VMware

NSX Automation Workshop

June 14 – 15 2022

Lab 3 Manual

LAB 3: Power CLI

In this lab we will use Power CLI to perform operations on our NSX environment. The following tasks will use basic authentication to NSX Manager with additional parameters

Task 1:

Open a PowerShell Window and check if PowerCLI is installed

```
Command: Get-Module -Name VMware.PowerCLI -ListAvailable

Result
   Directory: C:\Users\Administrator\Documents\PowerShell\Modules

ModuleType Version       PreRelease Name                         PSEdition ExportedCommands
---------- -------       ---------- ----                         --------- ----------------
Manifest   12.3.0.17…    VMware.PowerCLI                          Desk
```

Task 2:

Check if NSX-T Modules are there

```
Get-Module "VMware.VimAutomation.Nsx*" -ListAvailable

Result
    Directory: C:\Users\Administrator\Documents\PowerShell\Modules

ModuleType Version     PreRelease Name                           PSEdition ExportedCommands
---------- -------     ---------- ----                           --------- ----------------
Script     12.3.0.17…  VMware.VimAutomation.Nsxt                 Desk      {Connect-NsxtServer…
```

Task 3:

Let's connect to NSX-T

```
Connect-NsxtServer -Server IP-Address/DNS-Name -User Username -Password password
```
Replace IP Address/DNS-Name Username and password.

```
Result
--> It takes a looooong time. Be patient
```
…. Connected

What happens here ?

Task4:
Check if T1 exists

```
$t1routerdata = Get-NsxtPolicyService -Name com.vmware.nsx_policy.infra.tier1s
$t1routerdata.list().results
```

What these commands are doing ?

Task5:
Create a T1 using a script
Variables are set in the script below

```
#Variables for NSX Manager Connection
#General Variables
$description = "Created with VMware PowerCLI"
$tag = "powercli"
#Variables for T1 Router
$t1routerid = "T1-Test-001"
$t1routeradvertisement = @("TIER1_IPSEC_LOCAL_ENDPOINT","TIER1_CONNECTED")
#Connect to NSX Manager
#Connect-NsxtServer -Server $nsxmanagerip -User $nsxuser -Password $nsxpasswd
#Retrieve Router Information
```

```
$t1routerdata = Get-NsxtPolicyService -Name com.vmware.nsx_policy.infra.tier1s
#Set Variables
$t1routerspecification = $t1routerdata.Help.patch.tier1.Create()
$t1routerspecification.description = $description
$t1routerspecification.id = $t1routerid
$t1routerspecification.display_name = $t1routerid
# $t1routerspecification.tier0_path = $t1routerpath_to_t0_rtr
$t1routerspecification.route_advertisement_types = $t1routeradvertisement
#Add Tag to the Router
$t1routertag = $t1routerdata.Help.patch.tier1.tags.Element.Create()
$t1routertag.tag = $tag
$t1routerspecification.tags.Add($t1routertag) | Out-Null
#Create T1 Router
$t1routerdata.patch($t1routerspecification.id, $t1routerspecification)
```

Create a ps1 file and launch it.

Result

Task6:

Check the T1 is Created

Through UI

Using PowerCLI Commands

```
$t1routerdata = Get-NsxtPolicyService -Name com.vmware.nsx_policy.infra.tier1s
$t1routerdata.list().results
```

Result

Task7:

We used some commands to create this T1. How does it work ?

Details of the script

Task8:

List the Services you can use with NSX-T

```
$Serv = Get-NsxtPolicyService -Name com.vmware.*
$Serv.list
```

Task9:

Let's see how to use Help based on this example:

```
$segmentdata = Get-NsxtPolicyService -Name com.vmware.nsx_policy.infra.segments
$segmentspecification = $segmentdata.Help.patch.segment.Create()
```

Task 10:

Let's create a Segment, now

What do we need ?

T1 Router -> We're going to use the name of the T1 Router we've created in Task 5

Transport Zone

Retrieve transport zones Informations:

Through the UI

Via PowerCLI script

```
$tZoneSvc = Get-NsxtService -Name com.vmware.nsx.transport_zones
$tZoneSvc | Get-Member
$tZones = $tZoneSvc.list()
$tZones.results
```

Choose an IP Address and Mask for the Segment Gateway.

Script

```
#General Variables
$description = "Created with VMware PowerCLI"
$tag = "powercli"
#Variables for Segment
$segmentid = "Seg-Test-PowerCLI"
$transportzone = "/infra/sites/default/enforcement-points/default/transport-zones/
TRANSPORTZONEID"
$path_to_t1_rtr = "/infra/tier-1s/T1ROUTERNAME"
$defaultgateway = "IP-ADDRESS/MASK"
#Connect to NSX Manager
#Connect-NsxtServer -Server $nsxmanagerip -User $nsxuser -Password $nsxpasswd

#Retrieve Segment Information
$segmentdata = Get-NsxtPolicyService -Name com.vmware.nsx_policy.infra.segments

#Set Variables
$segmentdata = Get-NsxtPolicyService -Name com.vmware.nsx_policy.infra.segments
$segmentspecification = $segmentdata.Help.patch.segment.Create()
$segmentspecification.description = $description
$segmentspecification.id = $segmentid
$segmentspecification.transport_zone_path = $transportzone
$segmentspecification.connectivity_path = $path_to_t1_rtr

#Set Default Gateway Variables
$subnetSpec = $segmentdata.help.patch.segment.subnets.Element.Create()
$subnetSpec.gateway_address = $defaultgateway
$segmentspecification.subnets.Add($subnetSpec) | Out-Null
```

```
#Add Tag to the Segment
$segmenttag = $segmentdata.help.patch.segment.tags.Element.Create()
$segmenttag.tag = $tag
$segmentspecification.tags.Add($segmenttag) | Out-Null
#Create Segment
$segmentdata.patch($segmentid, $segmentspecification)
```

Create a ps1 file and launch it.

Task 11:

How to remove the Segment and T1 ?

Go to UI and check the Segment and T1.

Delete the Segment and T1

You can do that via PowerCLI too but not much time to do it now ;)

Task 12:

Disconnect from NSXT Manager

```
Disconnect-NsxtServer
```

To go further: Check an example like this one
https://luchodelorenzi.com/2020/08/12/quickly-create-nsx-t-segments-using-powercli-and-nsx-t-rest-api/