# Callback or Control scale testing steps

In this document, we explain how to run callback or control scale tests. This document is not intended to be a cookbook, but rather an outline of steps to help the engineer to navigate through the stages. Also, it explains how different scripts are connected.

There are two scripts.

1. Python scripts to setup and cleanup configuration
2. powerCLI scripts to get Guest VM IP-address.

This document is on top of the steps required for vertical scale as mention in another document (instructions running performance testbed.doc).

Let’s begin:

1. Go to F: \scripts. This is already loaded with Latest scripts are in the Git Hub.
2. (optional) If you desire to get the latest scripts.
3. cd to infrascripts-master. There are three directories. perfscripts, scalescripts & testscripts.
4. Follow the steps for to vertical scale tests from document (instructions running performance testbed.doc) & run the Options 5,7,8 belongs. **This will bring up 32 Guest VMs with 1 vNIC and ipv4-address configured.**
5. Validate the connectivity. run ./ping\_linux\_vm.sh from each Guest VM. This will test connectivity to all Linux-VMs newly added. If issues, please fix them.
6. Create only **one** Security Policy as required for the testing

A sample Security Policy:

*Create a Security Policy with the following configuration as a template.* ***Note*** *the Services are* ***optional*** *as well. We are showing 5 services here (ICMP -3, Syslog -2) for educational purpose only.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Network Introspection Services** | **Source** | **Destination** | **Action** | **Service** | **Service Name** | **Service Profile** |
| SP1 | Name - inbound | Policy's Security Group | SG1 | Redirect to Service | ICMP Echo | Partner Service Name | Partner Service Profile |
|  |  |  |  |  | ICMP Redirect |  |  |
|  |  |  |  |  | ICMP Destination U |  |  |
|  |  |  |  |  | Syslog (UDP) |  |  |
|  |  |  |  |  | Syslog (TCP) |  |  |
|  | Name - outbound | SG1 | Policy's Security Group | Redirect to Service | ICMP Echo | Partner Service Name | Partner Service Profile |
|  |  |  |  |  | ICMP Redirect |  |  |
|  |  |  |  |  | ICMP Destination U |  |  |
|  |  |  |  |  | Syslog (UDP) |  |  |
|  |  |  |  |  | Syslog (TCP) |  |  |

1. Open the powerCLI script

infrascripts\testscripts\CallbackScale362\GetAllVM\_IPAddr.ps1

Make sure that the prefix is matching the Guest VM names

$VMPrefix = "Linux-VM\*"

***PS: Also validate the VC IP address, username/password.***

Get all Guest VM IP-address. Run PowerCLI script

cd infrascripts\testscripts\CallbackScale362\

PS > .\GetAllVM\_IPAddr.ps1

This will generate a file with all valid guest VMs object-ids.

…\input\gvm\_object\_ids.txt

This will also generate a log file with more details.

…\output\ps\_gvm\_objects.log

1. Open PyCharm  by double-clicking on the icon on desktop
2. Open the python script

infrascripts\testscripts\CallbackScale362\constants.py

Make sure that the maximum number of security groups/policy scale count & SG-Binding is set .

infrascripts\testscripts\CallbackScale362\constants.py

SCALE\_COUNT\_TOTAL = 1024

#Set SG\_BINDING\_COUNT less than 128

SG\_BINDING\_COUNT = 64

The script will

* create SCALE\_COUNT\_TOTAL of security groups and security policies.
* SG-Binding to SP for count of SG\_BINDING\_COUNT only.

***PS: Also validate the NSX IP address, username/password.***

1. Run python script CallbackScaleTests.py

Call Option-1 to do the setup. As part of the Callback test setup, the script creates 32 IPSets (one per Guest VM IP address), SCALE\_COUNT\_TOTAL(=1000) Security Groups, and SCALE\_COUNT\_TOTAL(=100) Security Policies. Every security policy includes 5 policy rules as mention above. The 5 rules are hard-coded in the script.

infrascripts\testscripts\CallbackScale362\CallbackScaleTests.py

1. CB Scale Config Setup

2. CB Scale Config Cleanup

Enter Your Choice: 1

Started Callback Scale Setup...

SG CREATION : success\_count=1027 failure\_count=0

SP CREATION : success\_count=1027 failure\_count=0

TOTAL SP CREATION time is : 1349.0105699188025 seconds ...

This will also generate a log file with more details.

Sample LOG contents: output\ cb\_script.log

…\output\ cb\_script.log

12-02-2018:18:39:06,390 INFO [CallbackScaleUtils.py:93 :cb\_scale\_ipset\_create] CREATED IPSET IpSet\_2077 object-id = ipset-757

12-02-2018:18:39:06,533 INFO [CallbackScaleUtils.py:93 :cb\_scale\_ipset\_create] CREATED IPSET IpSet\_2078 object-id = ipset-758

12-02-2018:18:39:06,706 INFO [CallbackScaleUtils.py:93 :cb\_scale\_ipset\_create] CREATED IPSET IpSet\_2079 object-id = ipset-759

12-02-2018:18:39:07,35 INFO [CallbackScaleUtils.py:230 :cb\_scale\_sec\_group\_create] CREATED SG Security\_Group\_5001 object-id = securitygroup-971

12-02-2018:18:39:07,278 INFO [CallbackScaleUtils.py:230 :cb\_scale\_sec\_group\_create] CREATED SG Security\_Group\_5002 object-id = securitygroup-972

12-02-2018:18:39:07,499 INFO [CallbackScaleUtils.py:230 :cb\_scale\_sec\_group\_create] CREATED SG Security\_Group\_5003 object-id = securitygroup-973

13-02-2018:14:24:05,489 ERROR [CallbackScaleUtils.py:898 :cb\_scale\_sec\_policy\_apply] APPLYING SP policy-1195 status code = 500

13-02-2018:14:24:05,884 ERROR [CallbackScaleUtils.py:898 :cb\_scale\_sec\_policy\_apply] APPLYING SP policy-1196 status code = 500

14-02-2018:16:28:43,481 INFO [CallbackScaleUtils.py:799 :cb\_scale\_sec\_policy\_delete] DELETED SP policy-499

14-02-2018:16:28:47,365 INFO [CallbackScaleUtils.py:799 :cb\_scale\_sec\_policy\_delete] DELETED SP policy-500

14-02-2018:16:28:50,681 INFO [CallbackScaleUtils.py:799 :cb\_scale\_sec\_policy\_delete] DELETED SP policy-501

1. The script will generate CSV files for time taken

Security Policy Create: output\ sp\_creation\_time.csv

Security Policy Apply: output\ sp\_apply\_time.csv

These files can be used to plot graph.

*P.S: I really don’t understand the value in this time-measurement for now.*

1. Validate NSX that all IP-sets, security groups, and security policies are created as expected.

If for some reason, any security-policy status is “Failed” then please manually resolve it.

Click the policy and in the pop-up hit “Resolve All”.

*PS: This might be due to the script running back-2-back REST APIs. Try to experiment with time.sleep() calls in all the APIs to tune appropriately.*

1. Run python script CallbackScaleTests.py

Call Option-2 to do the Cleanup. As part of the Callback test cleanup, the script will use all the object-ids for IP-sets, security-groups, security-policies already stored in the /output/ temporary files and does the cleanup.

***PS: Please don’t keep any .../input/ or .../output/ directory files open during the python scripts running time, otherwise the cleanup might not delete those temporary files and you might have issues in next runs.***

infrascripts\testscripts\CallbackScale362\CallbackScaleTests.py

1. CB Scale Config Setup

2. CB Scale Config Cleanup

Enter Your Choice: 2

Started Callback Scale Cleanup...

Completed Callback Scale Cleanup. Done

This will also update the log file with more details.

…\output\ cb\_script.log

1. Tear down the setup. Follow the steps for to vertical scale tests from document (instructions running performance testbed.doc) & run the Options 9,10,11 belongs. This will power off 32 Guest VMs and Remove them from VC inventory.
2. Congratulations, you have completed callback scale test of NSX scale test plan

## Appendix:

1. The scripts are written & tested with Trend Micro which does support both “Guest Introspection” and “Network Introspection” Security Policies. If the partner supports only one Category either “Guest Introspection” or “Network Introspection” Security Policies, then open the file CallbackScaleUtils.py and edit the XML payload.

Goto API cb\_scale\_sec\_policy\_create\_xml(self):

Use below section for “Guest Introspection” in the file:

<actionsByCategory>

<category>endpoint</category>

<action class="endpointSecurityAction">

<name>EndPoint Rule</name>

<clientHandle></clientHandle>

<extendedAttributes/>

<isUniversal>false</isUniversal>

<universalRevision>0</universalRevision>

<category>endpoint</category>

<executionOrder>1</executionOrder>

<isEnabled>true</isEnabled>

<isActionEnforced>true</isActionEnforced>

<serviceId>'''+ service\_obj\_id + '''</serviceId>

<invalidServiceId>false</invalidServiceId>

<serviceProfile>

<objectId>'''+ serviceprofile\_obj\_id + '''</objectId>

</serviceProfile>

<invalidServiceProfile>false</invalidServiceProfile>

</action>

</actionsByCategory>

Use below section for “Network Introspection” in the file:

<actionsByCategory>

<category>traffic\_steering</category>

<action class="trafficSteeringSecurityAction">

<name>inbound</name>

<clientHandle></clientHandle>

…

..

(lots of lines)

<invalidApplications>false</invalidApplications>

</action>

<action class="trafficSteeringSecurityAction">

<name>outbound</name>

<clientHandle></clientHandle>

<isUniversal>false</isUniversal>

…

….

(lots of lines)

<invalidApplications>false</invalidApplications>

</action>

Checkout files …\input\se\_policy\_payload\_both.xml & se\_policy\_payload\_ni.xml for both GI and NI payloads. Use this appropriately in the API cb\_scale\_sec\_policy\_create\_xml().