# EITS RunBooks

### A. Create a Storage Account & Upload Arm Template

Run the powershell Create-storage.ps1. This requires the following parameters:

$ResourceGroupName = "rg-automation"  
$Location = "eastus2"  
$ProjectName = "hbtest"

The script will create a container and upload the arm template called LoadBalancedVirtualMachine.json to it

Make note of the url of the uploaded LoadBalancedVirtualMachine.json (you will pass this as a parameter). It will be something like:

https://hbtestautomationstorage.blob.core.windows.net/hbtestautomationstoragecontainer/LoadBalancedVirtualMachine.json

# B. Create File Storage For Installer Files

In the storage account created in Step A, create a file storage account with the following directory structure:

absg

ccmsetup

i386

ClientUpdate

x64

ClientUpdate

Upload ccmsetup installers to appropriate directories

**Make note of the Key & Storage Name** & File Path

Note down the Key & Storage Name. You will need it when creating the Credential Asset in the Automation Account

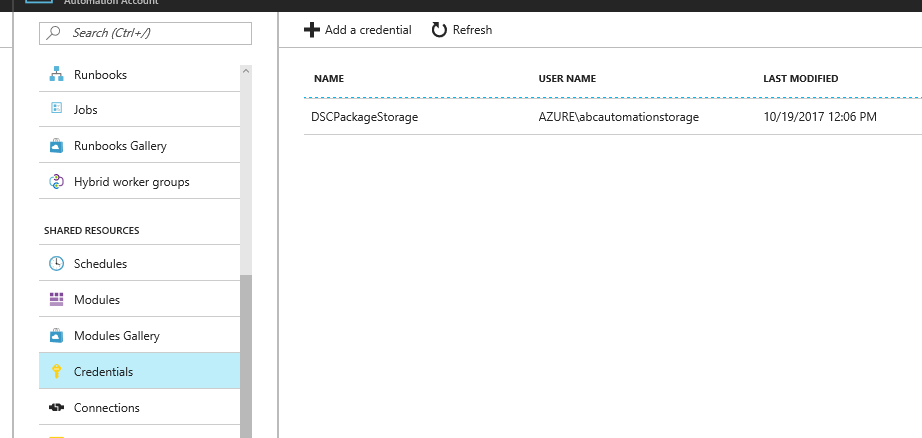
# C. Create an Automation account

### Add the following modules from the module gallery

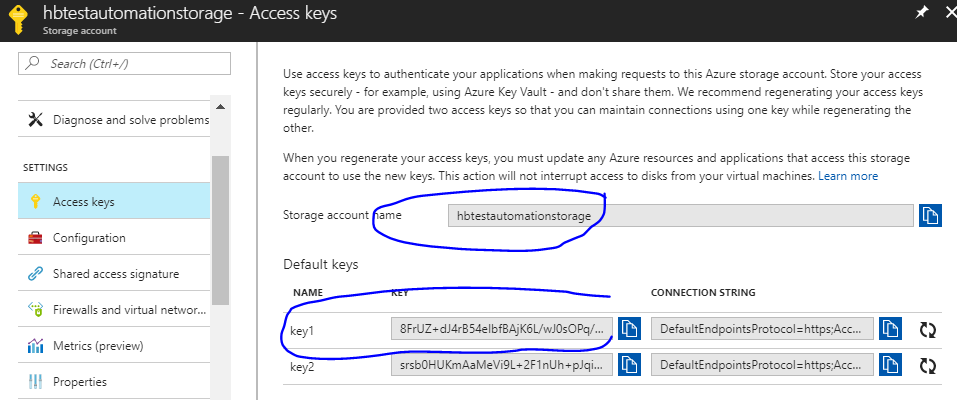
1. AzureRm.Profile (wait for a while after importing this)
2. AzureAD
3. AzureRm.Network
4. AzureRM.Tags
5. CredentialManager
6. AzureADPreview
7. AzureRM.KeyVault
8. MSOnline

### Create a Credential variable

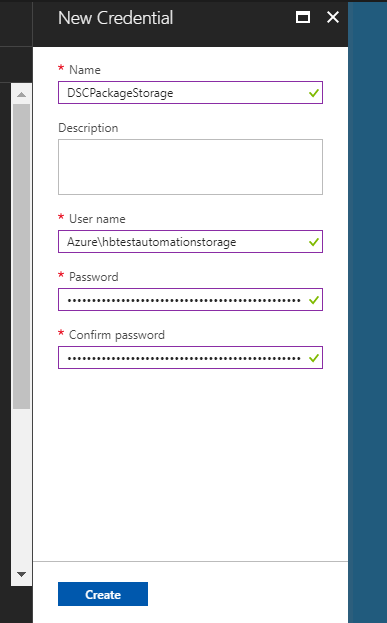
In Section B, you created a file storage to hold the installer files. In the automation account, you create a credential asset called **DSCPackageStorage** to point to that file storage



### To get the Username & Password refer to the Storage Account/Access Keys section as depicted in the screenshots below.



### Now create the Credential Asset DSCPackageStorage as below



**Name**: DSCPackageStorage

Username: AZURE\file-storage-name (this is the name of the storage account, prefixed by "AZURE\" )

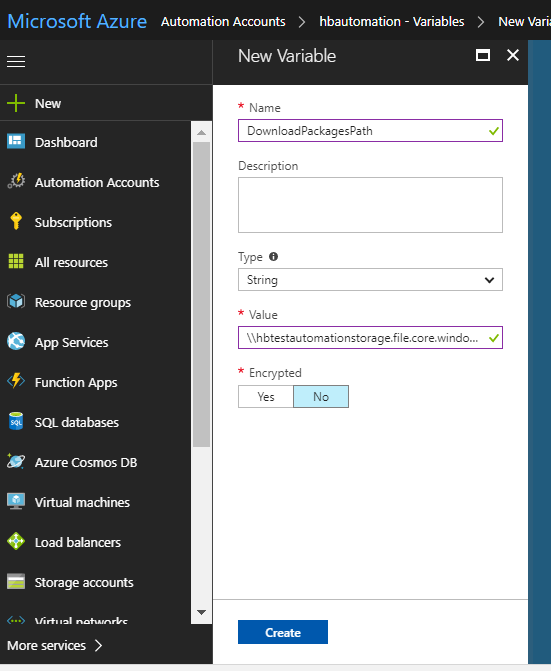
Password: Key of the file storage

Example:

Username: Azure\hbtestautomationstorage

Password: 8FrUZ+dJ4rB54eIbfBAjK6L/wJ0sOPq/XYQ9j4F7gig7kEiQWSY9mXlucHefGTCluq4kLPXvf7TgPZ+hPYNJ7A==

### Create an Automation Variable Asset



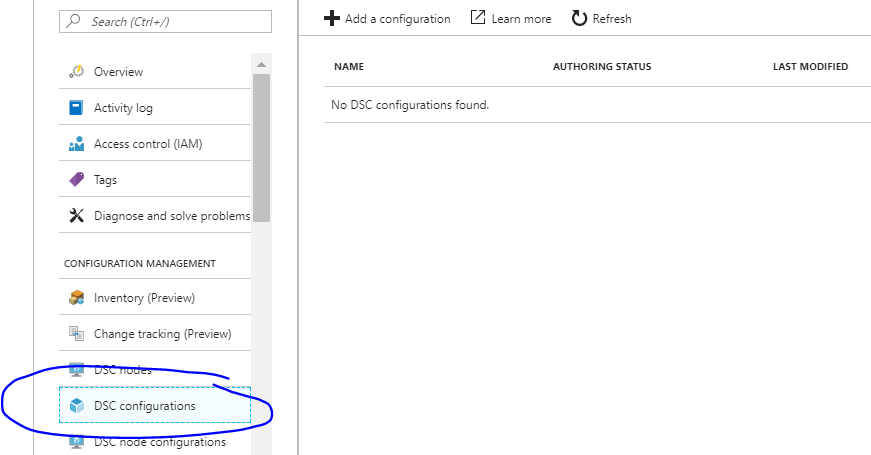
Name: DownloadPackagesPath

Value: \\storagename]file.core.windows.net\absg\Packages

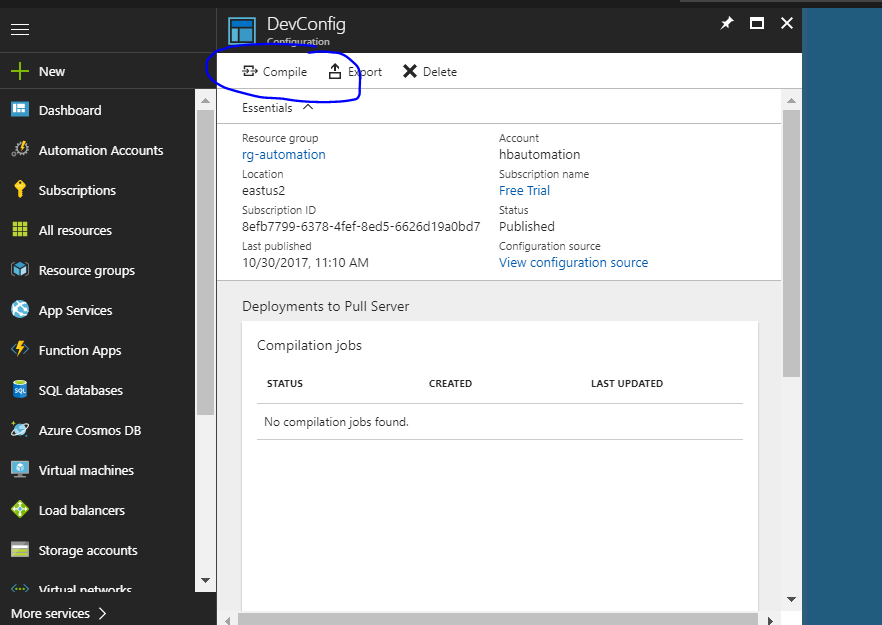
(example: \\hbtestautomationstorage.file.core.windows.net\absg\ccmsetup)

# D. DSC Configuration File

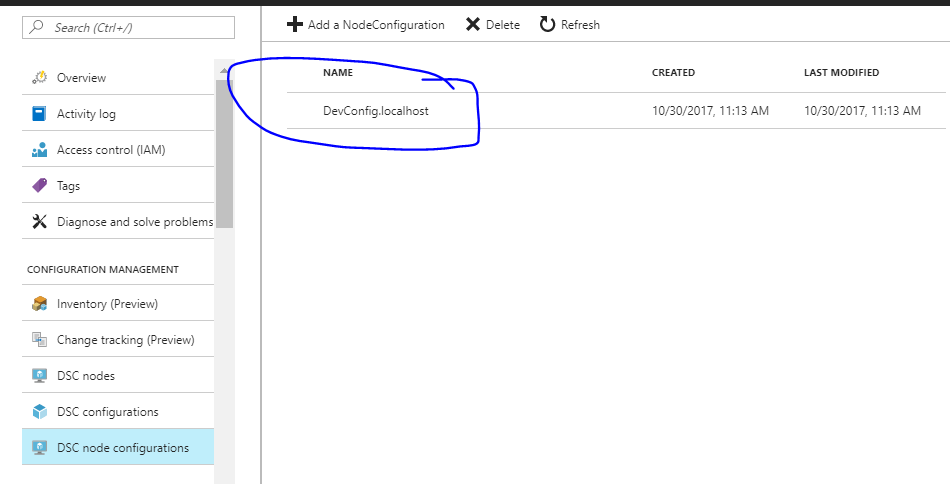
1. From this git project, import the file called DevConfig.ps1 to the **DSC Configurations** section in your-automation-account



1. When Imported, Click on it, compile it by choosing Compile on the Toolbar. Let it default for the ComputerName ("Default will be used").



1. The screen does not automatically refresh.
2. After it is published, click on the link "DSC node configurations, note the resultant name of the configuration (DevConfig.localhost) . You will need to pass this as a parameter to the Runbook



# E. Assign the automation account to Owner Role

1. In Azure Portal go to Subscriptions
2. Select your subscription
3. Click on Access control (IAM)
4. Click on Add
5. Role: Owner; Assign Access to: "Azure AD user, group, or application", In select, start typing the name of your automation user (imp. start typing else it is hard to find)
6. [pic OwnerToAutomation.png]

### Credential Automation Asset Of an AD User with "Owner Rights"

1. You need to have a AAD user with Owner rights. Follow the steps above to make him give him "owner rights".

# F. Import & Run Runbooks

The runbooks are numbered sequentially. There are 7 runbooks. The RunDeployEITS\_Env is a helper runbook that passes parameters to EITS2\_DeployEITS\_Env

$RegistrationKey : Key of the DSC automation account (found on the Keys section of the automation account)

$RegistrationUrl: URL of the automation account, found also in the Keys section of the automation account

$NodeConfigurationName: The name of the configuration noted in Step D4

.$TemplateFile: The full path to the arm template noted in Step A.