**VNF(Snort-IPS) Test Plan**

**Version: 1.0**

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Revision Date** | **Description** | **Approver** |
| 0.1 | 28-Feb-2018 | Initial draft |  |

Contents

[1. introduction - 3 -](#_Toc507598935)

[1.1 Purpose - 3 -](#_Toc507598936)

[1.2 VNF Description - 3 -](#_Toc507598937)

[1.3 Snort is an open source network intrusion detection system, capable of performing real time traffic analysis and packet logging on IP networks. It features analysis, content searching and matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, and much more. Snort uses a flexible rule language to describe traffic that it should collect or pass, as well as a detection engine. It has a modular real-time alerting capability, incorporating alerting and logging plugins for syslog and ASCII text files. The sniffing mechanism relies on libpcap. Snort provides capturing, logging and analyzing features and therefore outperforms tcpdump. Its broad functionality is by far more than we will need. The same argument as for nmap holds: the extraction of the relevant components may be a crucial task. It would be difficult to concentrate on the core features we are interested in and not to include stuff we do not need. - 3 -](#_Toc507598938)

[2. Scope - 3 -](#_Toc507598939)

[3. Topology - 3 -](#_Toc507598940)

[Topology1: Simple test (End to End) - 3 -](#_Toc507598941)

[Topology2: Multiple hosts covering single firewall interface - 4 -](#_Toc507598942)

[Topology3: isolation in multifirewall network - 4 -](#_Toc507598943)

[Topology4: horizontal scaling-load sharing - 5 -](#_Toc507598944)

[Topology5: horizontal scaling-active/standby - 5 -](#_Toc507598945)

[Topology6: vertical scaling - 6 -](#_Toc507598946)

[4. Test matrix - 6 -](#_Toc507598947)

[5. Test Cases - 7 -](#_Toc507598948)

[*5.1.* *SNORT ONBOARDING test cases* - 7 -](#_Toc507598949)

[*5.1.1.* *End to End packet flow VFIREWALL(snort)* - 8 -](#_Toc507598950)

[*5.1.2.* *End to End packet block VFIREWALL(snort)* - 8 -](#_Toc507598951)

[5.1.3. *Ingress and egress packet flow in NAT mode of VFIREWALL(snort)* - 9 -](#_Toc507598952)

[5.1.4. *http packet blocking/block Internet* - 10 -](#_Toc507598953)

[5.1.5. *All allowed/all drop* - 10 -](#_Toc507598954)

[5.1.6. *NIDS mode test* - 11 -](#_Toc507598955)

[**5.1.7.** *Sniffer mode test* - 12 -](#_Toc507598956)

[**5.1.8.** *Packet logger mode test* - 13 -](#_Toc507598957)

[**5.1.9.** *Packet logger Read mode test* - 14 -](#_Toc507598958)

[**5.1.10.** *Inline mode test* - 15 -](#_Toc507598959)

[**5.1.11.** *Passive mode test* - 15 -](#_Toc507598960)

[**5.1.12.** *Inline-test mode test* - 16 -](#_Toc507598961)

[5.1.13. *High Performance Configuration* - 17 -](#_Toc507598962)

[5.1.14. *Malware test* - 17 -](#_Toc507598963)

[5.1.15. *SSH test* - 18 -](#_Toc507598964)

[5.1.16. *Upgrade vnf* - 19 -](#_Toc507598965)

[5.1.17. *Multiple hosts covering a single firewall interface(http Test/allow TH1)* - 19 -](#_Toc507598966)

[5.1.18. *Multiple hosts covering a single firewall interface(http Test/allow TH2)* - 20 -](#_Toc507598967)

[5.1.19. *Multiple hosts covering a single firewall interface(http Test/all block)* - 20 -](#_Toc507598968)

[5.1.20. *Multiple hosts covering a single firewall interface(http Test/all allow)* - 21 -](#_Toc507598969)

[5.1.21. *Isolation policies in multifirewall network-snort1* - 21 -](#_Toc507598970)

[5.1.22. *Isolation policies in multifirewall network-snort2* - 22 -](#_Toc507598971)

[5.1.23. *Isolation policies in multifirewall network-all allow* - 23 -](#_Toc507598972)

[5.1.24. *Isolation policies in multifirewall network-all block* - 23 -](#_Toc507598973)

[5.1.25. *Isolation policies in multifirewall network-Private access blocking in snort2* - 24 -](#_Toc507598974)

[5.1.26. *Isolation policies in multifirewall network-public access blocking in snort2* - 24 -](#_Toc507598975)

[5.1.27. *Isolation policies in multifirewall network-host network blocking* - 25 -](#_Toc507598976)

[*5.1.28.* *Simple Horizontal scaling test for VFIREWALL(snort)* - 26 -](#_Toc507598977)

[*5.1.29.* *Openstack Horizontal Scaling test for Firewall with active-standby Firewalls* - 26 -](#_Toc507598978)

[*5.1.30.* *Simple Vertical scaling test for VFIREWALL(snort)* - 27 -](#_Toc507598979)

[*5.1.31.* *Multiple Configurations* - 28 -](#_Toc507598980)

[4 Test Tools - 29 -](#_Toc507598981)

[5 Software Used - 29 -](#_Toc507598982)

[6 References - 29 -](#_Toc507598983)

# introduction

## Purpose

This document outlines the validation scope and test cases to confirm the demonstration of running on OpenStack VIM over client Intel based hardware. The overall objective is to qualify different type VNFs in OpenStack setup on Intel based hardware platform. The VNFs verified is Snort.

## VNF Description

## Snort is an open source network intrusion detection system, capable of performing real time traffic analysis and packet logging on IP networks. It features analysis, content searching and matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, and much more. Snort uses a flexible rule language to describe traffic that it should collect or pass, as well as a detection engine. It has a modular real-time alerting capability, incorporating alerting and logging plugins for syslog and ASCII text files. The sniffing mechanism relies on libpcap. Snort provides capturing, logging and analyzing features and therefore outperforms tcpdump. Its broad functionality is by far more than we will need. The same argument as for nmap holds: the extraction of the relevant components may be a crucial task. It would be difficult to concentrate on the core features we are interested in and not to include stuff we do not need.

|  |  |  |  |
| --- | --- | --- | --- |
| vCPU | RAM(GB) | Disk Space(GB) | Network Interface |
| 1(\*) | 1(\*) | 10 | 3(\*) |

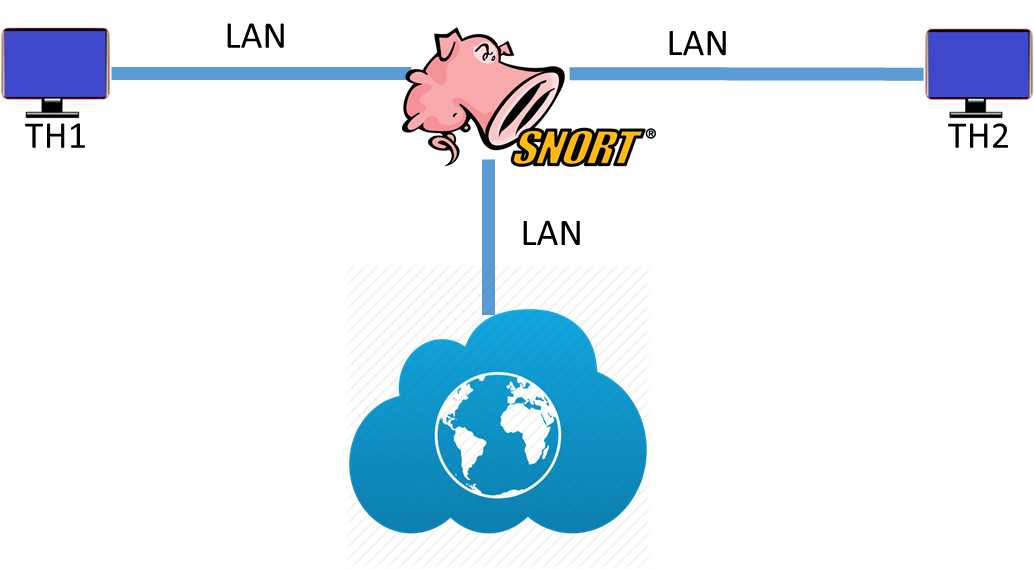
(\*) Variations are there in these resources during create/upgrade/downgrade

# Scope

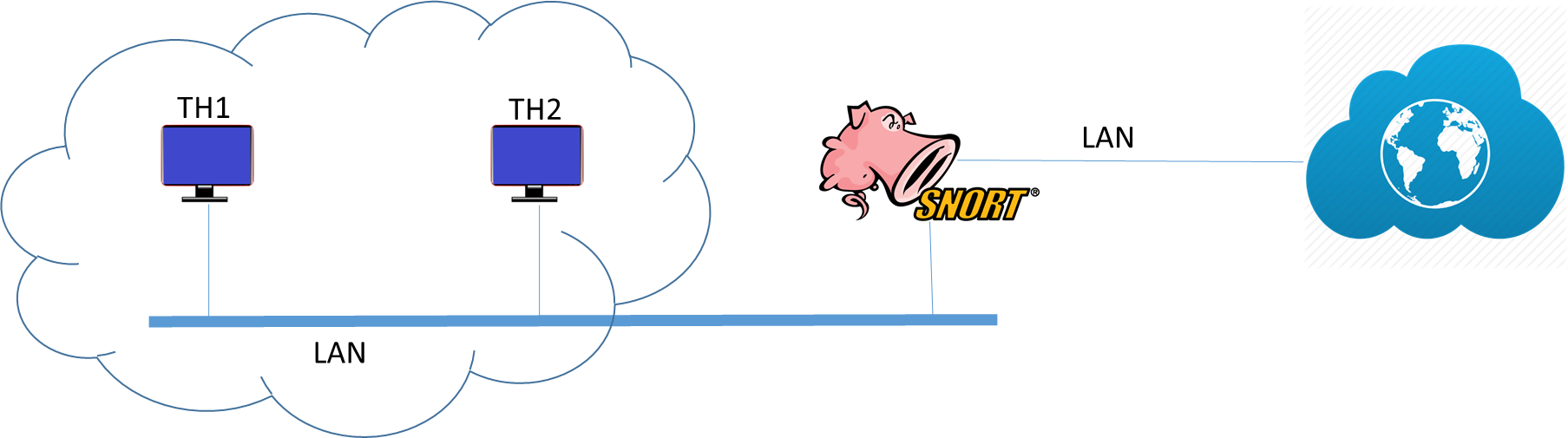
This test plan covers the VNF lifecycle management on OpenStack with two VNFs. Testing covers orchestration of VNF, upgrade/downgrade, snapshot, suspend, delete, migrate etc. The VNFs created will be validated with different flavors, access from external network with a network configuration.

# Topology

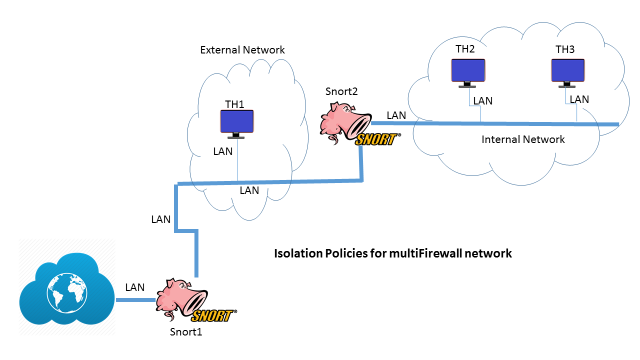
## Topology1: Simple test (End to End)



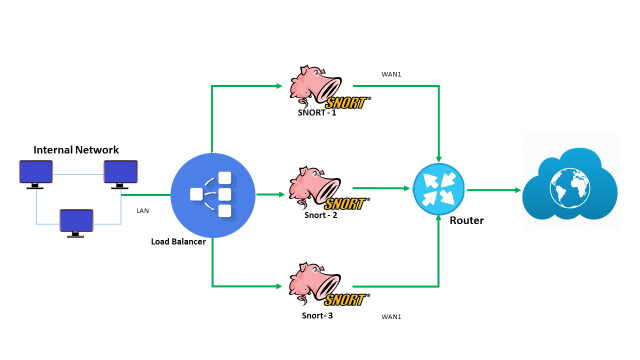
## Topology2: Multiple hosts covering single firewall interface



## Topology3: isolation in multifirewall network

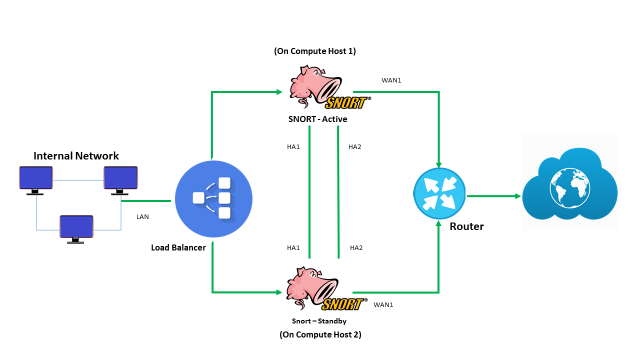


## Topology4: horizontal scaling-load sharing



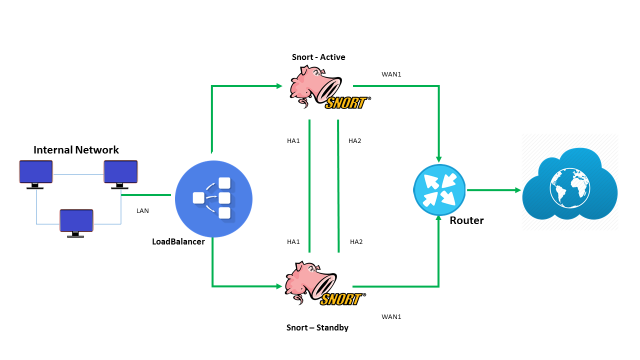
**Horizontal Scaling – Load Sharing**

## Topology5: horizontal scaling-active/standby



**Horizontal Scaling – Active / Standby**

## Topology6: vertical scaling



**Vertical Scaling**

# Test matrix

|  |  |
| --- | --- |
| **Test Area** | **Test cases** |
| END TO END PACKET FLOW VFIREWALL(SNORT) | SN\_VNF\_01 |
| END TO END PACKET BLOCK VFIREWALL(SNORT) | SN\_VNF\_02 |
| INGRESS AND EGRESS PACKET FLOW IN NAT MODE OF VFIREWALL(SNORT) | SN\_VNF\_03 |
| HTTP PACKET BLOCKING/BLOCK INTERNET | SN\_VNF\_04 |
| ALL ALLOWED/ALL DROP | SN\_VNF\_05 |
| NIDS MODE TEST | SN\_VNF\_06 |
| SNIFFER MODE TEST | SN\_VNF\_07 |
| PACKET LOGGER MODE TEST | SN\_VNF\_08 |
| PACKET LOGGER READ MODE TEST | SN\_VNF\_09 |
| INLINE MODE TEST | SN\_VNF\_10 |
| PASSIVE MODE TEST | SN\_VNF\_11 |
| INLINE-TEST MODE TEST | SN\_VNF\_12 |
| HIGH PERFORMANCE CONFIGURATION | SN\_VNF\_13 |
| MALWARE TEST | SN\_VNF\_14 |
| SSH TEST | SN\_VNF\_15 |
| UPGRADE VNF | SN\_VNF\_16 |
| MULTIPLE HOSTS COVERING A SINGLE FIREWALL INTERFACE(HTTP TEST/ALLOW TH1) | SN\_VNF\_17 |
| MULTIPLE HOSTS COVERING A SINGLE FIREWALL INTERFACE(HTTP TEST/ALLOW TH2) | SN\_VNF\_18 |
| MULTIPLE HOSTS COVERING A SINGLE FIREWALL INTERFACE(HTTP TEST/ALL BLOCK) | SN\_VNF\_19 |
| MULTIPLE HOSTS COVERING A SINGLE FIREWALL INTERFACE(HTTP TEST/ALL ALLOW) | SN\_VNF\_20 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-SNORT1 | SN\_VNF\_21 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-SNORT2 | SN\_VNF\_22 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-ALL ALLOW | SN\_VNF\_23 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-ALL BLOCK | SN\_VNF\_24 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-PRIVATE ACCESS BLOCKING IN SNORT2 | SN\_VNF\_25 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-PUBLIC ACCESS BLOCKING IN SNORT2 | SN\_VNF\_26 |
| ISOLATION POLICIES IN MULTIFIREWALL NETWORK-HOST NETWORK BLOCKING | SN\_VNF\_27 |
| SIMPLE HORIZONTAL SCALING TEST FOR VFIREWALL(SNORT) | SN\_VNF\_28 |
| OPENSTACK HORIZONTAL SCALING TEST FOR FIREWALL WITH ACTIVE-STANDBY FIREWALLS | SN\_VNF\_29 |
| SIMPLE VERTICAL SCALING TEST FOR VFIREWALL(SNORT) | SN\_VNF\_30 |
| MULTIPLE CONFIGURATIONS | SN\_VNF\_31 |

# Test Cases

## SNORT ONBOARDING test cases

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_01 | | | | |
| **Test Case Description:** | End to End packet flow VFIREWALL(snort) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the end-to-end packet flow from vm1 to vm2 through VFIREWALL(snort)(Router mode) with security policies enabled. | | | | |
| **Test Setup:** | As per diagram [Topology1: Snort Normal Mode](#_Topology1:_Installing_FortiGate) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create VNFD using YAML vnfd1.yaml * Verify all the VNFs are launched successfully as per the topology with mentioned configurations in yaml file. * Verify interface of VFIREWALL(snort) is connected to public network. * Verify lan interface of VFIREWALL(snort) is connected private network. * Configure a Default route in both VM1 And VM2 l to reach from VM1 to VM2 network via snort. * Configure a policy in VFIREWALL(snort) to allow all icmp packets from LAN. * Send icmp packets from Vm1 to VM2. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Vnf should be up and running * Traffic flow should be fine between two vms * Save the log via sniffer mode | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_02 | | | | |
| **Test Case Description:** | End to End packet block VFIREWALL(snort) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the end-to-end packet flow from vm1 to vm2 through VFIREWALL(snort)(Router mode) with security policies enabled. | | | | |
| **Test Setup:** | As per diagram [Topology1: Snort Normal Mode](#_Topology1:_Installing_FortiGate) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create VNFD using YAML vnfd1.yaml * Verify all the VNFs are launched successfully as per the topology with mentioned configurations in yaml file. * Verify interface of VFIREWALL(snort) is connected to public network. * Verify lan interface of VFIREWALL(snort) is connected private network. * Configure a Default route in both VM1 And VM2 l to reach from VM1 to VM2 network via snort. * Configure a policy in VFIREWALL(snort) to block all icmp packets from LAN. * Send icmp packets from Vm1 to VM2. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Vnf should be up and running * Traffic flow should be blocked between two vms * Save the log via sniffer mode | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_03 | | | | |
| **Test Case Description:** | Ingress and egress packet flow in NAT mode of VFIREWALL(snort) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the end-to-end packet flow in ingress and egress ports of VFIREWALL(snort)(Router mode) with security policies enabled. | | | | |
| **Test Setup:** | As per diagram [Topology1: Snort in NAT/Route Mode](#_Topology1:_Installing_FortiGate) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create VNFD using YAML vnfd1.yaml * Verify all the VNFs are launched successfully as per the topology with mentioned configurations in yaml file. * Verify interface of VFIREWALL(snort) is connected to public network. * Verify lan interface of VFIREWALL(snort) is connected private network. * Configure a Default route in both VM1 And VM2 l to reach from VM1 to VM2 network via snort. * Configure a policy in VFIREWALL(snort) to allow all tcp packets from LAN. * Send tcp packets from Vm1 to reach VM2. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * NSD&NS should be created successfully and should be ACTIVE. * VNFs should be up and running. * LAN interface should be configured with correct ip address. * Verify the routing is configured and available in RIB. * Configured policy should be available in VFIREWALL(snort). * The ingress functionalities of VFIREWALL(snort) should work as configured. * Save the log via sniffer mode | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_04 | | | | |
| **Test Case Description:** | http packet blocking/block Internet | | | | |
| **Test Purpose:** | The purpose of this test is to block/verify the http packet flow in ingress and egress ports of VFIREWALL(snort)(Router mode) with security policies enabled. | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Send http packet from vm1 to firewall. * Verify the http packet * Set rule to block http packet on firewall * Send http packet * Verify the http packet * Save the logs via sniffer mode | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * NSD&NS should be created successfully and should be ACTIVE. * VNFs should be up and running. * LAN interface should be configured with correct ip address. * Verify the routing is configured and available in RIB. * Configured policy should be available in VFIREWALL(snort). * The ingress functionalities of VFIREWALL(snort) should work as configured. | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_05 | | | | |
| **Test Case Description:** | All allowed/all drop | | | | |
| **Test Purpose:** | The purpose of this test is to verify the end-to-end packet flow in ingress and egress ports of VFIREWALL(snort) with multiple security policies enabled. | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Block all types of traffic in both nodes * Verify by sending traffic * Check and save logs via sniffer mode * Allow all types of traffic in both nodes * Verify by sending traffic * Check and save loga via sniffer mode | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Verify the packet is blocked * Check for the rule intact * Display from sniffer log * Verify the packet is allowed * Check for the rule intact * Display from sniffer log | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_06 | | | | |
| **Test Case Description:** | NIDS mode test | | | | |
| **Test Purpose:** | To enable Network Intrusion Detection System (NIDS) mode so that you don't record every single packet sent down the wire | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * ./snort -dev -l ./log -h 192.168.1.0/24 -c snort.conf * where snort.conf is the name of your snort configuration file. This will apply the rules configured in the snort.conf file to each packet to decide if an action based upon the rule type in the file should be taken. If you don't specify an output directory for the program, it will default to /var/log/snort. * NIDS Mode Output Options * -A fast * Fast alert mode. Writes the alert in a simple format with a timestamp, alert message, source and destination IPs/ports. * -A full * Full alert mode. This is the default alert mode and will be used automatically if you do not specify a mode. * -A unsock * Sends alerts to a UNIX socket that another program can listen on. * -A none * Turns off alerting. * -A console * Sends ``fast-style'' alerts to the console (screen). * -A cmg * Generates ``cmg style'' alerts. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * NIDS alert mode should be enabled * Output should come in expected way | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_07 | | | | |
| **Test Case Description:** | * + 1. *Sniffer mode test* | | | | |
| **Test Purpose:** | To enable sniffer mode so that you can print tcp/ip headers to the screen,packet data,data link layer headers | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the topology using the yaml file stored in local * Install the snort * To print only tcp/ip headers run ./snort –v command * To print tcp/ip headers along with packet data run ./snort –vd * To print above two along with data link layer run ./snort -vde | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The outputs from each command should be printed successfully | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_08 | | | | |
| **Test Case Description:** | * + 1. *Packet logger mode test* | | | | |
| **Test Purpose:** | To enable packet logger mode so that you can record the packets to the disk, you need to specify a logging directory and Snort will automatically know to go into packet logger mode. | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the topology using the yaml file stored in local * Install the snort * To store the log we need to specify the log directory, if it run on a switch we can point it to store the logs into another machine. * ./snort -dev -l ./log -h 192.168.1.0/24 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The log should be successfully stored in specified directory | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_09 | | | | |
| **Test Case Description:** | * + 1. *Packet logger Read mode test* | | | | |
| **Test Purpose:** | To enable packet logger mode so that you can record the packets to the disk, you need to specify a logging directory and Snort will automatically know to go into packet logger mode. | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the topology using the yaml file stored in local * Install the snort * To store the log we need to specify the log directory, if it run on a switch we can point it to store the logs into another machine. * ./snort -dev -l ./log -h 192.168.1.0/24 * Now try to read the log file by * ./snort -dvr packet.log http | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * This above command should only read the http logs from the file packet.log | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_10 | | | | |
| **Test Case Description:** | * + 1. *Inline mode test* | | | | |
| **Test Purpose:** | * When Snort is in Inline mode, it acts as an IPS allowing drop rules to trigger | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the topology using the yaml file stored in local * Install the snort * Enable inline mode   snort -Q   * config policy\_mode:inline | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The inline mode should be activated properly | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_11 | | | | |
| **Test Case Description:** | * + 1. *Passive mode test* | | | | |
| **Test Purpose:** | * When Snort is in Passive mode, it acts as a IDS. Drop rules are not loaded (without –treat-drop-as-alert). | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the topology using the yaml file stored in local * Install the snort * Enable passive mode   snort -Q   * config policy\_mode:tap | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The passive mode should be activated properly | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_12 | | | | |
| **Test Case Description:** | * + 1. *Inline-test mode test* | | | | |
| **Test Purpose:** | Inline-Test mode simulates the inline mode of snort, allowing evaluation of inline behavior without affecting traffic. The drop rules will be loaded and will be triggered as a Wdrop (Would Drop) alert. | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the topology using the yaml file stored in local * Install the snort * Enable inline-test mode   snort --enable-inline-test   * config policy\_mode:inline\_test | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The inline-test mode should be activated properly | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_13 | | | | |
| **Test Case Description:** | High Performance Configuration | | | | |
| **Test Purpose:** | The purpose of this test to make snort to go first like keep up with a 1000 Mbps connection | | | | |
| **Test Setup:** | As per diagram Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Set up the Ethernet link connecting to snort as 1000mbps * Send packets from TH1 to TH2 * If you want a text file that’s easily parsed, but still somewhat fast, try using binary logging with the “fast” output mechanism. * This will log packets in tcpdump format and produce minimal alerts. For example: * ./snort -b -A fast -c snort.conf | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Logging of packets will be successful in snort | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_14 | | | | |
| **Test Case Description:** | Malware test | | | | |
| **Test Purpose:** | The purpose of this test is to verify the malware in packet flow and block the same | | | | |
| **Test Setup:** | As per diagram [Topology1](#_Topology4:_Creating_Security) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Send a customized/regular malware traffic * Block the malware traffic by applying snort rules | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Check for the malware has been blocked * Save it in sniffer log | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_15 | | | | |
| **Test Case Description:** | SSH test | | | | |
| **Test Purpose:** | The purpose of this test is to verify the malware in packet flow and block the same | | | | |
| **Test Setup:** | As per diagram [Topology1](#_Topology4:_Creating_Security) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Try accessing from vm1 to vm2 via ssh * Check for the connection * Block ssh connection in snort from vm1 * Again access it from vm1 to vm2 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Vm1 should be able to access the vm2 in first place * After blocking the vm1 shouldnot be able to establish ssh connection with vm2 | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_16 | | | | |
| **Test Case Description:** | Upgrade vnf | | | | |
| **Test Purpose:** | The purpose of this test is to verify the upgradation of a VFIREWALL(snort) vnf | | | | |
| **Test Setup:** | As per diagram [Topology1](#_Topology4:_Creating_Security) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Select a higher version of centos to upgrade and put it in a yaml * For e.g:-cirros2.8.0 to cirros3.0.0 * Using the upgrade command the vm will be upgraded | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The updated VM should be reachable with all interfaces up | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_17 | | | | |
| **Test Case Description:** | Multiple hosts covering a single firewall interface(http Test/allow TH1) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the filtering of packets(http) in this case to block a single host and verify the blocking across multi host network | | | | |
| **Test Setup:** | As per diagram Topology2 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * The two THs including snort VFIREWALL(snort) should be up and ruuning * Enable rule in snort to block http traffic from TH1 and allow from TH2 and send http packet from both hosts | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The http packet coming from the TH1 should be blocked in first scenario and will be allowed for TH2 | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_18 | | | | |
| **Test Case Description:** | Multiple hosts covering a single firewall interface(http Test/allow TH2) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the filtering of packets(http) in this case to block a single host and verify the blocking across multi host network | | | | |
| **Test Setup:** | As per diagram Topology2 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * The two THs including snort VFIREWALL(snort) should be up and ruuning * Enable rule in snort to block http traffic from TH2 and allow from TH1 and send http packet from both hosts | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The http packet coming from the TH2 should be blocked in first case and will be allowed for TH2 | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_19 | | | | |
| **Test Case Description:** | Multiple hosts covering a single firewall interface(http Test/all block) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the filtering of packets(http) in this case to block a single host and verify the blocking across multi host network | | | | |
| **Test Setup:** | As per diagram Topology2 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * The two THs including snort VFIREWALL(snort) should be up and ruuning * Enable blocking for both host in the network and send http packet from both hosts | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Http traffic should be blocked for both hosts for this scenario | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_20 | | | | |
| **Test Case Description:** | Multiple hosts covering a single firewall interface(http Test/all allow) | | | | |
| **Test Purpose:** | The purpose of this test is to verify the filtering of packets(http) in this case to block a single host and verify the blocking across multi host network | | | | |
| **Test Setup:** | As per diagram Topology2 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * The two THs including snort VFIREWALL(snort) should be up and ruuning * Disable blocking for both host in the network and send http packet from both hosts | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Http traffic should be unblocked for both hosts for this scenario | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_21 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-snort1 | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * Block http in snort1 for TH1 * Allow http in snort2 for TH1 * Send http packet from TH1 towards snort1 and snort2 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The http packet should be blocked in snort1 * The http packet should be allowed in snort2 * The applied policies in both snorts shouldn’t be hampered the overall functionality | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_22 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-snort2 | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * Block http in snort2 for TH1 * Allow http in snort1 for TH1 * Send http packet from TH1 towards snort1 and snort2 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The http packet should be blocked in snort2 * The http packet should be allowed in snort1 * The applied policies in both snorts shouldn’t be hampered the overall functionality | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_23 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-all allow | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * Allow http in snort2 for TH1 * Allow http in snort1 for TH1 * Send http packet from TH1 towards snort1 and snort2 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The http packet should be allowed in snort2 * The http packet should be allowed in snort1 * The applied policies in both snorts shouldn’t be hampered the overall functionality | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_24 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-all block | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * block http in snort2 for TH1 * block http in snort1 for TH1 * Send http packet from TH1 towards snort1 and snort2 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The http packet should be blocked in snort2 * The http packet should be blocked in snort1 * The applied policies in both snorts shouldn’t be hampered the overall functionality | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_25 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-Private access blocking in snort2 | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * Block TH3 in snort2 * Access/send packets to TH3 from TH1 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The host TH1 shouldn’t be able to access TH3 * All snort functionality should be working fine | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_26 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-public access blocking in snort2 | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * Block TH1 in snort2 * Access/send packets to TH1 from TH2 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The host TH2 shouldn’t be able to access TH1 * All snort functionality should be working fine | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_27 | | | | |
| **Test Case Description:** | Isolation policies in multifirewall network-host network blocking | | | | |
| **Test Purpose:** | The purpose of this test is to verify the policy check within multi firewall network which can be used for isolation with use of http traffic | | | | |
| **Test Setup:** | As per diagram Topology3 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file for NFV. | | | | |
| **Procedure:** | * Create the topology with the specified yml file * Block TH2, TH3 network in snort2 * Access/send packets to TH2,TH3 from TH1 | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The host TH2/3 shouldn’t be able to accessed from TH1 * All snort functionality should be working fine | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_28 | | | | |
| **Test Case Description:** | Simple Horizontal scaling test for VFIREWALL(snort) | | | | |
| **Test Purpose:** | The purpose of this test is to verify Openstack horizontal scaling of Snort VFIREWALL(snort) with multiple instance | | | | |
| **Test Setup:** | As per diagram – [Topology4](#_Topology10:_Horizontal_Scaling) | | | | |
| **Prerequisites:** | * Have a snort image in openstack * OpenStack services are up and running. * Required yaml files are attached in separate file. * Heat template will be used for Horizontal Scaling. * IP addresses are dynamically assigned by DHCP once interfaces are attached to networks. | | | | |
| **Procedure:** | * In Openstack, create a VNF instance by selecting the snort image and with CPU=1 * Access the snort instance and verify the verify all the functionalities * Assuming the load of the snort is loaded with more users, create another VNF instance with the same image and with flavor CPU=1 * Have load balancer VNF created and make configuration to distribute the traffic between the two snort instances   + In this way, by having the same license, we can create multiple instances by dividing the CPU and distribute the traffic * Now, again the steps to be repeated for a maximum of 4 CPUs utilization * After this, verify accessing of snort web page from multiple users * Delete NSD and NS. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Traffic should distribute b/w the available snort VNF instances * Page loading time should be faster | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_29 | | | | |
| **Test Case Description:** | Openstack Horizontal Scaling test for Firewall with active-standby Firewalls | | | | |
| **Test Purpose:** | The purpose of this test is to verify Openstack Horizontal Scaling with Active Standby Firewalls. | | | | |
| **Test Setup:** | As per diagram - [Topology5](#_Topology11:_Horizontal_Scaling) | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file. * Heat template will be used for Horizontal Scaling. * IP addresses are dynamically assigned by DHCP once interfaces are attached to networks. | | | | |
| **Procedure:** | * In Openstack, create one VFIREWALL(snort) Firewall 1 using HEAT template Resource type ResourceGroup. * Create one load balancer using HEAT template Resource type LoadBalancer or HAProxy called Server. * Create two routers (Router 1 and Router 2) and End-Server as destination server. * Create NSD using above instances and one user (Ostinato). * Verify NSD is created successfully. * Create and verify NS as shown in topology and apply its configuration. * As per the topology, let’s assume Firewall 1 is active and all traffic is going through it. * Create one or more VFIREWALL(snort)(s) (Firewall 2 in topology) as in step 1 and make it standby. * Send traffic from User (Ostinato) to End\_Server and observe the behavior of Load balancer. * Delete NSD and NS. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Configuration successful in step 1-4, 6- 8, 10. * Traffic flows perfectly in step 9 through Firewall 1. * Deletion of NSD&NS should be successful in step 10. | | | | |
| **Configuration:** | Server Resource Type - OS::Heat::ResourceGroup  Load Balancer Resource Type - OS::Neutron::LoadBalancer or OS::Nova::Server | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_30 | | | | |
| **Test Case Description:** | Simple Vertical scaling test for VFIREWALL(snort) | | | | |
| **Test Purpose:** | The purpose of this test is to verify Openstack vertical scaling of Snort VFIREWALL(snort). | | | | |
| **Test Setup:** | As per diagram - Topology6 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file. * Heat template will be used for vertical Scaling. * IP addresses are dynamically assigned by DHCP once interfaces are attached to networks. | | | | |
| **Procedure:** | * In Openstack, create a VNF by selecting the Snort image and with lower CPU/RAM/HDD size * Access the Snort instance and verify the verify all the functionalities * Access the Snort page with multiple users so that the response time of the UI is decreasing * Now, Upgrade vSnort VNF instance with a bigger flavor of CPU/RAM/HDD * After this, accessing the Snort web page from multiple users * Delete NSD and NS. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * Once the VNF is upgraded, the access to web page should not give any time out * Page loading time should be faster | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID:** | SN\_VNF\_31 | | | | |
| **Test Case Description:** | Multiple Configurations | | | | |
| **Test Purpose:** | Snort now supports multiple configurations based on VLAN Id or IP subnet within a single instance of Snort. | | | | |
| **Test Setup:** | As per diagram - Topology1 | | | | |
| **Prerequisites:** | * OpenStack services are up and running. * Required yaml files are attached in separate file. | | | | |
| **Procedure:** | * Allow administrators to specify multiple snort configuration files and bind each configuration to one or more VLANs or subnets rather than running one Snort for each configuration required. * Create a new configuration instance within snort. VLANs/Subnets not bound to any specific configuration will use the default configuration. Each configuration can have different preprocessor settings and detection rules. * Create multiple network in snort * Config specific elements like policy\_id,mode,version * Connect TH1 and TH2 in separate vlans * Send http packet to each vlan from each testing host. | | | | |
| **Checks:** |  | | | | |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
|  |  | **Pass:** | ☐ | **Fail:** | ☐ |
| **Expected Results:** | * The snort functionality should work properly * The vlan id should be set as per the command * Reachability should be happening within the network | | | | |
| **Configuration:** |  | | | | |
| **Results:** |  | | | | |
| **Reason for Failure:** |  | | | | |
| **Remarks:** |  | | | | |

# Test Tools

|  |  |
| --- | --- |
| YAML files |  |

# Software Used

|  |  |
| --- | --- |
| VNF | Snort |
| Host VMs | Ubuntu 16.04, CentOS |
| OpenStack | Release: (ocata) |
| Installation Procedure | <https://www.upcloud.com/support/installing-snort-on-centos/>  <https://stackoverflow.com/questions/25236662/snort-installation-error-on-centos-6-5> |

# References

1. <https://docs.openstack.org/ocata/>
2. <https://www.snort.org/>
3. <https://en.wikipedia.org/wiki/Snort_(software)>