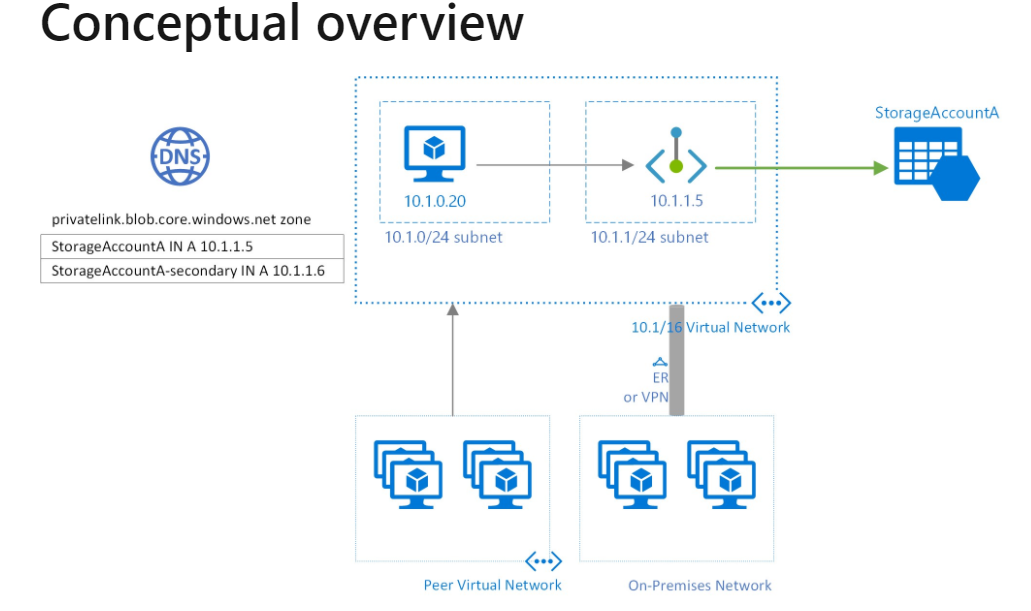
The expectation of the operation team is to rather than deploying resources in Azure independently, they should leverage Azure ARM templates to deploy and provision all resources in templatized format.

**Following requirements should be met:**

* Define the network
* Extend that with Compute & Storage
* Create the Storage account container for Images & configure service endpoints

Credits: Aman Puriaz of Simplilearn – For his help in getting me off to a good start.

Implementation:



Step 1: Download the ARM template and image files from the attached Google drive share path:

<https://drive.google.com/drive/folders/1q1GUZtE64k52u8Ve7qwzCSI9zrNQBqO1>

These files will be needed for the power shell scripts shown below.

Step 2: Create a Resource Group

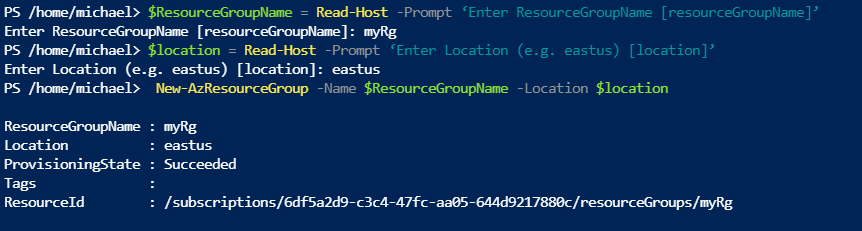
Step 2.1 Open your windows power shell as administrator and connect to you Azure Account by using the below command Connect-AzAccount if using local machine, or launch Azure Portal and Cloud Shell and select Power Shell.

Step 2.2 The following power shell commands prompt for ResourceGroupName and Location then creates a resource group:

PS > $ResourceGroupName = Read-Host -Prompt ‘Enter ResourceGroupName [resourceGroupName]’

$location = Read-Host -Prompt ‘Enter Location (e.g. eastus) [location]’

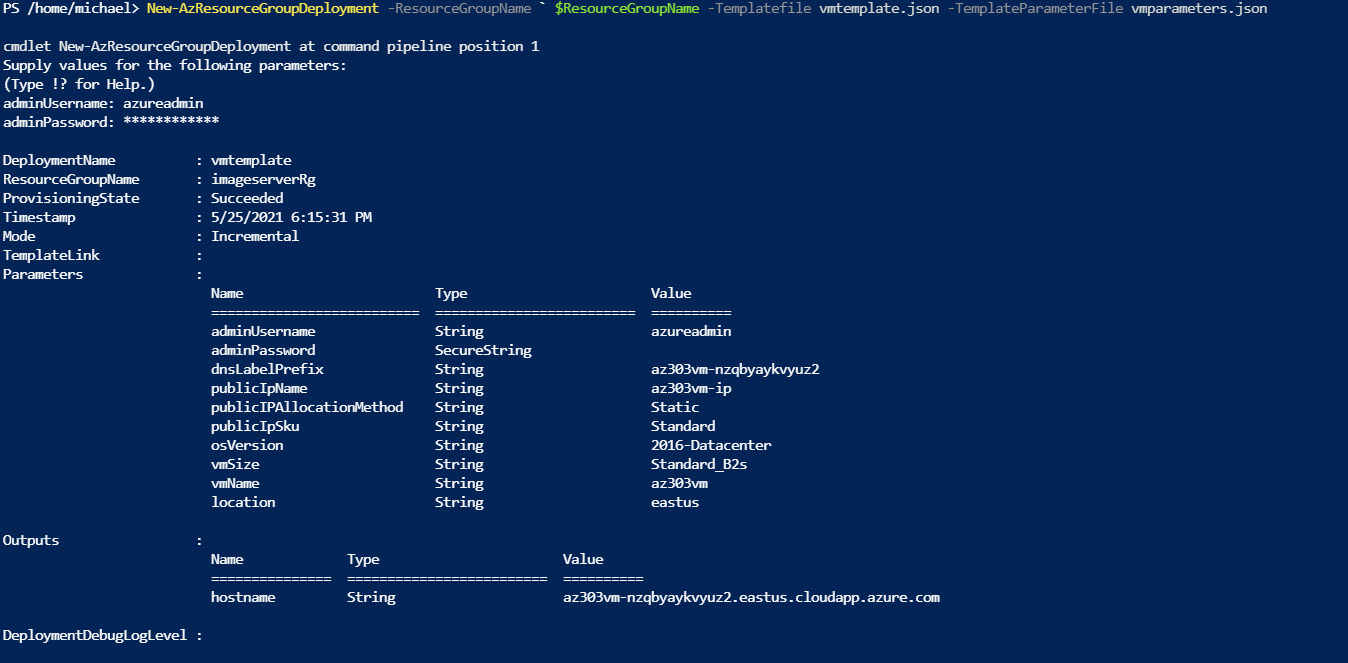
New-AzResourceGroup -Name $ResourceGroupName -Location $location



Step 3: Deploy VM Creation Template

PS > New-AzResourceGroupDeployment -ResourceGroupName `

$ResourceGroupName -Templatefile vmtemplate.json -TemplateParameterFile vmparameters.json



Step 4: Deploy Storage Template:

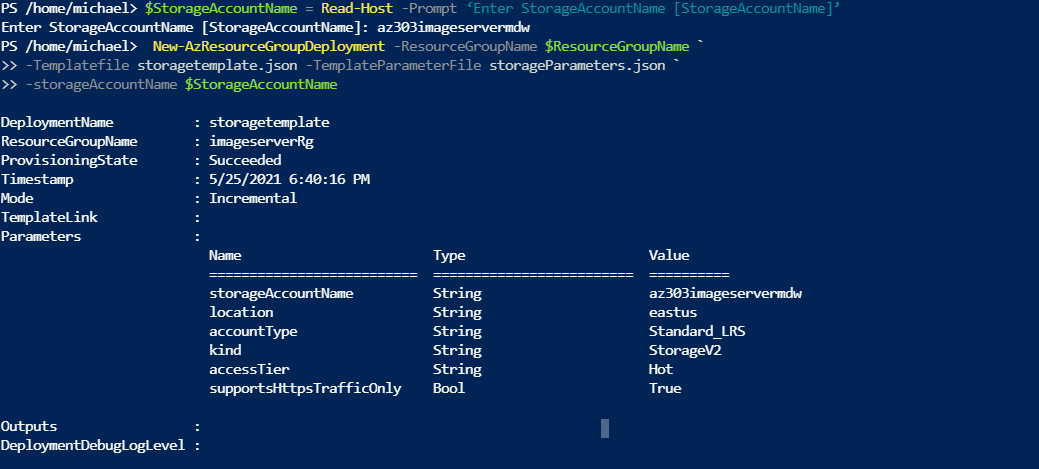
The following power shell commands prompt for StorageAccountName and create StorageAccount:

PS > $StorageAccountName = Read-Host -Prompt ‘Enter StorageAccountName [StorageAccountName]’

New-AzResourceGroupDeployment -ResourceGroupName $ResourceGroupName `

-Templatefile storagetemplate.json -TemplateParameterFile storageParameters.json `

-storageAccountName $StorageAccountName



Step 5: Deploy Storage Container:

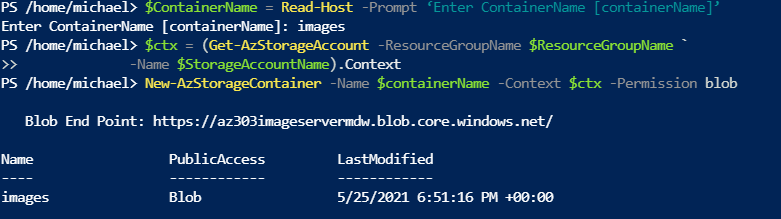
The following powershell commands prompt for ContainerName, get the Storage Context, and and create StorageAccount:

PS > $ContainerName = Read-Host -Prompt ‘Enter ContainerName [containerName]’

$ctx = (Get-AzStorageAccount -ResourceGroupName $ResourceGroupName `

-Name $StorageAccountName).Context

New-AzStorageContainer -Name $containerName -Context $ctx -Permission container



Note: Public Access Level Should be: Anonymous read access for container and blobs

Step 6: Load Test Files to Container:

PS > $StorageAccountName = ‘az303imageservermdw’

PS > $ResourceGroupName = ‘imageserverRg’

PS> $ctx = (Get-AzStorageAccount -ResourceGroupName $ResourceGroupName `

-Name $StorageAccountName).Context

Example:

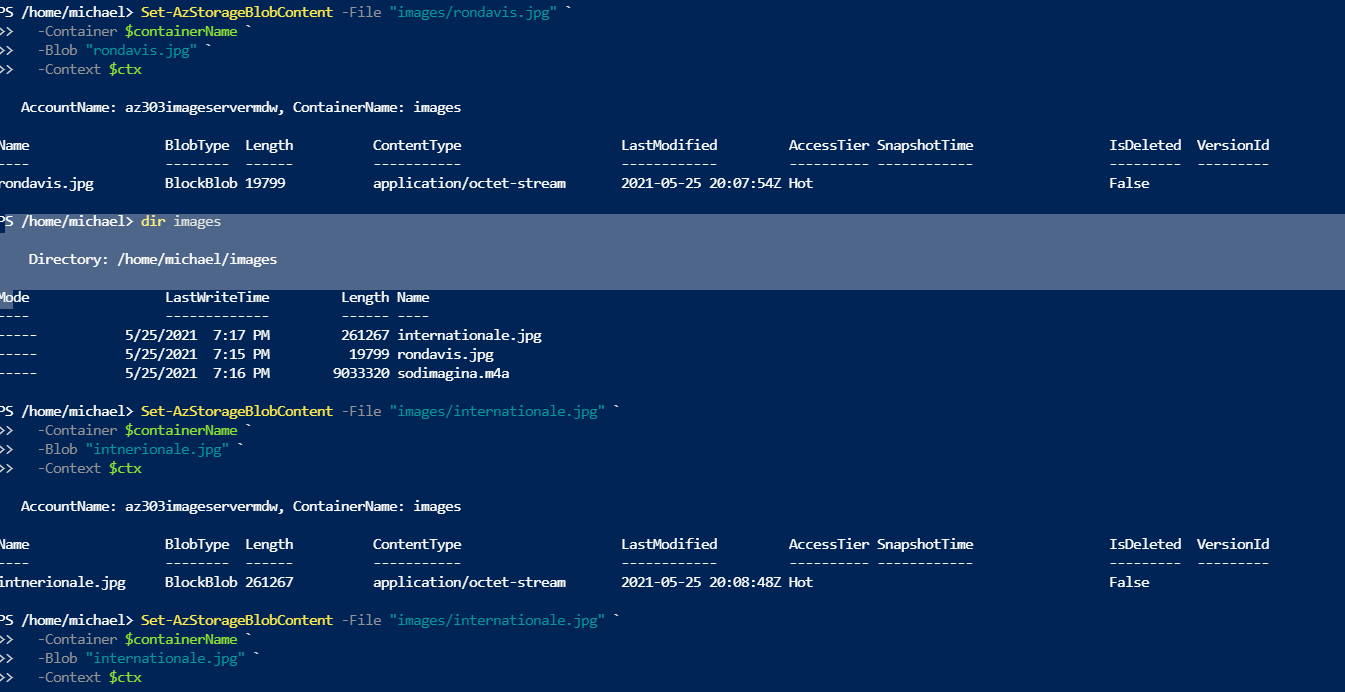
PS> Set-AzStorageBlobContent -File "images/rondavis.jpg" `

-Container $containerName `

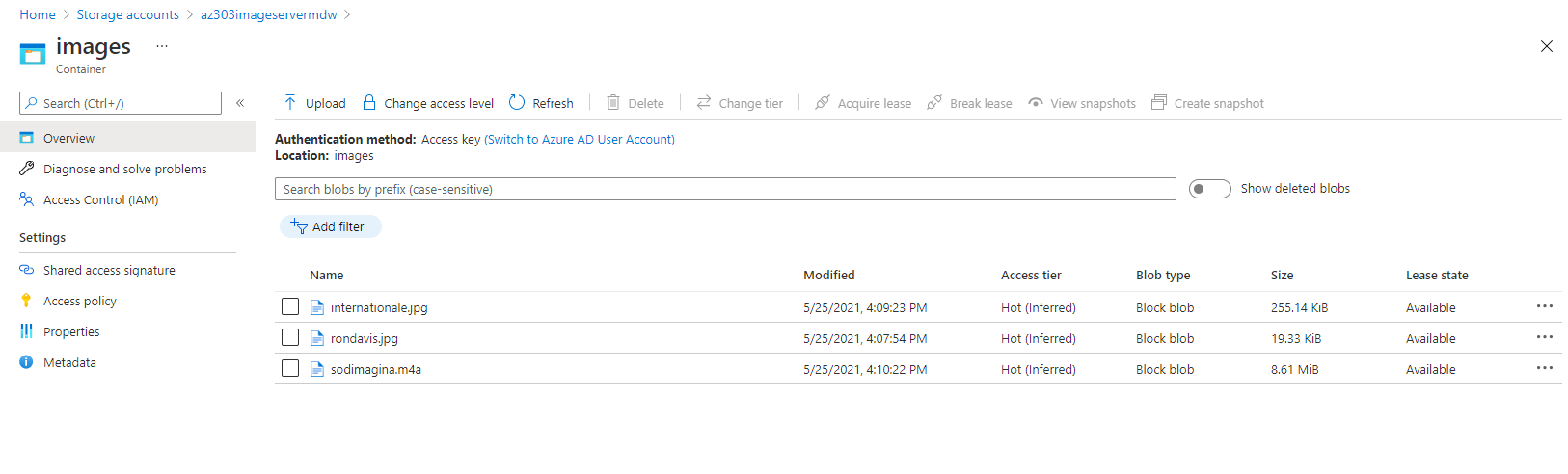
-Blob “rondavis.jpg” `

-Context $ctx `

-StandardBlobTier Hot



Step 7: Check that Test Images are uploaded successfully in the Storage Container:



Step 8: The VM and Storage Account are in place but should to be tested. The resources could be tested by using Storage Explorer in Azure Portal but using the Content Delivery Network (CDN) is more interesting and visual.

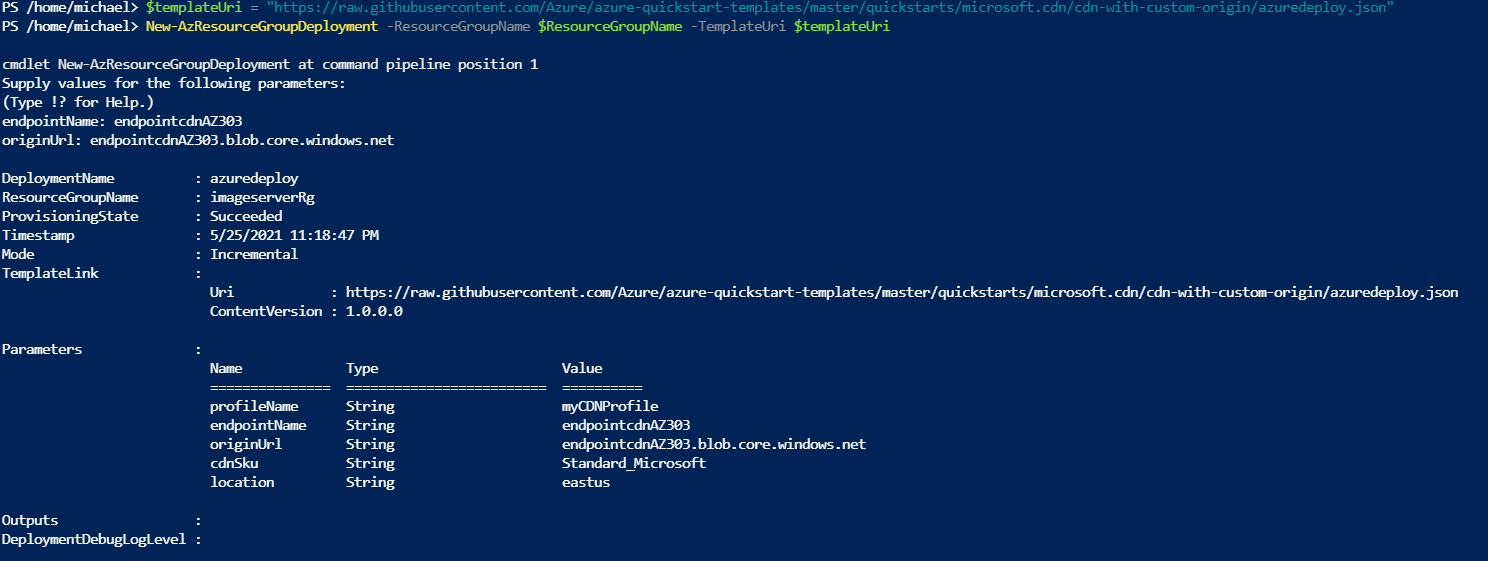
Step 8.1: Deploy the CDN Template (Retrieve from github):

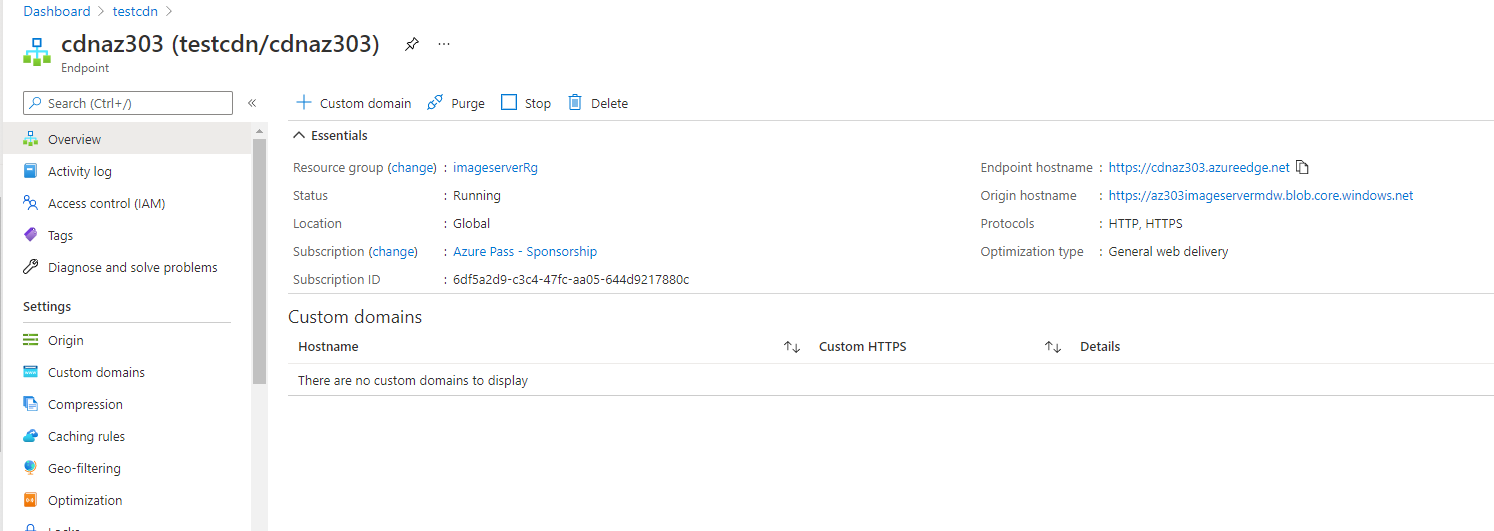
$templateUri = "https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/quickstarts/microsoft.cdn/cdn-with-custom-origin/azuredeploy.json"

New-AzResourceGroupDeployment -ResourceGroupName $ResourceGroupName -TemplateUri $templateUri

Two parameters must be supplied to the ARM deploy command:

endpointName: this must be a globally unique name as part of the url for retrievining images  
 originUrl: endpointPointName + .blob.core.windows.net (See below)

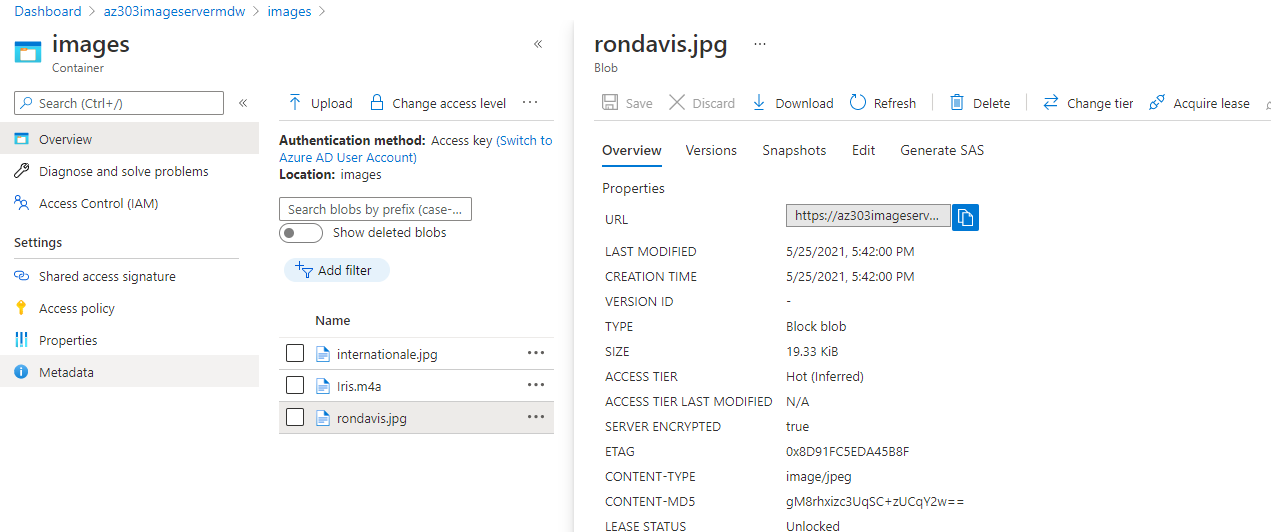


It takes around 15 minutes or more for the CDN Configuration to Complete (Check for EndPoint Hostname) -- This will the base url for retrieving images

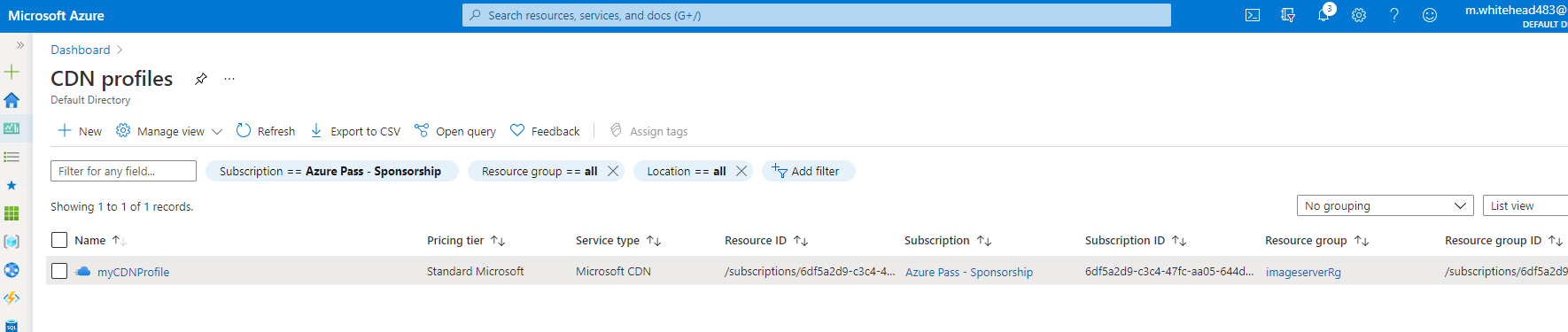
Full URL is base url + /<containerName>/imagefileName for example:

https:/ /az303imageservermdw.blob.core.windows.net/images/rondavis.jpg

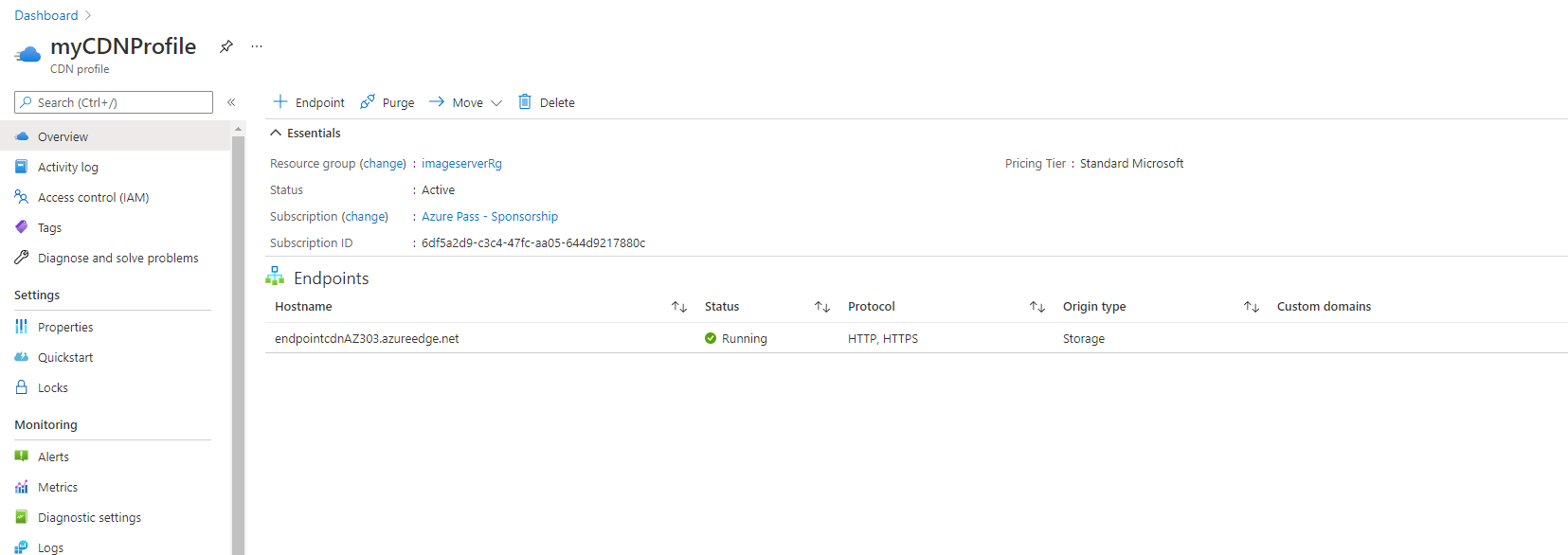
Full URL can be found by clicking on an object in the container:



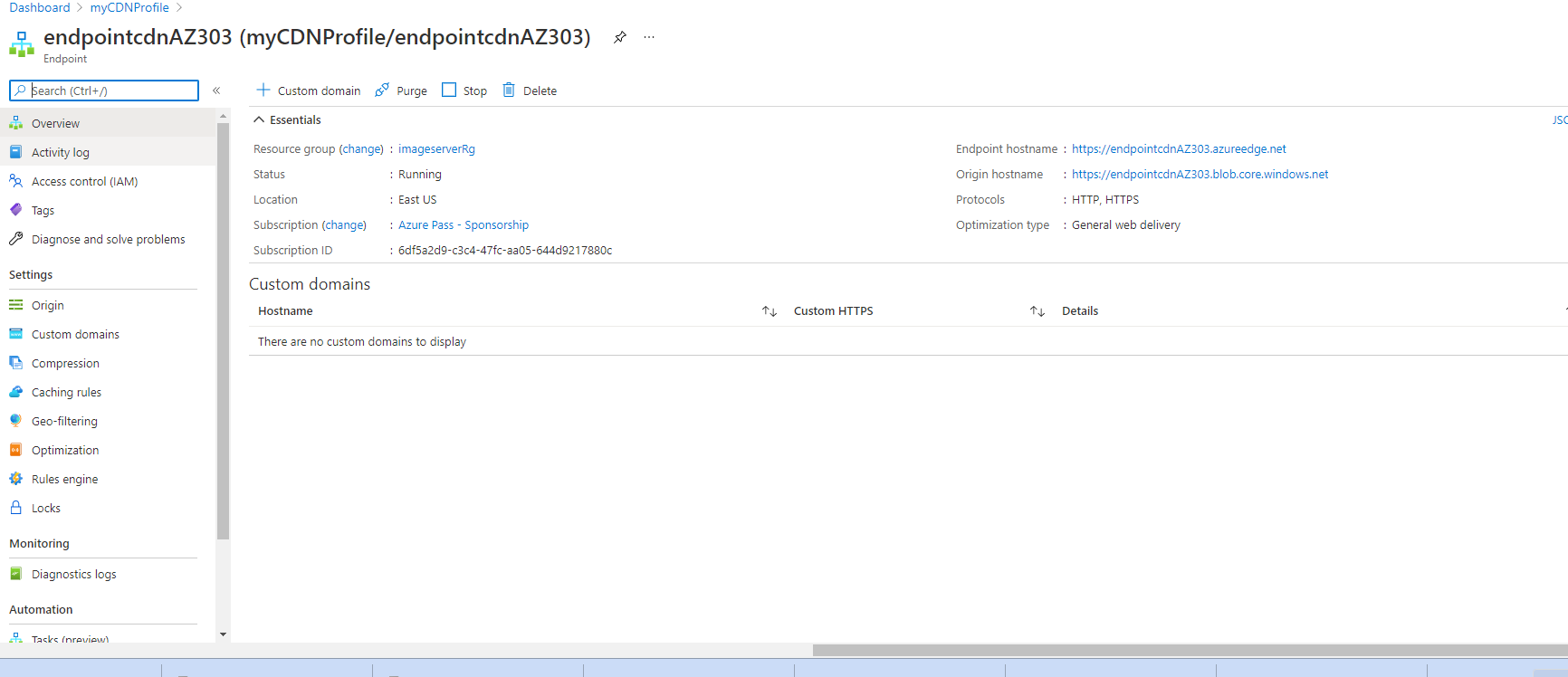
List of CDN Profiles



CDN Profile Detail



CDN Profile Detail Endpoint



Step 9: Blob Image Server in Action

Image files examples CDN blob item:

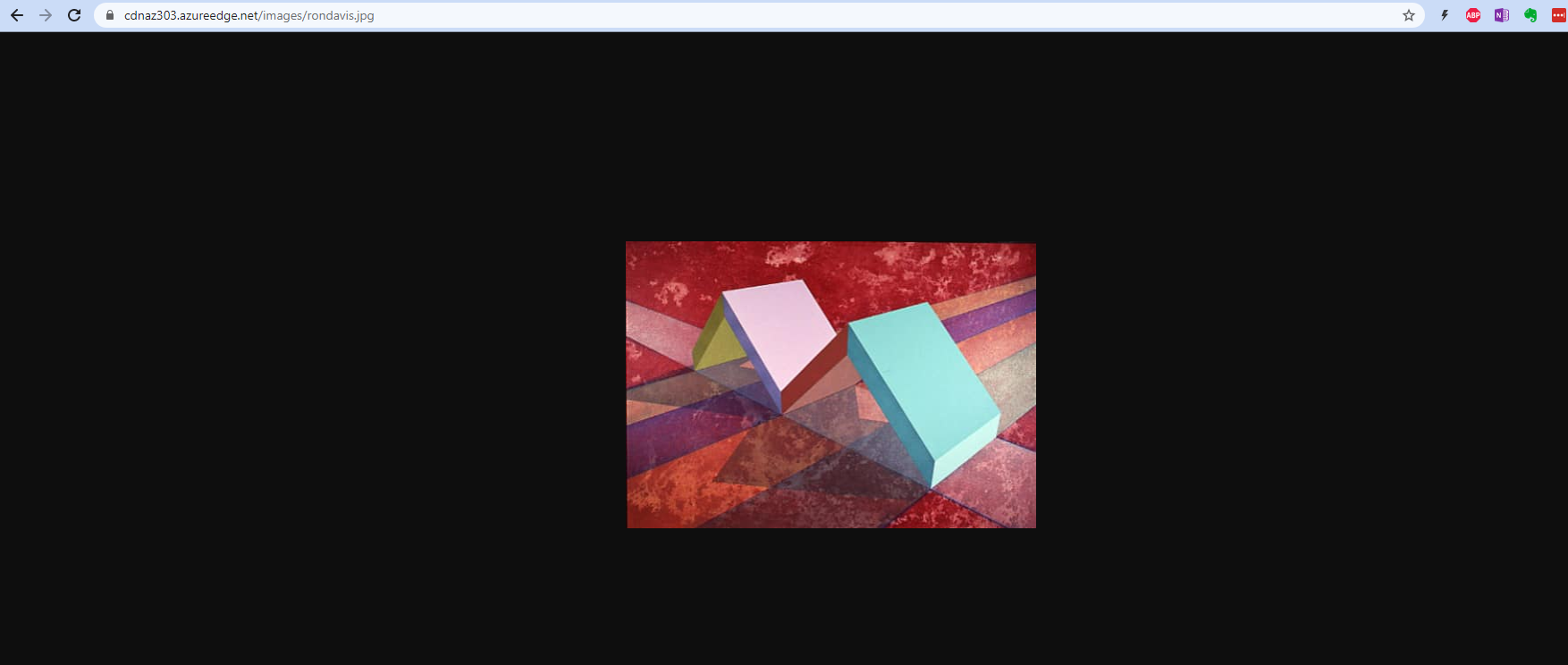
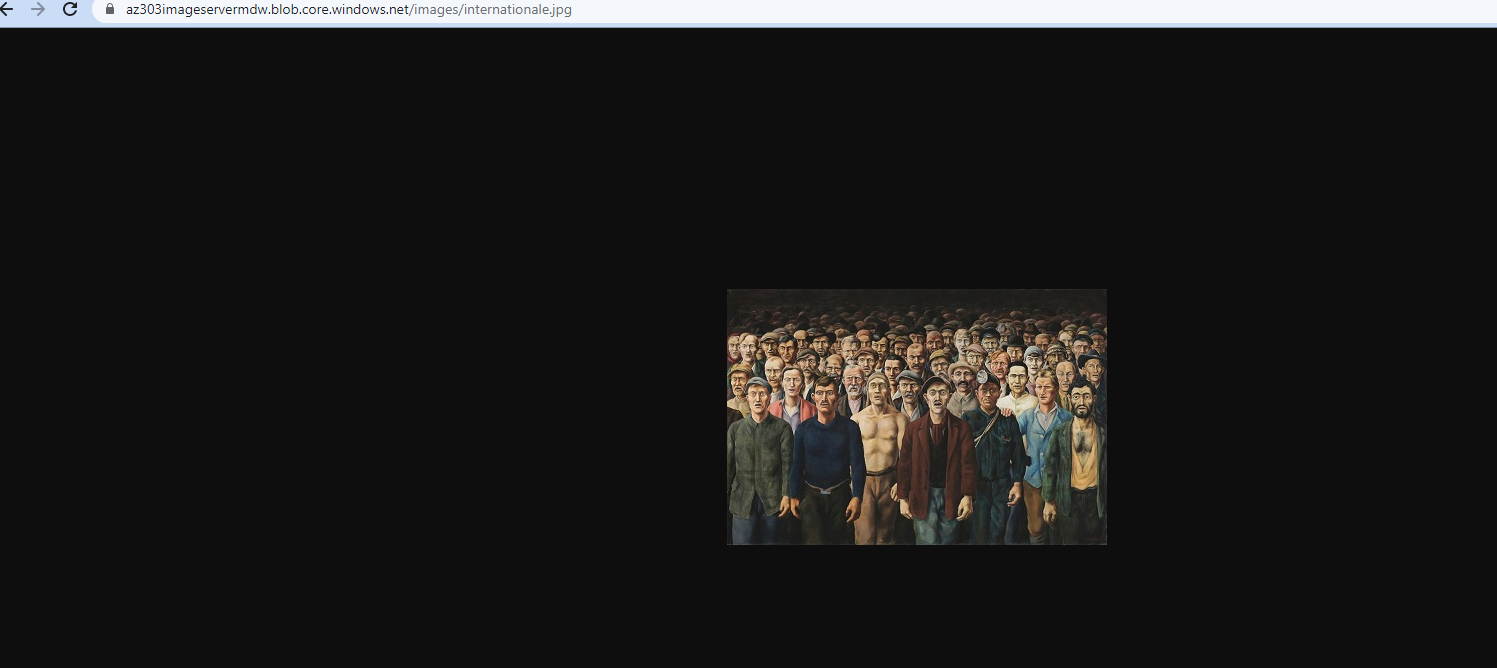


Image files examples CDN blob item:



Example music file blob (Yes it plays):

