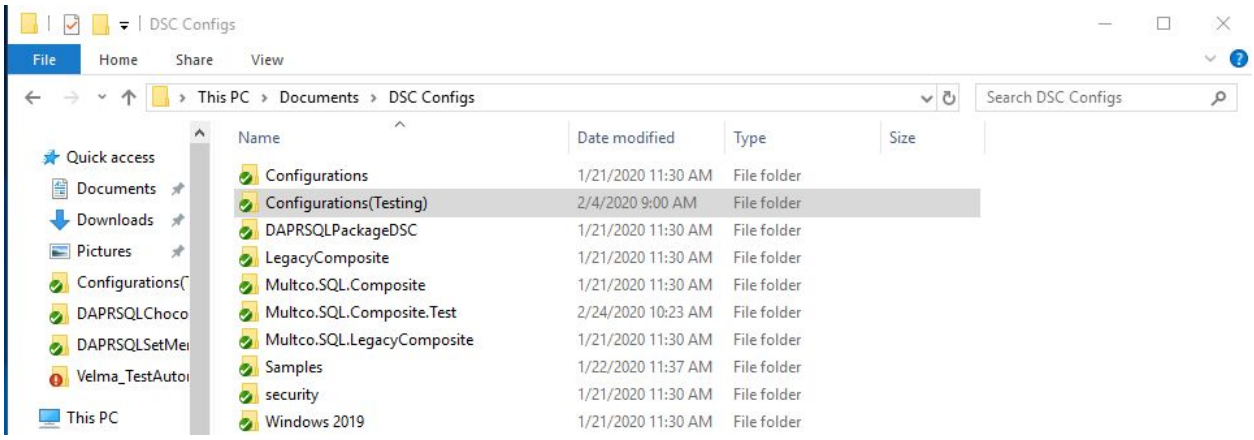
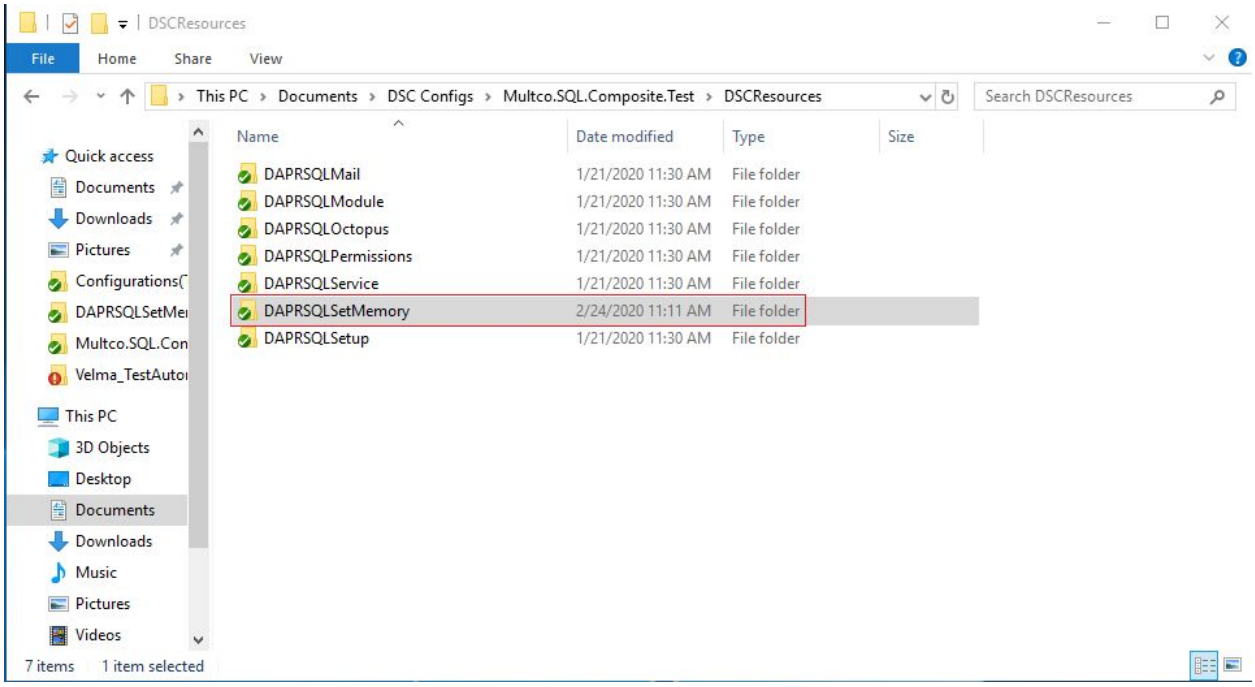
So you want to add a resource to DSC?
This is a complicated and hard process but hopefully this guide will help you publish your resource. One thing to note about this tutorial is, I will be using the “Configurations(Testing)” folder as well as the Multco.SQL.Composite.Test folder.

1. The first and easier step is to checkout the DSC config folder from SVN. While the setup may be different from when I write this, here is what was contained in the folder when I used it last:

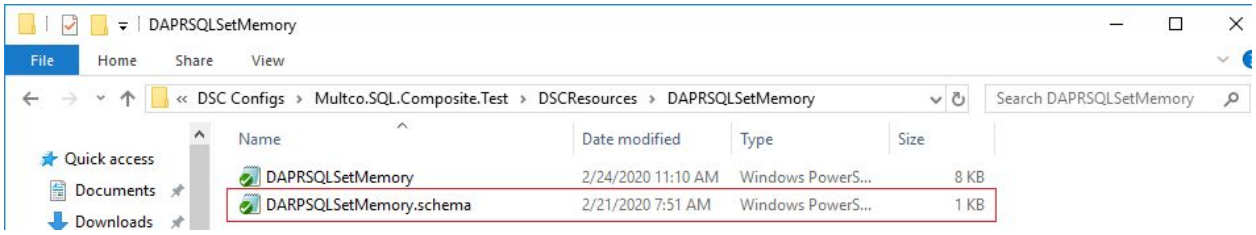


2. From there, go into the DSCResources folder, I will be creating a resource today called “DAPRSQLSetMemory”. The goal of this resource is to set the max memory for a VM that uses the “Configurations(Testing)” configuration. Create a new folder with the name of the resource you want. I copied the “DAPRSQL” part of the resource name to follow naming procedure:

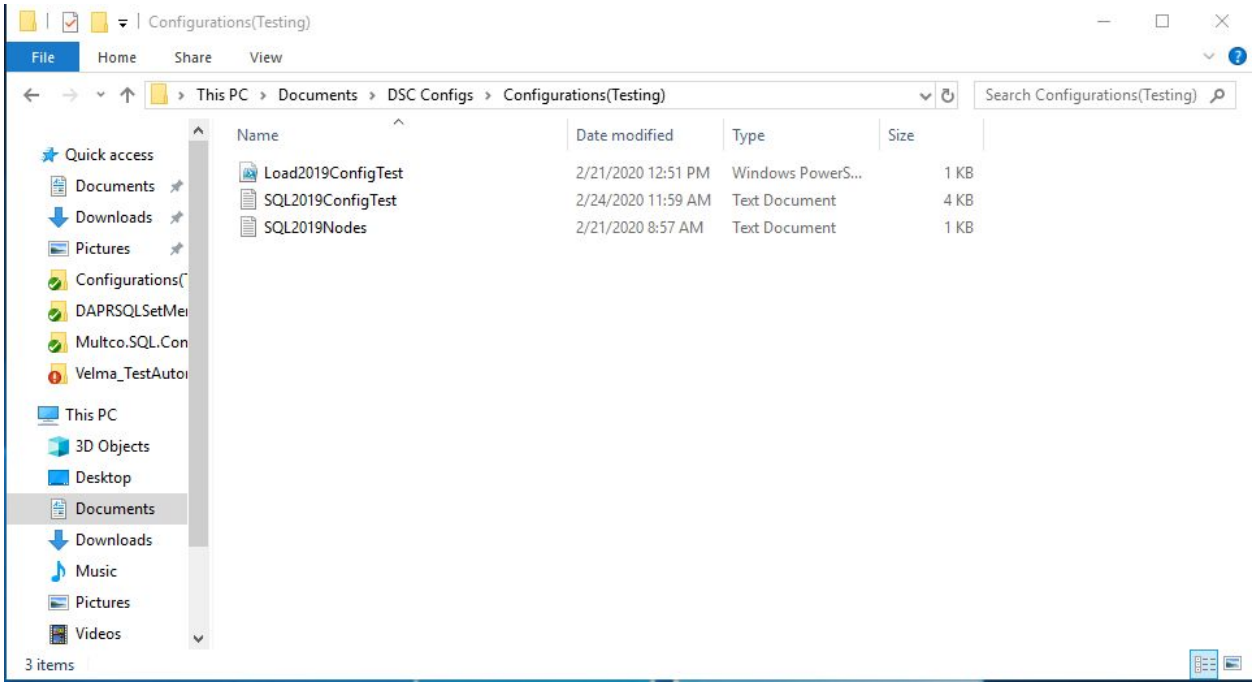


(When you create this file, you may see a red exclamation mark instead of a green check mark on the folder, that simply means you have not committed that folder to SVN.)

3. Next up, You want to make a power shell module file with the following name: “$ResourceName.schema.psm1”. Insert your resource name into the folder name and make sure the file is a power shell module file (psm1). Go ahead and open that up with Powershell ISE and we will begin editing the resource. A quick side note: when it comes to editing a powershell script, powershell ISE will be easier to use than the traditional powershell terminal.



4. After writing the code for what you would like your module to do, go back to the DSC Configs file and to go “Configurations(Testing)”, you should have the following three files:



5. The first of these files is the load2019ConfigTest file, this will be the file you run when in powershell ISE when we finish making the changes we want to make. For now, the only two things you need to worry about for this file is making sure you have the Az.Automation module installed (if you do not try: install -module “Az.Automation” (You may need to get admin permissions if the download denies you access, in which case, contact the help desk)). The second thing is to make sure you have the correct path for your source path. You want this to point to your “SQL2019ConfigTest” file, like so:

```
roupName "rg-DAPR-DSCSBX" -SourcePath "$scriptPath\SQL2019ConfigTest.txt" -Force -Published

Params
sourceGroupName "rg-DAPR-DSCSBX" -AutomationAccountName  "aaDAPRSBX"  -ConfigurationData $NodeDa
```

6. The next step is to go into the SQL2019ConfigTest file and do a few things. First make sure the name of the configuration function at the top of the game has the same name as the file. Second, make sure the resource imports list the name of the resource folder you want to use. I say this because when creating a new config, you may need to borrow code from other configurations to get the correct syntax and format.

```
Configuration SQL2019ConfigTest {

        Import-DscResource -ModuleName Multco.SQL.Composite.Test
        Import-DscResource -ModuleName PSDesiredStateConfiguration
        Import-DscResource -ModuleName Az.Automation
        $SqlInstallCredential = Get-AutomationPSCredential 'IT_SQLInstall'
        $MailServer = Get-AutomationVariable -Name 'MailServer'
```

Finally, you must add a function of your module for the configuration to use:

(INSERT PICTURE OF FINISHED SQL2019 CONFIG TEST FUNCTION HERE).

7. The last file in the configuration folder we will be looking at is “SQL2019Nodes”. This file is responsible for which nodes will be affected by our configuration as well as which parameters will be referenced throughout the rest of the configuration scripts. For me, I needed to add one parameter, that being the max memory of the nodes I wanted to influence. For this I had to look up what to add and discovered that the max mem parameter is measured in megabytes. With that in mind, here is what I added for a test of node parameters:

```
@{NodeName = 'SQL703SBX'
Role = "SQL"
Dept = "DARS"
Env = "SBX"
SQLInstances = @"
{"Instances":[
{"SQLRole":"OLTP",
"SQLVersion":"2019",
"SQLEdition":"Developer",
"SQLInstance":"SQLSVR",
"SQLServiceAccount":"SVC_SQL702SBX",
"SQLPort":"2505",
"SQLBackupDay":"Monday",
"SQLBackupHour":"19",
"SQLBackupMinute":"30",
"SQLMaintDay":"Sunday",
"SQLMaintHour":"15",
"SQLMaintMinute":"30",
"SQLMaxMem":"2048"}
]
```

8. Now, remember when we were making our schema file and there were two files there? The second file is generated from a script to give the resource its own unique GUID, which becomes important later. For now, this is the script that was given to me to generate the GUID, to run this, you must first change the folder path names to match up

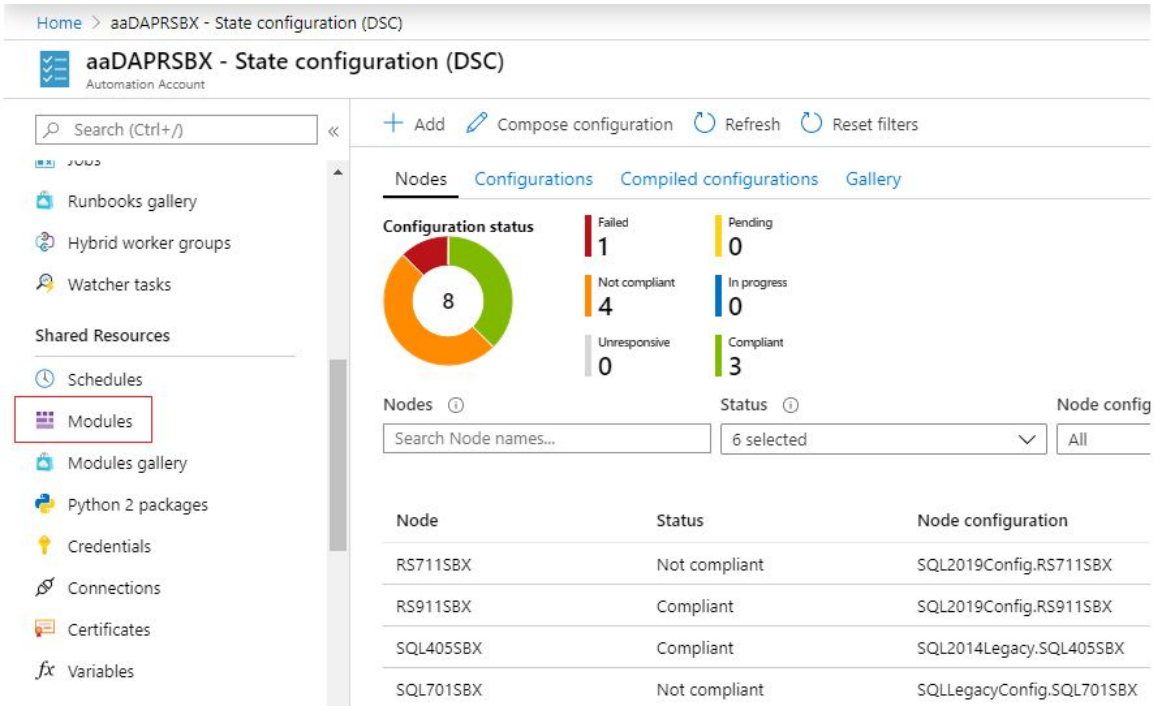
with the schema file that was created earlier, I will box the areas I had to change for this script to work:

```
$ModuleFolder = "C:\Users\brandond\Documents\DSC Configs"

$CompositeName = "Multco.SQL.Composite.Test"

#Base composite
#New-Item -Path $ModuleFolder\$CompositeName -ItemType Directory -Force
#New-ModuleManifest -Path $ModuleFolder\$CompositeName\$CompositeName.psd1
#New-Item -Path $ModuleFolder\$CompositeName\DSCResources -ItemType Directory -Force


#First Resource
$ResourceName = "DAPRSQLSetMemory"

New-Item -Path $ModuleFolder\$CompositeName\DSCResources\$ResourceName -ItemType Directory -Force
New-Item -Path $ModuleFolder\$CompositeName\DSCResources\$ResourceName\$ResourceName.schema.psm1  -ItemType File -Force
New-ModuleManifest -Path $ModuleFolder\$CompositeName\DSCResources\$ResourceName\$ResourceName.psd1 -RootModule .\$ResourceNam
```

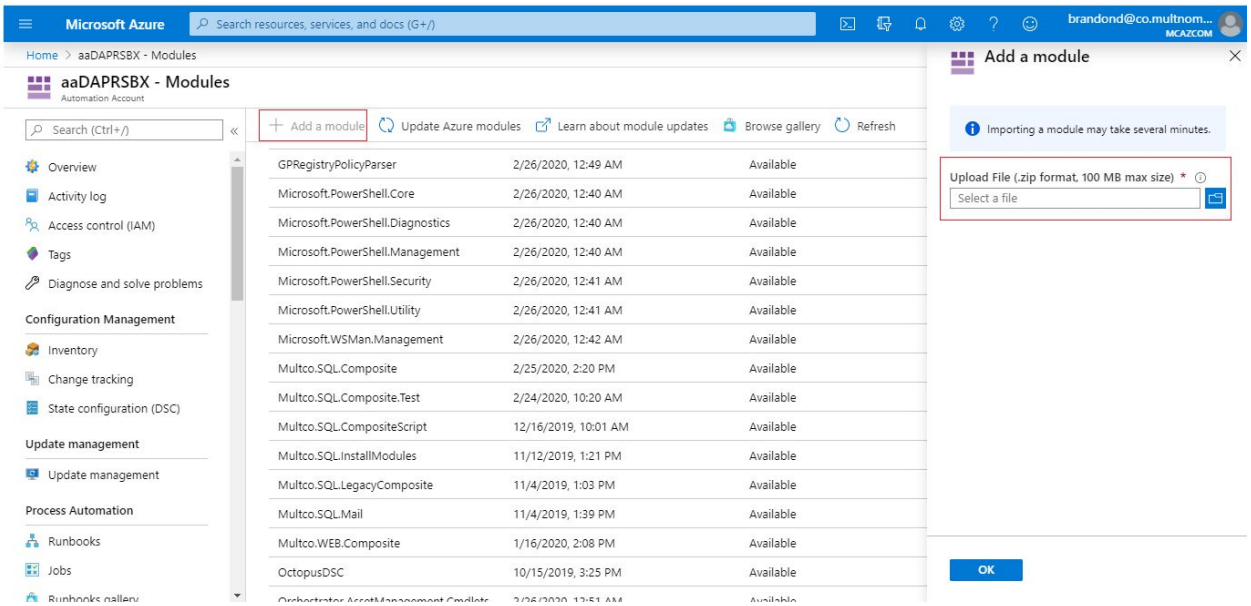
9. Now that all of that is complete, the next few steps take place in azure. Here is the portal to log into azure:

   https://portal.azure.com/#@mcazcom.onmicrosoft.com/resource/subscriptions/305ea49c-0f9d-472c-b7a4-e286fde3cb1b/resourceGroups/rg-DAPR-DSCSBX/providers/Microsoft.Automation/automationAccounts/aaDAPRSBX/overview

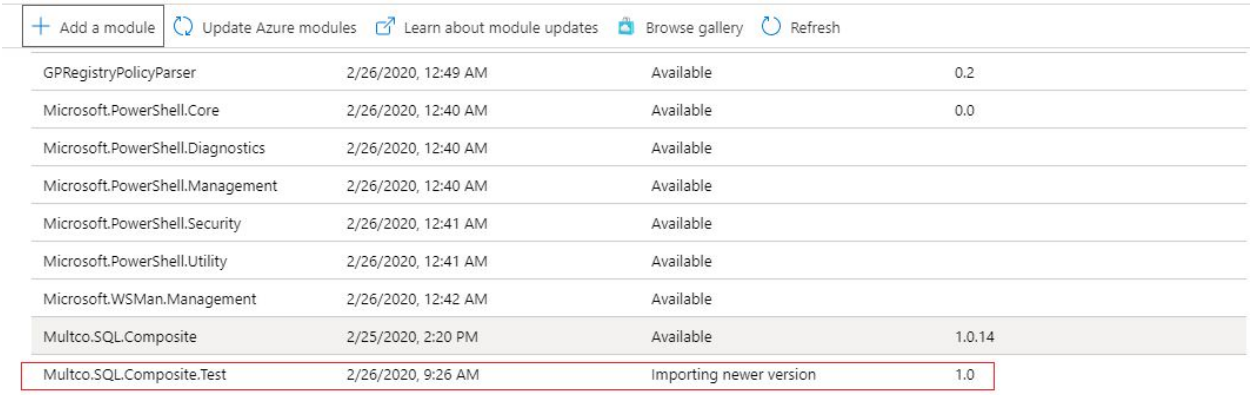
10. Once you are there, you will need to upload your module to azure before you attempt to run your "Load2019ConfigTest" file. To do this, first navigate to the "Modules" tab on your left hand side.



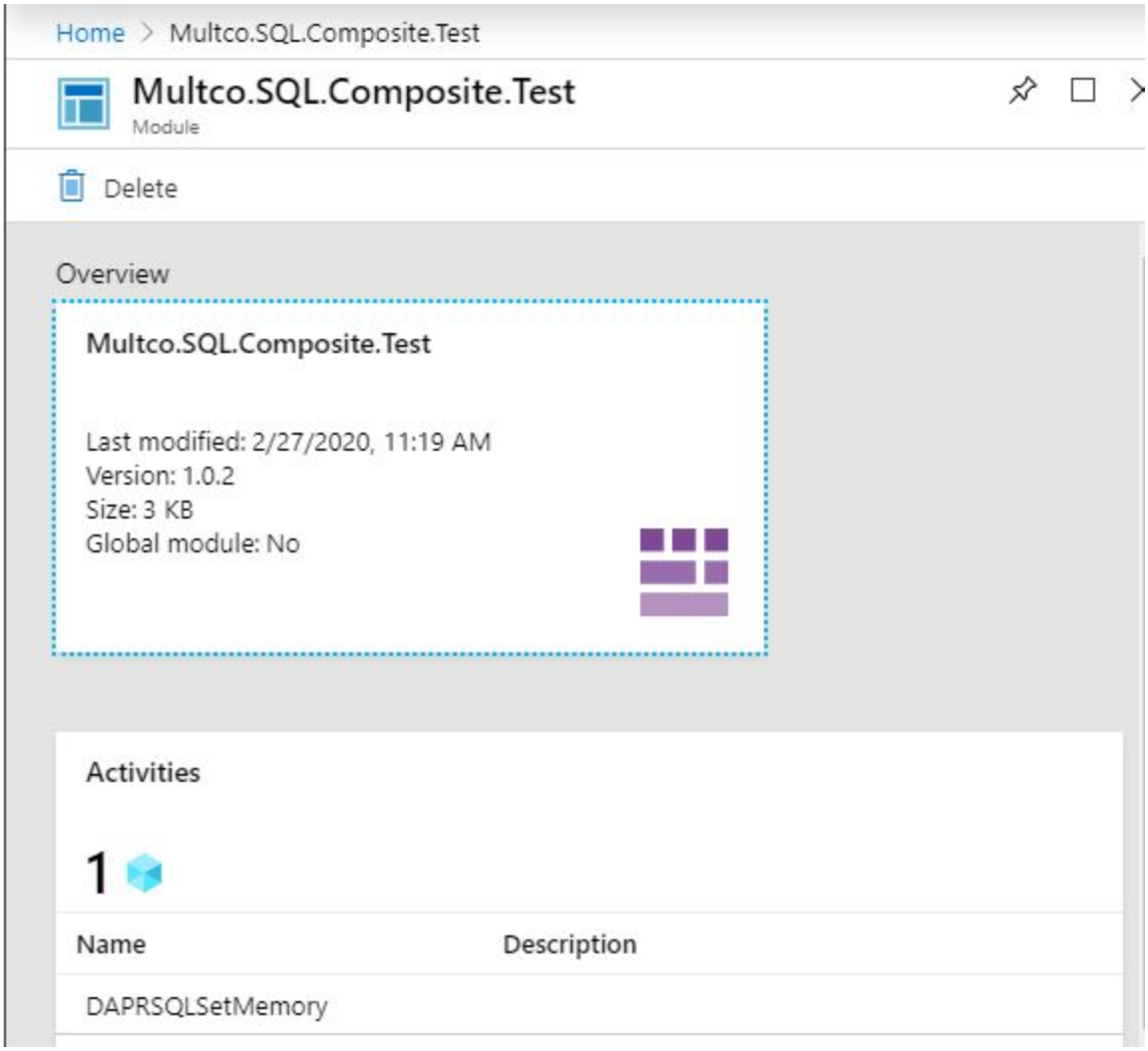
11. Since there is an older version of Multco.SQL.Composite.Test already created, we will be overwriting that version with the new addition we have made. To do that, click on the "Add a Module" opton. You will be prompted to add a zip file containing your resource. You will need to send your resource to a zip file and upload it in azure.



12. Go back to your file explorer and create a zip file for your resource. In my case, from the DSC config folder, I simply send the “Multco.Composite.Test” folder to a zip folder and then uploaded that.
13. Once you have done this, you can find the module you are trying to import or overwrite to see its progress. Since I am overwriting a module, here is what it looks like for me:

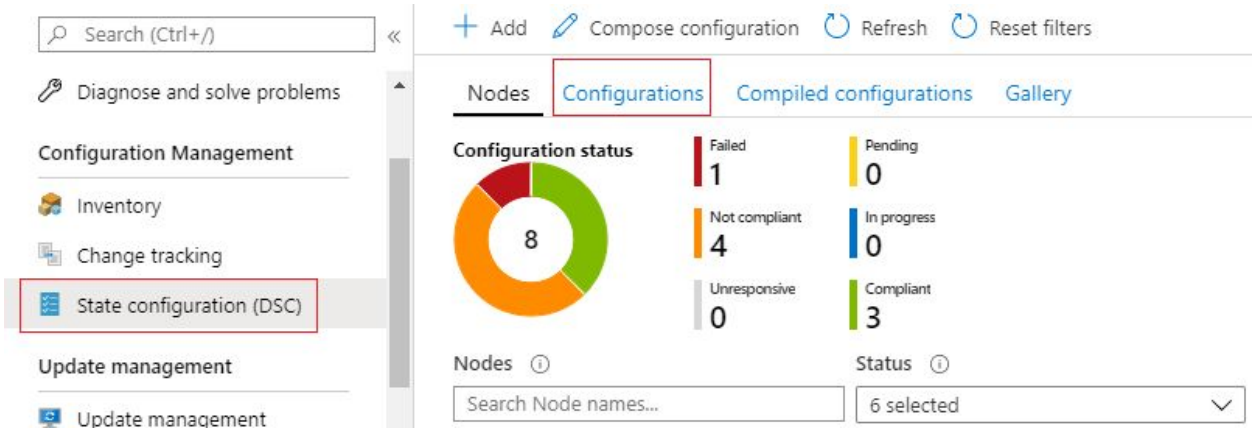


14. Once the import is done, you can click on that module and confirm that the resource you have created has been added to azure.

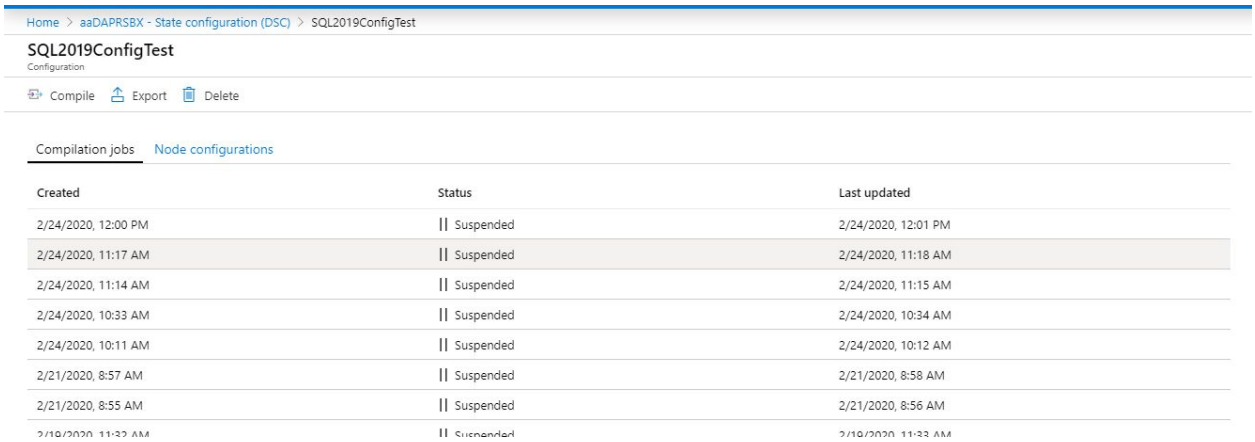


15. Now to run the “Load2019ConfigTest” file. Open it up in Powershell ISE and run it. You should be prompted for credentials. For username, I put the name of the profile for logging into my local machine followed by or.multnomah.or.us, so like this: “$login@co.multnomah.or.us”. Use your normal password for the password. If there are no errors that appear you can now go see the status of your configuration push on azure.

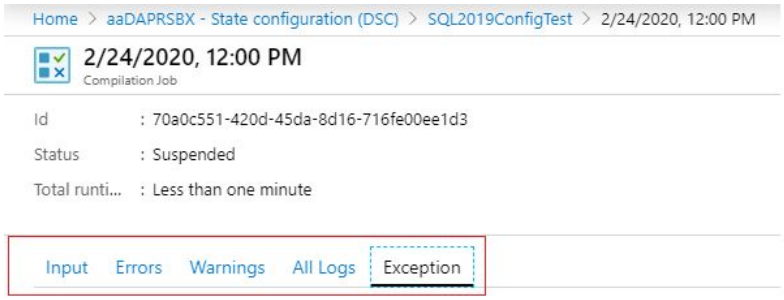
16. Go back to azure and find the DSC option on the left, followed by pressing the “configurations” tab.



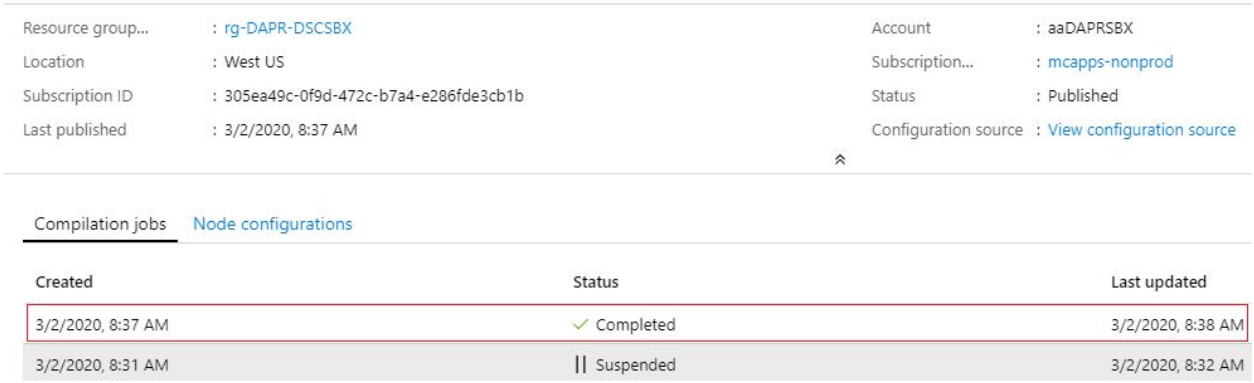
17. Look up the configuration we are working on, which in this case is SQL2019ConfigTest. You will see a screen that looks like this:



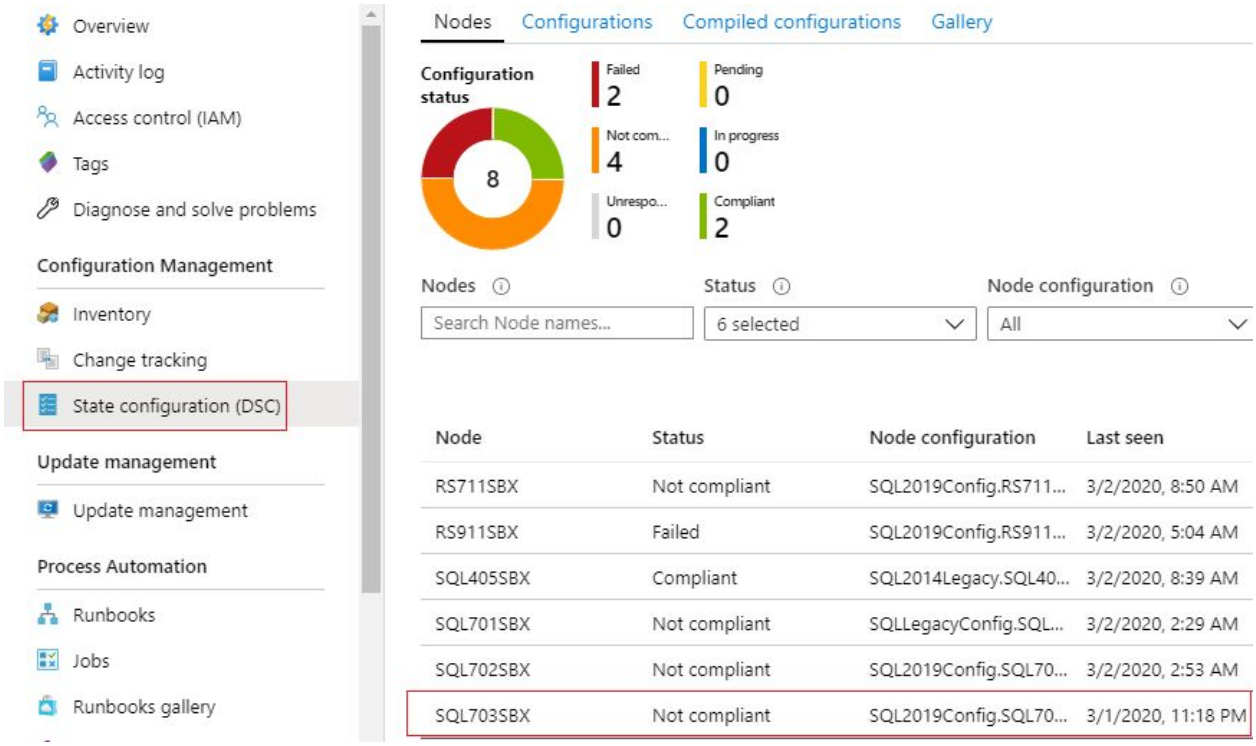
18. This will show each time you try to run your Load2019ConfigTest. Suspended means there was an error or exception thrown that prevented the configuration from being uploaded. You can click on each instance to view the error or exception. This is how you will debug your DSC configuration.



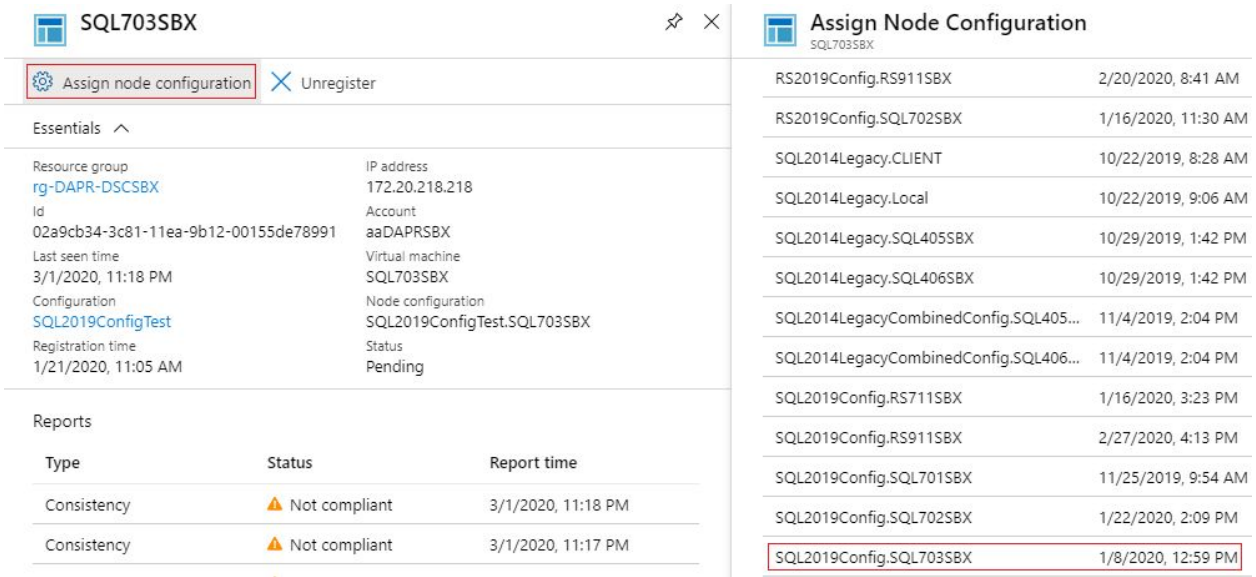
19. Once your debugging is finished, you will have a line that confirms that your script has been completed.



20. Now that your configuration has been uploaded, you should bind that configuration to the node you want to adhere to your configuration's rules. To do this, first go back to the DSC tab on the left and click on the node you want to change. In my case, that would be SQL703SBX.



21. After clicking on the node, you can add a configuration to it by clicking on the “Assign node configuration” button and then assigning your configuration to the node.



22. Once the configuration is completed and you hooked up your node to that configuration, depending on what you have altered, you may now go an check on the results. In some cases, if you have a pull configuration, you may need to either:
    1. Use an onboarding script to have the changes take effect immediately.
    2. Wait until the pull interval has passed to check your work.

Resources

- https://github.com/dsccommunity/SqlServerDsc#sqlservermemory
- https://devblogs.microsoft.com/powershell/powershell-dsc-resource-design-and-testing-checklist/
- https://app.pluralsight.com/library/courses/azure-automation-getting-started/table-of-contents
- https://app.pluralsight.com/library/courses/play-by-play-azure-automation-dsc-elastic-scale-consistency/table-of-contents
- https://app.pluralsight.com/library/courses/azure-automation-dsc-managing-configurations/table-of-contents