BMWG
Internet Draft
Intended status: Informational
Expires: January 2017

S. Kommu
VMware
B. Basler
VMware
J. Rapp
VMware

July 8, 2016

Considerations for Benchmarking Network Virtualization Platforms
draft-bmwg-nvp-00.txt

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on January 8, 2009.

Copyright Notice

Abstract

Current network benchmarking methodologies are focused on physical networking components and do not consider the actual application layer traffic patterns and hence do not reflect the traffic that virtual networking components work with. The purpose of this document is to distinguish and highlight benchmarking considerations when testing and evaluating virtual networking components in the data center.

Table of Contents

1. Introduction

Datacenter virtualization that includes both compute and network virtualization is growing rapidly as the industry continues to look for ways to improve productivity, flexibility and at the same time cut costs. Network virtualization, is comparatively new and expected to grow tremendously similar to compute virtualization. There are multiple vendors and solutions out in the market, each with their own benchmarks to showcase why a particular solution is better than another. Hence, the need for a vendor and product agnostic way to benchmark multivendor solutions to help with comparison and make informed decisions when it comes to selecting the right network virtualization solution.

Applications traditionally have been segmented using VLANs and ACLs between the VLANs. This model does not scale because of the 4K scale limitations of VLANs. Overlays such as VXLAN were designed to address the limitations of VLANs

With VXLAN, applications are segmented based on VXLAN encapsulation (specifically the VNI field in the VXLAN header), which is similar to VLAN ID in the 802.1Q VLAN tag, however without the 4K scale limitations of VLANs. For a more detailed discussion on this subject please refer RFC 7364 "Problem Statement: Overlays for Network Virtualization".

VXLAN is just one of several Network Virtualization Overlays(NVO). Some of the others include STT, Geneve and NVGRE. . STT and Geneve have expanded on the capabilities of VXLAN. Please refer IETF's nvo3 working group < https://datatracker.ietf.org/wg/nvo3/documents/> for more information.

Modern application architectures, such as Micro-services, are going beyond the three tier app models such as web, app and db. Benchmarks MUST consider whether the proposed solution is able to scale up to the demands of such applications and not just a three-tier architecture.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

3. Definitions

3.1. System Under Test (SUT)

Traditional hardware based networking devices generally use the device under test (DUT) model of testing. In this model, apart from any allowed configuration, the DUT is a black box from a testing perspective. This method works for hardware based networking devices since the device itself is not influenced by any other components outside the DUT.

Virtual networking components cannot leverage DUT model of testing as the DUT is not just the virtual device but includes the hardware components that were used to host the virtual device

Hence SUT model MUST be used instead of the traditional device under test

With SUT model, the virtual networking component along with all software and hardware components that host the virtual networking component MUST be considered as part of the SUT.

Virtual networking components may also work with higher level TCP segments such as TSO. In contrast, all physical switches and routers, including the ones that act as initiators for NVOs, work with L2/L3 packets.

Please refer to section 5 Figure 1 for a visual representation of System Under Test in the case of Intra-Host testing and section 5 Figure 2 for System Under Test in the case of Inter-Host testing

3.2. Network Virtualization Platform

This document does not focus on Network Function Virtualization.

Network Function Virtualization focuses on being independent of networking hardware while providing the same functionality. In the case of NFV, traditional benchmarking methodologies recommended by IETF may be used. Considerations for Benchmarking Virtual Network Functions and Their Infrastructure IETF document addresses benchmarking NFVs.

Network Virtualization Platforms, apart from providing hardware agnostic network functions, also leverage performance optimizations provided by the TCP stacks of hypervisors.

Network Virtualization Platforms are architected differently when compared to NFV and are not limited by packet size constraints via MTU that exist for both NFV and Hardware based network platforms.

NVPs leverage TCP stack optimizations such as TSO that enables NVPs to work with much larger payloads of 64K unlike their counterparts such as NFVs.

Because of the difference in the payload and thus the overall segment sizes, normal benchmarking methods are not relevant to the NVPs.

Instead, newer methods that take into account the built in advantages of TCP provided optimizations MUST be used for testing Network Virtualization Platforms.

3.3. Micro-services

Traditional monolithic application architectures such as the three tier web, app and db architectures are hitting scale and deployment limits for the modern use cases.

Micro-services make use of classic unix style of small app with single responsibility.

These small apps are designed with the following characteristics:

- Each application only does one thing - like unix tools
- Small enough that you could rewrite instead of maintain
- Embedded with a simple web container
- Packaged as a single executable
- Installed as daemons
- Each of these applications are completely separate
- Interact via uniform interface
    - REST (over HTTP/HTTPS) being the most common

With Micro-services architecture, a single web app of the three tier application model could now have 100s of smaller apps dedicated to do just one job.

These 100s of small one responsibility only services will MUST be secured into their own segment - hence pushing the scale boundaries of the overlay from both simple segmentation perspective and also from a security perspective

4. Scope

This document does not address Network Function Virtualization has been covered already by previous IETF documents (https://datatracker.ietf.org/doc/draft-ietf-bmwg-virtual-net/?include_text=1) the focus of this document is Network Virtualization Platform where the network functions are an intrinsic part of the hypervisor’s TCP stack, working closer to the application layer and leveraging performance optimizations such TSO/RSS provided by the TCP stack and the underlying hardware.

4.1. Virtual Networking for Datacenter Applications

While virtualization is growing beyond the datacenter, this document focuses on the virtual networking for east-west traffic within the datacenter applications only. For example, in a three tier app such web, app and db, this document focuses on the east-west traffic between web and app. It does not address north-south web traffic accessed from outside the datacenter. A future document would address north-south traffic flows.

This document addresses scale requirements for modern application architectures such as Micro-services to consider whether the proposed solution is able to scale up to the demands of micro-services application models that basically have 100s of small services communicating on some standard ports such as http/https using protocols such as REST

4.2. Interaction with Physical Devices

Virtual network components cannot be tested independent of other components within the system. Example, unlike a physical router or a firewall, where the tests can be focused directly solely on the device, when testing a virtual router or firewall, multiple other devices may become part of the system under test. Hence the characteristics of these other traditional networking switches and routers, LB, FW etc. MUST be considered.

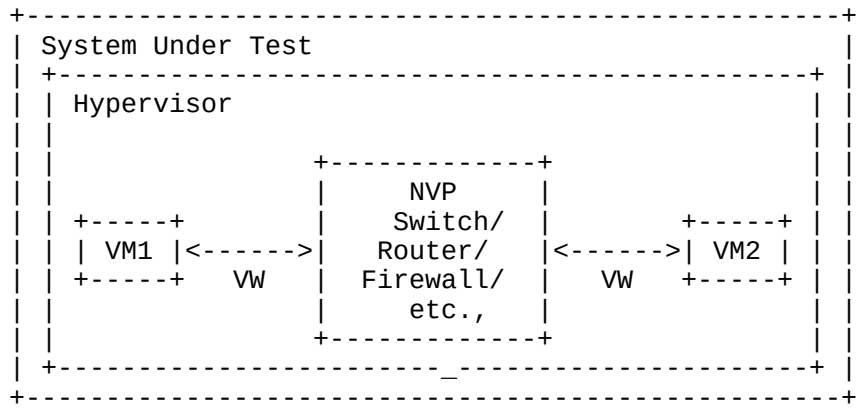
- Hashing method used
- Over-subscription rate
- Throughput available
- Latency characteristics

5. Interaction with Physical Devices

In virtual environments, System Under Test (SUT) may often share resources and reside on the same Physical hardware with other components involved in the tests. Hence SUT MUST be clearly defined. In this tests, a single hypervisor may host multiple servers, switches, routers, firewalls etc.,

Intra host testing: Intra host testing helps in reducing the number of components involved in a test. For example, intra host testing would help focus on the System Under Test, logical switch and the hardware that is running the hypervisor that hosts the logical switch, and eliminate other components. Because of the nature of virtual infrastructures and multiple elements being hosted on the

same physical infrastructure, influence from other components cannot be completely ruled out. For example, unlike in physical infrastructures, logical routing or distributed firewall MUST NOT be benchmarked independent of logical switching. System Under Test definition MUST include all components involved with that particular test.

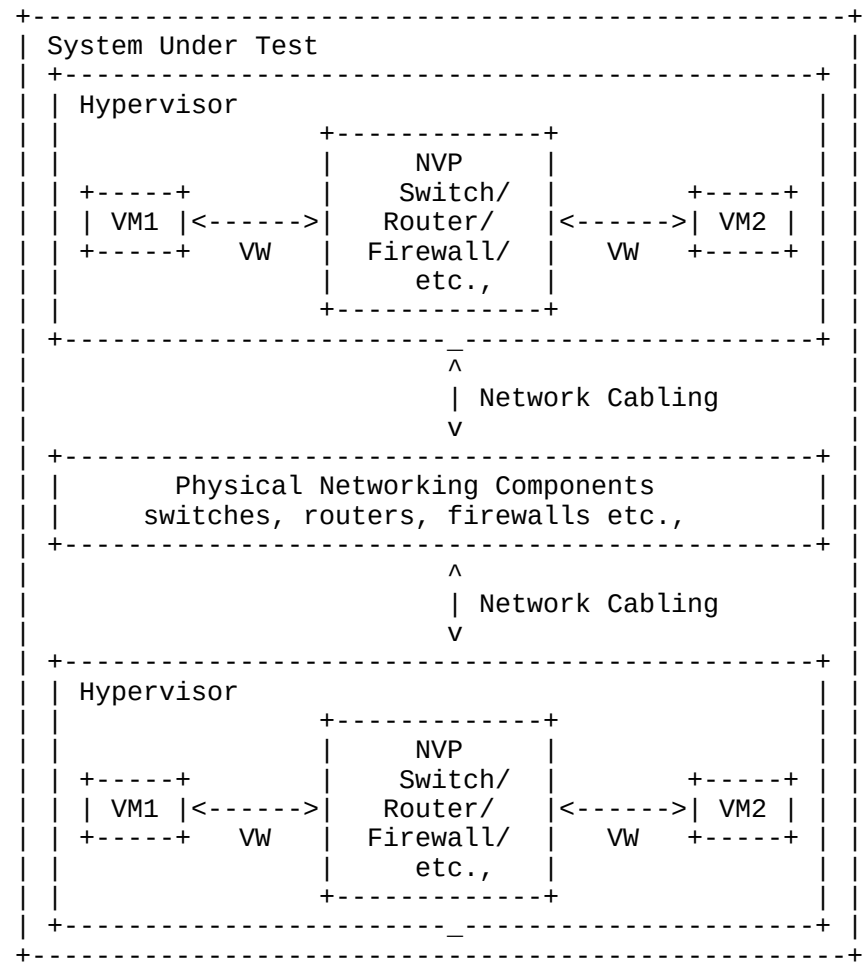


Legend
VM: Virtual Machine
VW: Virtual Wire

Figure 1 Intra-Host System Under Test

Inter host testing: Inter host testing helps in profiling the underlying network interconnect performance. For example, when testing Logical Switching, inter host testing would not only test the logical switch component but also any other devices that are part of the physical data center fabric that connects the two hypervisors. System Under Test MUST be well defined to help with repeatability of tests. System Under Test definition in the case of inter host testing, MUST include all components, including the underlying network fabric.

Figure 2 is a visual representation of system under test for inter-host testing



Legend
VM: Virtual Machine
VW: Virtual Wire

Figure 2 Inter-Host System Under Test

Virtual components have a direct dependency on the physical infrastructure that is hosting these resources. Hardware characteristics of the physical host impact the performance of the virtual components. The components that are being tested and the impact of the other hardware components within the hypervisor on the performance of the SUT MUST be documented. Virtual component performance is influenced by the physical hardware components within

the hypervisor. Access to various offloads such as TCP segmentation offload, may have significant impact on performance. Firmware and driver differences may also significantly impact results based on whether the specific driver leverages any hardware level offloads offered. Hence, all physical components of the physical server running the hypervisor that hosts the virtual components MUST be documented along with the firmware and driver versions of all the components used to help ensure repeatability of test results. For example, BIOS configuration of the server MUST be documented as some of those changes are designed to improve performance. Please refer to Appendix A for a partial list of parameters to document.

5.1. Server Architecture Considerations

When testing physical networking components, the approach taken is to consider the device as a black-box. With virtual infrastructure, this approach would no longer help as the virtual networking components are an intrinsic part of the hypervisor they are running on and are directly impacted by the server architecture used. Server hardware components define the capabilities of the virtual networking components. Hence, server architecture MUST be documented in detail to help with repeatability of tests. And the entire hardware and software components become the SUT.

5.1.1. Frame format/sizes within the Hypervisor

Maximum Transmission Unit (MTU) limits physical network component’s frame sizes. The most common max supported MTU for physical devices is 9000. However, 1500 MTU is the standard. Physical network testing and NFV uses these MTU sizes for testing. However, the virtual networking components that live inside a hypervisor, may work with much larger segments because of the availability of hardware and software based offloads. Hence, the normal smaller packets based testing is not relevant for performance testing of virtual networking components. All the TCP related configuration such as TSO size, number of RSS queues MUST be documented along with any other physical NIC related configuration.

Virtual network components work closer to the application layer then the physical networking components. Hence virtual network components work with type and size of segments that are often not the same type and size that the physical network works with. Hence, testing virtual network components MUST be done with application layer segments instead of the physical network layer packets.

5.1.2. Baseline testing with Logical Switch

Logical switch is often an intrinsic component of the test system along with any other hardware and software components used for

testing. Also, other logical components cannot be tested independent of the Logical Switch.

5.1.3. Tunnel encap/decap outside the hypervisor

Logical network components may also have performance impact based on the functionality available within the physical fabric. Physical fabric that supports NVO encap/decap is one such case that has considerable impact on the performance. Any such functionality that exists on the physical fabric MUST be part of the test result documentation to ensure repeatability of tests. In this case SUT MUST include the physical fabric

5.1.4. SUT Hypervisor Profile

Physical networking equipment has well defined physical resource characteristics such as type and number of ASICs/SoCs used, amount of memory, type and number of processors etc., Virtual networking components' performance is dependent on the physical hardware that hosts the hypervisor. Hence the physical hardware usage, which is part of SUT, for a given test MUST be documented. Example, CPU usage when running logical router.

CPU usage changes based on the type of hardware available within the physical server. For example, TCP Segmentation Offload greatly reduces CPU usage by offloading the segmentation process to the NIC card on the sender side. Receive side scaling offers similar benefit on the receive side. Hence, availability and status of such hardware MUST be documented along with actual CPU/Memory usage when the virtual networking components have access to such offload capable hardware.

Following is a partial list of components that MUST be documented - both in terms of what's available and also what's used by the SUT -

- CPU - type, speed, available instruction sets (e.g. AES-NI)
- Memory - type, amount
- Storage - type, amount
- NIC Cards - type, number of ports, offloads available/used, drivers, firmware (if applicable), HW revision
- Libraries such as DPDK if available and used
- Number and type of VMs used for testing and
  - vCPUs

- o RAM
- o Storage
- o Network Driver
- o Any prioritization of VM resources
- o Operating System type, version and kernel if applicable
- o TCP Configuration Changes - if any
- o MTU

. Test tool

- o Workload type
- o Protocol being tested
- o Number of threads
- o Version of tool

. For inter-hypervisor tests,

- o Physical network devices that are part of the test

. Note: For inter-hypervisor tests, system under test is no longer only the virtual component that is being tested but the entire fabric that connects the virtual components become part of the system under test.

6. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization of a Device Under Test/System Under Test (DUT/SUT) using controlled stimuli in a laboratory environment, with dedicated address space and the constraints specified in the sections above.

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the DUT/SUT.

Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

7. IANA Considerations

No IANA Action is requested at this time.

8. Conclusions

Network Virtualization Platforms, because of their proximity to the application layer and since they can take advantage of TCP stack optimizations, do not function on packets/sec basis. Hence, traditional benchmarking methods, while still relevant for Network Function Virtualization, are not designed to test Network Virtualization Platforms. Also, advances in application architectures such as micro-services, bring new challenges and need benchmarking not just around throughput and latency but also around scale. New benchmarking methods that are designed to take advantage of the TCP optimizations or needed to accurately benchmark performance of the Network Virtualization Platforms

9. References

9.1. Normative References

[RFC7364] T. Narten, E. Gray, D. Black, L. Fang, L. Kreeger, M. Napierala, "Problem Statement: Overlays for Network Virtualization", RFC 7364, October 2014, https://datatracker.ietf.org/doc/rfc7364/

[nv03] IETF, WG, Network Virtualization Overlays, < https://datatracker.ietf.org/wg/nvo3/documents/>

9.2. Informative References

[1] A. Morton " Considerations for Benchmarking Virtual Network Functions and Their Infrastructure", draft-ietf-bmwg-virtual-net-03, < https://datatracker.ietf.org/doc/draft-ietf-bmwg-virtual-net/?include_text=1>

Appendix A. Partial List of Parameters to Document

A.1. CPU

CPU Vendor

CPU Number

CPU Architecture

# of Sockets (CPUs)

# of Cores

Clock Speed (GHz)

Max Turbo Freq. (GHz)

Cache per CPU (MB)

# of Memory Channels

Chipset

Hyperthreading (BIOS Setting)

Power Management (BIOS Setting)

VT-d

A.2. Memory

Memory Speed (MHz)

DIMM Capacity (GB)

# of DIMMs

DIMM configuration

Total DRAM (GB)

A.3. NIC

Vendor

Model

Port Speed (Gbps)

Ports

PCIe Version

PCIe Lanes

Bonded

Bonding Driver

Kernel Module Name

Driver Version

VXLAN TSO Capable

VXLAN RSS Capable

Ring Buffer Size RX

Ring Buffer Size TX

A.4. Hypervisor

Hypervisor Name

Version/Build

Based on

Hotfixes/Patches

OVS Version/Build

IRQ balancing

vCPUs per VM

Modifications to HV

Modifications to HV TCP stack

Number of VMs

IP MTU

Flow control TX (send pause)

Flow control RX (honor pause)

Encapsulation Type

A.5. Guest VM

Guest OS & Version

Modifications to VM

IP MTU Guest VM (Bytes)

Test tool used

Number of NetPerf Instances

Total Number of Streams

Guest RAM (GB)

A.6. Overlay Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

A.7. Gateway Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

Authors' Addresses

Samuel Kommu
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: skommu@vmware.com

Benjamin Basler
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: bbasler@vmware.com

Jacob Rapp
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: jrapp@vmware.com

BMWG
Internet-Draft
Intended status: Informational
Expires: September 11, 2017

L. Huang, Ed.
R. Gu, Ed.
China Mobile
Bob. Mandeville
Iometrix
Brooks. Hickman
Spirent Communications
March 10, 2017

Benchmarking Methodology for Virtualization Network Performance
draft-huang-bmwg-virtual-network-performance-02

Abstract

As the virtual network has been widely established in IDC, the performance of virtual network has become a valuable consideration to the IDC managers. This draft introduces a benchmarking methodology for virtualization network performance based on virtual switch.

Status of This Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 11, 2017.

Table of Contents

1. Introduction

As the virtual network has been widely established in IDC, the performance of virtual network has become a valuable consideration to the IDC managers. This draft introduces a benchmarking methodology

for virtualization network performance based on virtual switch as the DUT.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Test Considerations

In a conventional test setup with Non-Virtual test ports, it is quite legitimate to assume that test ports provide the golden standard in measuring the performance metrics. If test results are sub optimal, it is automatically assumed that the Device-Under-Test (DUT) is at fault. For example, when testing throughput at a given frame size, if the test result shows less than 100% throughput, we can safely conclude that it’s the DUT that can not deliver line rate forwarding at that frame size(s). We never doubt that the tester can be an issue.

While in a virtual test environment where both the DUT as well as the test tool itself are software based, it’s quite a different story. Just like the DUT, tester running as software will have its own performance peak under various conditions.

There are two types of vSwitch according to different installation location. One is VM based vSwitch which is installed on a virtual machine, another is vSwitch directly installed on the host OS (similar to hypervisor).The latter is much more popular currently.

Tester’s calibration is essential in benchmarking testing in a virtual environment. Furthermore, to reduce the enormous combination of various conditions, tester must be calibrated with the exact same combination and parameter settings the user wants to measure against the DUT. A slight variation of conditions and parameter values will cause inaccurate measurements of the DUT.

While it is difficult to list the exact combination and parameter settings, the following table attempts to give the most common example how to calibrate a tester before testing a DUT (VSWITCH).

Sample calibration permutation:

Hypervisor Type	VM VNIC Speed	VM Memory CPU Allocation	Frame Size	Throughput
ESXi	1G/10G	512M/1Core	64	
			128	
			256	
			512	
			1024	
			1518	

Figure 1: Sample Calibration Permutation

Key points are as following:

a) The hypervisor type is of ultimate importance to the test results. VM tester(s) MUST be installed on the same hypervisor type as the DUT (VSWITCH). Different hypervisor type has an influence on the test result.

b) The VNIC speed will have an impact on testing results. Testers MUST calibrate against all VNIC speeds.

c) VM allocations of CPU resources and memory have an influence on test results.

d) Frame sizes will affect the test results dramatically due to the nature of virtual machines.

e) Other possible extensions of above table: The number of VMs to be created, latency reading, one VNIC per VM vs. multiple VM sharing one VNIC, and uni-directional traffic vs. bi-directional traffic.

Besides, the compute environment including the hardware should be also recorded.

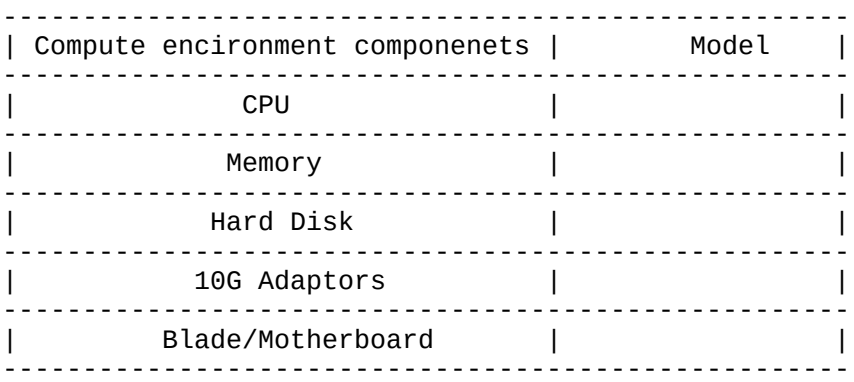


Figure 2: Compute Environment

It's important to confirm test environment for tester's calibration as close to the environment a virtual DUT (VSWITCH) involved in for the benchmark test. Key points which SHOULD be noticed in test setup are listed as follows.

1. One or more VM tester(s) need to be created for both traffic generation and analysis.

2. vSwitch has an influence on performance penalty due to extra resource occupation.

3. VNIC and its type is needed in the test setup to once again accommodate performance penalty when DUT (VSWITCH) is created.

In summary, calibration should be done in such an environment that all possible factors which may negatively impact test results should be taken into consideration.

4. Key Performance Indicators

We listed numbers of key performance indicators for virtual network below:

a) Throughput under various frame sizes: forwarding performance under various frame sizes is a key performance indicator of interest.

b) DUT consumption of CPU: when adding one or more VM(s), DUT (VSWITCH) will consume more CPU. Vendors can allocate appropriate CPU to reach the line rate performance.

c) DUT consumption of MEM: when adding one or more VM(s), DUT (VSWITCH) will consume more memory. Vendors can allocate appropriate MEM to reach the line rate performance.

d) Latency readings: Some applications are highly sensitive on latency. It's important to get the latency reading with respective to various conditions.

Other indicators such as VxLAN maximum supported by the virtual switch and so on can be added in the scene when VxLAN is needed.

5. Test Setup

The test setup is classified into two traffic models: Model A and Model B.

In traffic model A: A physical tester connects to the server which bears the DUT (VSWITCH) and Virtual tester to verify the benchmark of server.

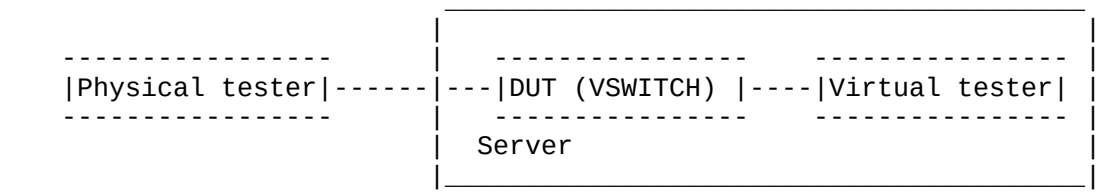


Figure 3: test model A

In traffic model B: Two virtual testers are used to verify the benchmark. In this model, two testers are installed in one server.

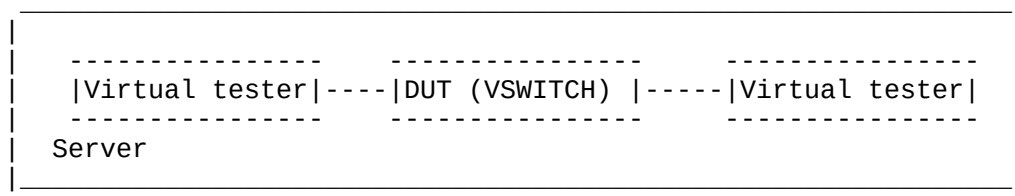


Figure 4: test model B

In our test, the test bed is constituted by physical servers of the Dell with a pair of 10GE NIC and physical tester. Virtual tester which occupies 2 vCPU and 8G MEM and DUT (VSWITCH) are installed in the server. 10GE switch and 1GE switch are used for test traffic and management respectively.

This test setup is also available in the VxLAN measurement.

6. Benchmarking Tests

6.1. Throughput

Unlike traditional test cases where the DUT and the tester are separated, virtual network test has been brought in unparalleled challenges. In virtual network test, the virtual tester and the DUT (VSWITCH) are in one server which means they are physically converged, so the test and DUT (VSWITCH) are sharing the same CPU and MEM resources of one server. Theoretically, the virtual tester's operation may have influence on the DUT (VSWITCH)'s performance. However, for the specialty of virtualization, this method is the only way to test the performance of a virtual DUT.

Under the background of existing technology, when we test the virtual switch's throughput, the concept of traditional physical switch CANNOT be applicable. The traditional throughput indicates the switches' largest forwarding capability, for certain bytes selected and under zero-packet-lose conditions. But in virtual environments, virtual variations on virtual network will be much greater than that of dedicated physical devices. As the DUT and the tester cannot be separated, it proves that the DUT (VSWITCH) realize such network performances under certain circumstances.

Therefore, we change the bytes in virtual environment to test the maximum value which we think of the indicator of throughput. It's conceivable that the throughput should be tested on both the test model A and B. The tested throughput has certain referential meanings to value the performance of the virtual DUT.

6.1.1. Objectives

The objective of the test is to determine the throughput of the DUT (VSWITCH), which the DUT can support.

6.1.2. Configuration parameters

Network parameters should be defined as follows:

a) the number of virtual tester (VMs)

b) the number of vNIC of virtual tester

c) the CPU type of the server

d) vCPU allocated for virtual tester (VMs)

e) memory allocated for virtual tester (VMs)

f) the number and rate of server NIC

6.1.3. Test parameters

a) test repeated times

b) test frame length

6.1.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch.

2. Increase the number of vCPU in the tester until the traffic has no packet loss.

3. Record the max throughput on VSwitch.

4. Change the frame length and repeat from step1 to step4.

6.1.5. Test result format

Byte	Throughput (Gbps)
64	
128	
256	
512	
1024	
1518	

Figure 5: test result format

6.2. Frame loss rate

Frame loss rate is also an important indicator in evaluating the performance of virtual switch.As is defined in RFC 1242, percentage of frames that should have been forwarded which actually fails to be forwarded due to lack of resources needs to be tested.Both model A

and model B are tested.Frame loss rate is an important indicator in evaluating the performance of virtual switches.

6.2.1. Objectives

The objective of the test is to determine the frame loss rate under different data rates and frame sizes..

6.2.2. Configuration parameters

Network parameters should be defined as follows:

a) the number of virtual tester (VMs)

b) the number of vNIC of virtual tester

c) the CPU type of the server

d) vCPU allocated for virtual tester (VMs)

e) memory allocated for virtual tester (VMs)

f) the number and rate of server NIC

6.2.3. Test parameters

a) test repeated times

b) test frame length

c) test frame rate

6.2.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch with the input frame changing from the maximum rate to the rate with no frame loss at reducing 10% intervals according to RFC 2544.

2. Record the input frame count and output count on VSwitch.

3. Calculate the frame loss percentage under different frame rate.

4. Change the frame length and repeat from step1 to step4.

6.2.5. Test result format

Byte	Maxmum rate (Gbps)	90% Maximum rate (Gbps)	80% Maximum rate (Gbps)	...	rate with no loss (Gbps)
64					
128					
256					
512					
1024					
1518					

Figure 6: test result format

6.3. CPU consumption

The objective of the test is to determine the CPU load of DUT(VSWITCH). The operation of DUT (VSWITCH) can increase the CPU load of host server. Different V-Switches have different CPU occupation. This can be an important indicator in benchmarking the virtual network performance.

6.3.1. Objectives

The objective of this test is to verify the CPU consumption caused by the DUT (VSWITCH).

6.3.2. Configuration parameters

Network parameters should be defined as follows:

a) the number of virtual tester (VMs)

b) the number of vNIC of virtual tester

c) the CPU type of the server

d) vCPU allocated for virtual tester (VMs)

e) memory allocated for virtual tester (VMs)

f) the number and rate of server NIC

6.3.3. Test parameters

a) test repeated times

b) test frame length

c) traffic rate

6.3.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch with certain traffic rate. The traffic rate could be different ratio of NIC’s speed.

2. Record vSwitch’s CPU usage on the host OS if no packets loss happens.

3. Change the traffic rate and repeat from step1 to step2.

4. Change the frame length and repeat from step1 to step3.

6.3.5. Test result format

Byte	Traffic Rate(Gbps)	CPU usage of vSwitch
64	50% of NIC speed	
	75%	
	90%	
128	50% of NIC speed	
	75%	
	90%	
~	~	~
1500	50% of NIC speed	
	75%	
	90%	

Figure 7: test result format

6.4. MEM consumption

The objective of the test is to determine the Memory load of DUT(VSWITCH). The operation of DUT (VSWITCH) can increase the Memory load of host server. Different V-Switches have different memory occupation. This can be an important indicator in benchmarking the virtual network performance.

6.4.1. Objectives

The objective of this test is to verify the memory consumption by the DUT (VSWITCH) on the Host server.

6.4.2. Configuration parameters

Network parameters should be defined as follows:

a) the number of virtual tester (VMs)

b) the number of vNIC of virtual tester

c) the CPU type of the server

d) vCPU allocated for virtual tester (VMs)

e) memory allocated for virtual tester (VMs)

f) the number and rate of server NIC

6.4.3. Test parameters

a) test repeated times

b) test frame length

6.4.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch with certain traffic rate. The traffic rate could be different ratio of NIC’s speed.

2. Record vSwitch’s MEM usage on the host OS if no packets loss happens.

3. Change the traffic rate and repeat from step1 to step2.

4. Change the frame length and repeat from step1 to step3.

6.4.5. Test result format

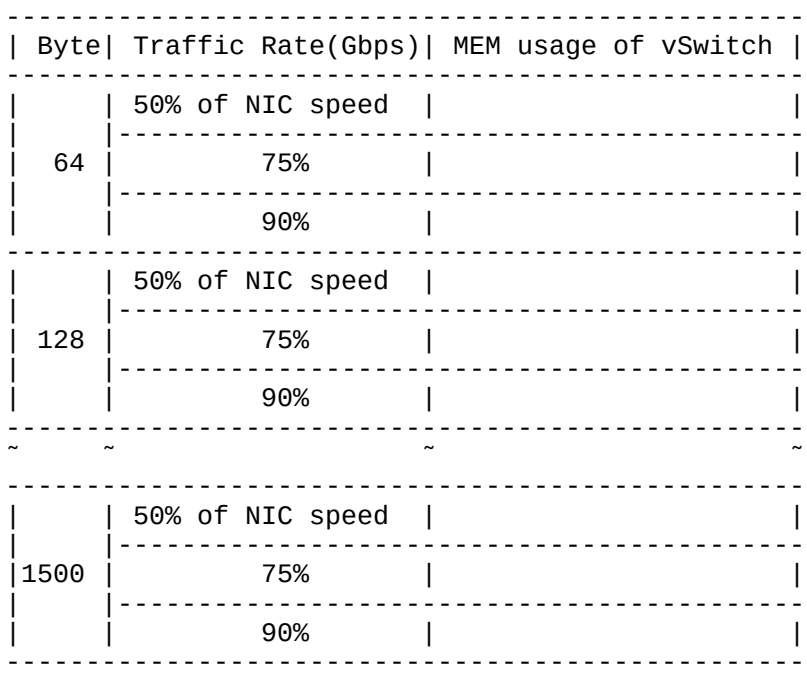


Figure 8: test result format

6.5. Latency

Physical tester’s time refers from its own clock or other time source, such as GPS, which can achieve the accuracy of 10ns. While in virtual network circumstances, the virtual tester gets its reference time from the clock of Linux systems. However, due to current methods, the clock of different servers or VMs can’t synchronize accuracy. Although VMs of some higher versions of CentOS or Fedora can achieve the accuracy of 1ms, we can get better results if the network can provide better NTP connections.

Instead of finding a better synchronization of clock to improve the accuracy of the test, we consider to use an echo server in order to forward the traffic back to the vitual switch.

We use the traffic model A as the latency test model by substituting the virtual tester with the echo server, which is used to echo the traffic.Thus the one-way delay equals to half of the round-trip time.

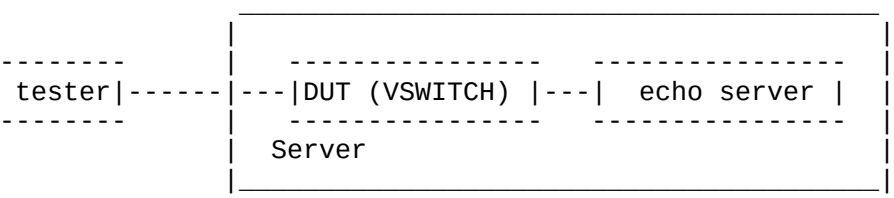


Figure 9: time delay test model

6.5.1. Objectives

The objective of this test is to verify the DUT (VSWITCH) for latency of the flow. This can be an important indicator in benchmarking the virtual network performance.

6.5.2. Configuration parameters

Network parameters should be defined as follows:

a) the number of virtual tester (VMs)

b) the number of vNIC of virtual tester

c) the CPU type of the server

d) vCPU allocated for virtual tester (VMs)

e) memory allocated for virtual tester (VMs)

f) the number and rate of server NIC

6.5.3. Test parameters

a) test repeated times

b) test frame length

6.5.4. Test process

1. Configure the physical tester to offer traffic to the V-Switch with the traffic value of throughput tested in 6.1.

2. Under the same throughput, record the time of transmitting the traffic and receiving the traffic by the physical tester with and without the DUT.

3. Calculate the time difference value between receiving and transmitting the traffic..

4. Calculate the time delay with time difference value with and without the DUT.

5. Change the frame length and repeat from step1 to step4.

6.5.5. Test result format

Byte	Latency
64	
128	
256	
512	
1024	
1518	

Figure 10: test result format

7. Security Considerations

None.

8. IANA Considerations

None.

9. Normative References

[RFC1242] Bradner, S., "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, DOI 10.17487/RFC1242, July 1991, <http://www.rfc-editor.org/info/rfc1242>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, DOI 10.17487/RFC2234, November 1997, <http://www.rfc-editor.org/info/rfc2234>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <http://www.rfc-editor.org/info/rfc2544>.

Authors' Addresses

Lu Huang (editor)
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing 100053
China

Email: hlisname@yahoo.com

Rong Gu (editor)
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing 100053
China

Email: gurong@chinamobile.com

Bob Mandeville
Iometrix
3600 Fillmore Street Suite 409
San Francisco, CA 94123
USA

Email: bob@iometrix.com

Brooks Hickman
Spirent Communications
1325 Borregas Ave
Sunnyvale, CA 94089
USA

Email: Brooks.Hickman@spirent.com

Internet-Draft
Network Working Group
Intended Status: Informational
Expires: April 01, 2018

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Mark Tassinari
Hewlett-Packard
Vishwas Manral
Nano Sec
Sarah Banks
VSS Monitoring
October 01, 2017

Benchmarking Methodology for SDN Controller Performance
draft-ietf-bmwg-sdn-controller-benchmark-meth-05

Abstract

This document defines the methodologies for benchmarking control plane performance of SDN controllers. Terminology related to benchmarking SDN controllers is described in the companion terminology document. SDN controllers have been implemented with many varying designs in order to achieve their intended network functionality. Hence, the authors have taken the approach of considering an SDN controller as a black box, defining the methodology in a manner that is agnostic to protocols and network services supported by controllers. The intent of this document is to provide a standard mechanism to measure the performance of all controller implementations.

Table of Contents

1. Introduction

This document provides generic methodologies for benchmarking SDN controller performance. An SDN controller may support many northbound and southbound protocols, implement a wide range of applications, and work solely, or as a group to achieve the desired functionality. This document considers an SDN controller as a black box, regardless of design and implementation. The tests defined in the document can be used to benchmark SDN controller for performance, scalability, reliability and security independent of northbound and southbound protocols. These tests can be performed on an SDN controller running as a virtual machine (VM) instance or on a bare metal server. This document is intended for those who want to measure the SDN controller performance as well as compare various SDN controllers performance.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. Scope

This document defines methodology to measure the networking metrics of SDN controllers. For the purpose of this memo, the SDN controller is a function that manages and controls Network Devices. Any SDN controller without a control capability is out of scope for this memo. The tests defined in this document enable benchmarking of SDN Controllers in two ways; as a standalone controller and as a cluster of homogeneous controllers. These tests are recommended for execution in lab environments rather than in live network deployments. Performance benchmarking of a federation of controllers is beyond the scope of this document. Test Setup

The tests defined in this document enable measurement of an SDN controllers performance in standalone mode and cluster mode. This section defines common reference topologies that are later referred to in individual tests (Additional forwarding Plane topologies are provided in Appendix A).

3. Test Setup

3.1. Test setup - Controller working in Standalone Mode

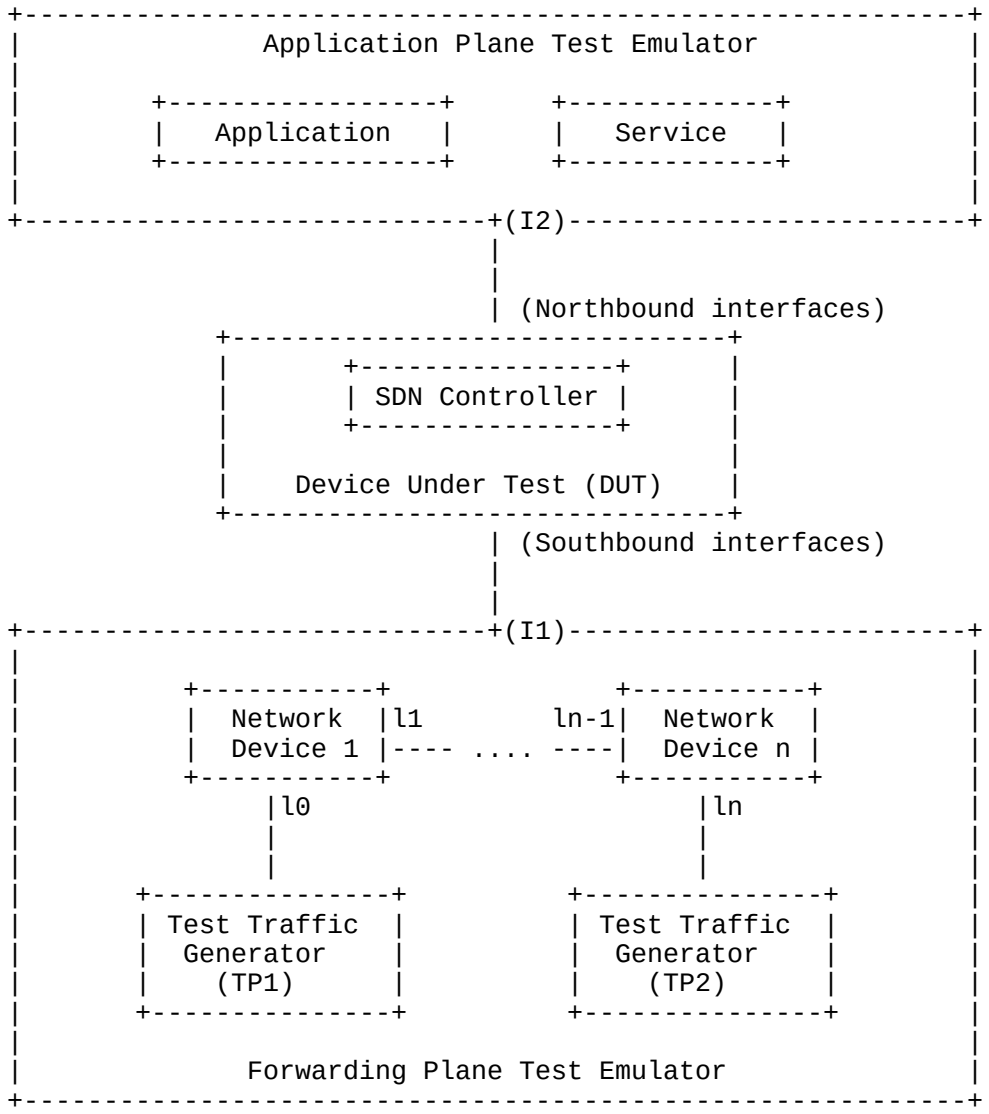


Figure 1

3.2. Test setup - Controller working in Cluster Mode

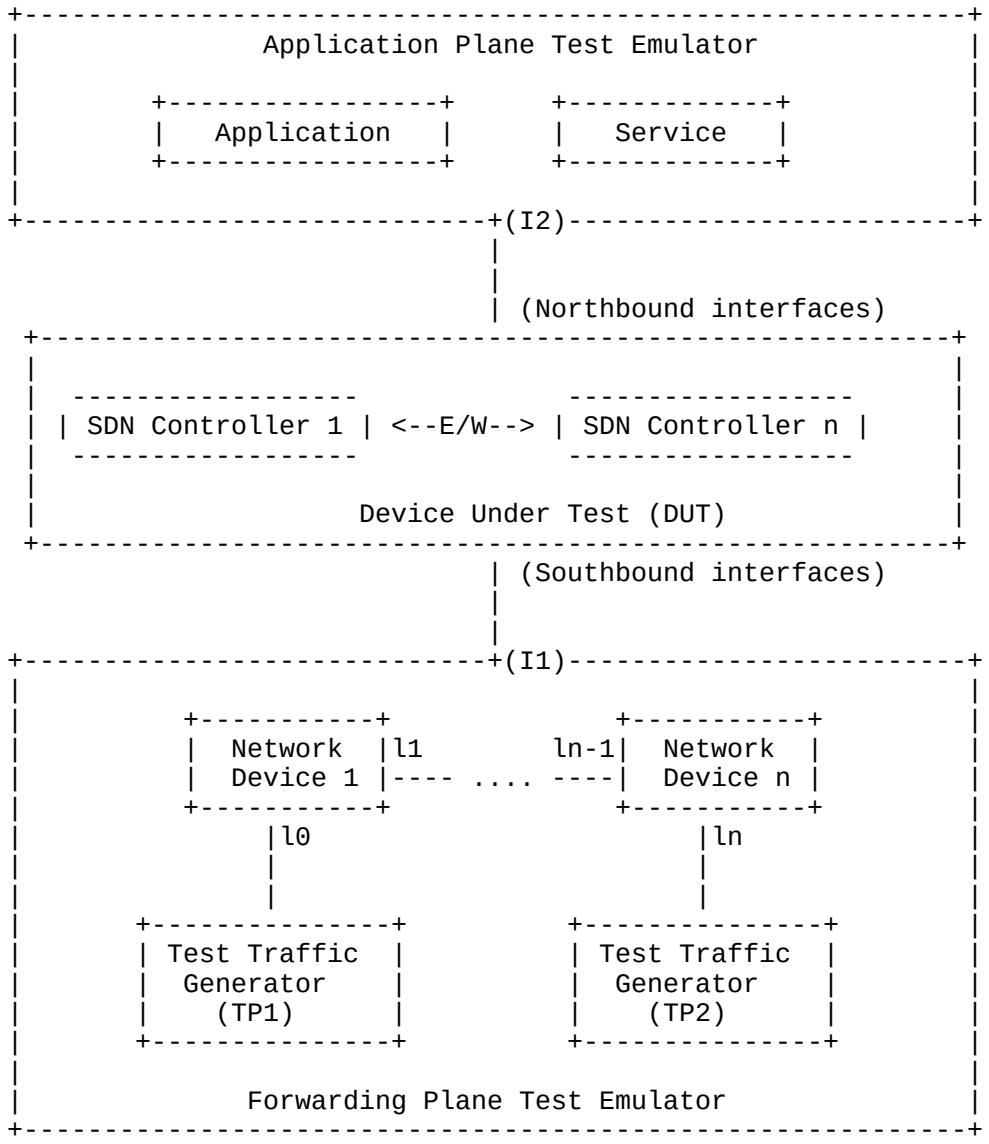


Figure 2

4. Test Considerations

4.1. Network Topology

The test cases SHOULD use Leaf-Spine topology with at least 1 Network Device in the topology for benchmarking. The test traffic generators TP1 and TP2 SHOULD be connected to the first and the last leaf Network Device. If a test case uses test topology with 1 Network Device, the test traffic generators TP1 and TP2 SHOULD be connected to the same node. However to achieve a complete performance characterization of the SDN controller, it is recommended that the controller be benchmarked for many network topologies and a varying number of Network Devices. This document includes two sample test topologies, defined in Section 10 - Appendix A for reference. Further, care should be taken to make sure that a loop prevention mechanism is enabled either in the SDN controller, or in the network when the topology contains redundant network paths.

4.2. Test Traffic

Test traffic is used to notify the controller about the asynchronous arrival of new flows. The test cases SHOULD use frame sizes of 128, 512 and 1508 bytes for benchmarking. Testing using jumbo frames are optional.

4.3. Test Emulator Requirements

The Test Emulator SHOULD time stamp the transmitted and received control messages to/from the controller on the established network connections. The test cases use these values to compute the controller processing time.

4.4. Connection Setup

There may be controller implementations that support unencrypted and encrypted network connections with Network Devices. Further, the controller may have backward compatibility with Network Devices running older versions of southbound protocols. It may be useful to measure the controller performance be measured with one or more applicable connection setup methods defined below.

1. Unencrypted connection with Network Devices, running same protocol version.
2. Unencrypted connection with Network Devices, running different protocol versions.
   Example:

   a. Controller running current protocol version and switch running older protocol version
   b. Controller running older protocol version and switch running current protocol version
3. Encrypted connection with Network Devices, running same protocol version
4. Encrypted connection with Network Devices, running different protocol versions.
   Example:
   a. Controller running current protocol version and switch running older protocol version
   b. Controller running older protocol version and switch running current protocol version

4.5. Measurement Point Specification and Recommendation

The measurement accuracy depends on several factors including the point of observation where the indications are captured. For example, the notification can be observed at the controller or test emulator. The test operator SHOULD make the observations/ measurements at the interfaces of test emulator unless it is explicitly mentioned otherwise in the individual test. In any case, the locations of measurement points MUST be reported.

4.6. Connectivity Recommendation

The SDN controller in the test setup SHOULD be connected directly with the forwarding and the management plane test emulators to avoid any delays or failure introduced by the intermediate devices during benchmarking tests. When the controller is implemented as a virtual machine, details of the physical and logical connectivity MUST be reported.

4.7. Test Repeatability

To increase the confidence in measured result, it is recommended that each test SHOULD be repeated a minimum of 10 times.

Test Reporting

Each test has a reporting format that contains some global and identical reporting components, and some individual components that are specific to individual tests. The following test configuration parameters and controller settings parameters MUST be reflected in the test report.

Test Configuration Parameters:

1. Controller name and version
2. Northbound protocols and versions
3. Southbound protocols and versions
4. Controller redundancy mode (Standalone or Cluster Mode)
5. Connection setup (Unencrypted or Encrypted)
6. Network Topology (Mesh or Tree or Linear)
7. Network Device Type (Physical or Virtual or Emulated)
8. Number of Nodes
9. Number of Links
10. Dataplane Test Traffic Type
11. Controller System Configuration (e.g., Physical or Virtual Machine, CPU, Memory, Caches, Operating System, Interface Speed, Storage)
12. Reference Test Setup (e.g., Section 3.1 etc.,)

Controller Settings Parameters:

1. Topology re-discovery timeout
2. Controller redundancy mode (e.g., active-standby etc.,)
3. Controller state persistence enabled/disabled

To ensure the repeatability of test, the following capabilities of test emulator SHOULD be reported

1. Maximum number of Network Devices that the forwarding plane emulates
2. Control message processing time (e.g., Topology Discovery Messages)

One way to determine the above two values are to simulate the required control sessions and messages from the control plane.

5. Benchmarking Tests

5.1. Performance

5.1.1. Network Topology Discovery Time

Objective:

The time taken by controller(s) to determine the complete network topology, defined as the interval starting with the first discovery message from the controller(s) at its Southbound interface, ending with all features of the static topology determined.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST support network discovery.
2. Tester should be able to retrieve the discovered topology information either through the controller's management interface, or northbound interface to determine if the discovery was successful and complete.
3. Ensure that the controller's topology re-discovery timeout has been set to the maximum value to avoid initiation of re-discovery process in the middle of the test.

Procedure:

1. Ensure that the controller is operational, its network applications, northbound and southbound interfaces are up and running.
2. Establish the network connections between controller and Network Devices.
3. Record the time for the first discovery message (Tm1) received from the controller at forwarding plane test emulator interface I1.
4. Query the controller every 3 seconds to obtain the discovered network topology information through the northbound interface or the management interface and compare it with the deployed network topology information.
5. Stop the trial when the discovered topology information matches the deployed network topology, or when the discovered topology information for 3 consecutive queries return the same details.
6. Record the time last discovery message (Tmn) sent to controller from the forwarding plane test emulator interface (I1) when the trail completed successfully. (e.g., the topology matches).

Measurement:

Topology Discovery Time Tr1 = Tmn-Tm1.

$$\text{Average Topology Discovery Time} = \frac{Tr1 + Tr2 + Tr3 \,..\, Trn}{\text{Total Trails}}$$

Reporting Format:

The Topology Discovery Time results MUST be reported in the format of a table, with a row for each successful iteration. The last row of the table indicates the average Topology Discovery Time.

If this test is repeated with varying number of nodes over the same topology, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the Number of nodes (N), the Y coordinate SHOULD be the average Topology Discovery Time.

If this test is repeated with same number of nodes over different topologies, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the Topology Type, the Y coordinate SHOULD be the average Topology Discovery Time.

5.1.2. Asynchronous Message Processing Time

Objective:

The time taken by controller(s) to process an asynchronous message, defined as the interval starting with an asynchronous message from a network device after the discovery of all the devices by the controller(s), ending with a response message from the controller(s) at its Southbound interface.

Reference Test Setup:

This test SHOULD use one of the test setup described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST have successfully completed the network topology discovery for the connected Network Devices.

Procedure:

1. Generate asynchronous messages from every connected Network Device, to the SDN controller, one at a time in series from the forwarding plane test emulator for the trail duration.
2. Record every request transmit (T1) timestamp and the corresponding response (R1) received timestamp at the forwarding plane test emulator interface (I1) for every successful message exchange.

Measurement:

$$\text{Asynchronous Message Processing Time } Tr1 = \frac{(R1-T1) + (R2-T2)..(Rn-Tn)}{Nrx}$$

Where Nrx is the total number of successful messages exchanged

$$\text{Average Asynchronous Message Processing Time} = \frac{Tr1 + Tr2 + Tr3..Trn}{\text{Total Trails}}$$

Reporting Format:

The Asynchronous Message Processing Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the average Asynchronous Message Processing Time.

The report should capture the following information in addition to the configuration parameters captured in section 5. - Successful messages exchanged (Nrx)

If this test is repeated with varying number of nodes with same topology, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the Number of nodes (N), the Y coordinate SHOULD be the average Asynchronous Message Processing Time.

If this test is repeated with same number of nodes using different topologies, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the Topology Type, the Y coordinate SHOULD be the average Asynchronous Message Processing Time.

5.1.3. Asynchronous Message Processing Rate

Objective:

Measure the number of responses to asynchronous messages (such as new flow arrival notification message, etc.) for which the controller(s) performed processing and replied with a valid and productive (non-trivial) response message

This test will measure two benchmarks on Asynchronous Message Processing Rate using a single procedure. The two benchmarks are (see section 2.3.1.3 of [I-D.sdn-controller-benchmark-term]):

1. Loss-free Asynchronous Message Processing Rate

2. Maximum Asynchronous Message Processing Rate

Here two benchmarks are determined through a series of trials where the number of messages are sent to the controller(s), and the responses from the controller(s) are counted over the trial duration. The message response rate and the message loss ratio are calculated for each trial.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller(s) MUST have successfully completed the network topology discovery for the connected Network Devices.
2. Choose and record the Trial Duration (Td), the sending rate step-size (STEP), the tolerance on equality for two consecutive trials (P%),and the maximum possible message sending rate (Ntx1/Td).

Procedure:

1. Generate asynchronous messages continuously at the maximum possible rate on the established connections from all the emulated/simulated Network Devices for the given trial Duration (Td).
2. Record the total number of responses received from the controller (Nrx1) as well as the number of messages sent (Ntx1) to the controller within the trial duration(Td).
3. Calculate the Asynchronous Message Processing Rate (Tr1) and the Message Loss Ratio (Lr1). Ensure that the controller(s) have returned to normal operation.
4. Repeat the trial by reducing the asynchronous message sending rate used in last trial by the STEP size.
5. Continue repeating the trials and reducing the sending rate until both the maximum value of Nrxn and the Nrxn corresponding to zero loss ratio have been found.
6. The trials corresponding to the benchmark levels MUST be repeated using the same asynchronous message rates until the responses received from the controller are equal (+/-P%) for two consecutive trials.
7. Record the number of responses received from the controller (Nrxn) as well as the number of messages sent (Ntxn) to the controller in the last trial.

Measurement:

$$\text{Asynchronous Message Processing Rate } Trn = \frac{Nrxn}{Td}$$

Maximum Asynchronous Message Processing Rate = MAX(Trn) for all n

$$\text{Asynchronous Message Loss Ratio } Lrn = 1 - \frac{Nrxn}{Ntxn}$$

Loss-free Asynchronous Message Processing Rate = MAX(Trn) given Lrn=0

Reporting Format:

The Asynchronous Message Processing Rate results MUST be reported in the format of a table with a row for each trial.

The table should report the following information in addition to the configuration parameters captured in section 5, with columns:

- Offered rate (Ntxn/Td)

- Asynchronous Message Processing Rate (Trn)

- Loss Ratio (Lr)

- Benchmark at this iteration (blank for none, Maximum, Loss-Free)

The results MAY be presented in the form of a graph. The X axis SHOULD be the Offered rate, and dual Y axes would represent Asynchronous Message Processing Rate and Loss Ratio, respectively.

If this test is repeated with varying number of nodes over same topology, the results SHOULD be reported in the form of a graph. The X axis SHOULD be the Number of nodes (N), the Y axis SHOULD be the Asynchronous Message Processing Rate. Both the Maximum and the Loss-Free Rates should be plotted for each N.

If this test is repeated with same number of nodes over different topologies, the results SHOULD be reported in the form of a graph. The X axis SHOULD be the Topology Type, the Y axis SHOULD be the Asynchronous Message Processing Rate. Both the Maximum and the Loss-Free Rates should be plotted for each topology.

5.1.4. Reactive Path Provisioning Time

Objective:

The time taken by the controller to setup a path reactively between source and destination node, defined as the interval starting with the first flow provisioning request message received by the controller(s) at its Southbound interface, ending with the last flow provisioning response message sent from the controller(s) at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A. The number of Network Devices in the path is a parameter of the test that may be varied from 2 to maximum discovery size in repetitions of this test.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should have the knowledge about the location of destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for ’flow miss’ in Network Device is configured to ’send to controller’.
4. Ensure that each Network Device in a path requires the controller to make the forwarding decision while paving the entire path.

Procedure:

1. Send a single traffic stream from the test traffic generator TP1 to test traffic generator TP2.
2. Record the time of the first flow provisioning request message sent to the controller (Tsf1) from the Network Device at the forwarding plane test emulator interface (I1).
3. Wait for the arrival of first traffic frame at the Traffic Endpoint TP2 or the expiry of trail duration (Td).
4. Record the time of the last flow provisioning response message received from the controller (Tdf1) to the Network Device at the forwarding plane test emulator interface (I1).

Measurement:

Reactive Path Provisioning Time Tr1 = Tdf1-Tsf1.

$$\text{Average Reactive Path Provisioning Time} = \frac{Tr1 + Tr2 + Tr3 \ldots Trn}{\text{Total Trails}}$$

Reporting Format:

The Reactive Path Provisioning Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Average Reactive Path Provisioning Time

The report should capture the following information in addition to the configuration parameters captured in section 5.

- Number of Network Devices in the path

5.1.5. Proactive Path Provisioning Time

Objective:

The time taken by the controller to setup a path proactively between source and destination node, defined as the interval starting with the first proactive flow provisioned in the controller(s) at its Northbound interface, ending with the last flow provisioning response message sent from the controller(s) at it Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should have the knowledge about the location of destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.

3. Ensure that the default action for flow miss in Network Device is 'drop'.

Procedure:

1. Send a single traffic stream from test traffic generator TP1 to TP2.
2. Install the flow entries to reach from test traffic generator TP1 to the test traffic generator TP2 through controller's northbound or management interface.
3. Wait for the arrival of first traffic frame at the test traffic generator TP2 or the expiry of trail duration (Td).
4. Record the time when the proactive flow is provisioned in the Controller (Tsf1) at the management plane test emulator interface I2.
5. Record the time of the last flow provisioning message received from the controller (Tdf1) at the forwarding plane test emulator interface I1.

Measurement:

Proactive Flow Provisioning Time Tr1 = Tdf1-Tsf1.

$$\text{Average Proactive Path Provisioning Time} = \frac{Tr1 + Tr2 + Tr3 \ldots Trn}{\text{Total Trails}}$$

Reporting Format:

The Proactive Path Provisioning Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Average Proactive Path Provisioning Time.

The report should capture the following information in addition to the configuration parameters captured in section 5.

- Number of Network Devices in the path

5.1.6. Reactive Path Provisioning Rate

Objective:

The maximum number of independent paths a controller can concurrently establish between source and destination nodes reactively, defined as the number of paths provisioned by the controller(s) at its Southbound interface for the flow provisioning

requests received for path provisioning at its Southbound interface between the start of the test and the expiry of given trail duration.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should have the knowledge about the location of destination addresses for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for ’flow miss’ in Network Device is configured to ’send to controller’.
4. Ensure that each Network Device in a path requires the controller to make the forwarding decision while provisioning the entire path.

Procedure:

1. Send traffic with unique source and destination addresses from test traffic generator TP1.
2. Record total number of unique traffic frames (Ndf) received at the test traffic generator TP2 within the trail duration (Td).

Measurement:

$$\text{Reactive Path Provisioning Rate Tr1} = \frac{\text{Ndf}}{\text{Td}}$$

$$\text{Average Reactive Path Provisioning Rate} = \frac{\text{Tr1} + \text{Tr2} + \text{Tr3} .. \text{Trn}}{\text{Total Trails}}$$

Reporting Format:

The Reactive Path Provisioning Rate results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Average Reactive Path Provisioning Rate.

The report should capture the following information in addition to the configuration parameters captured in section 5.

- Number of Network Devices in the path

- Offered rate

5.1.7. Proactive Path Provisioning Rate

Objective:

Measure the maximum rate of independent paths a controller can concurrently establish between source and destination nodes proactively, defined as the number of paths provisioned by the controller(s) at its Southbound interface for the paths requested in its Northbound interface between the start of the test and the expiry of given trail duration . The measurement is based on dataplane observations of successful path activation

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.

2. The controller should have the knowledge about the location of destination addresses for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.

3. Ensure that the default action for flow miss in Network Device is ’drop’.

Procedure:

1. Send traffic continuously with unique source and destination addresses from test traffic generator TP1.

2. Install corresponding flow entries to reach from simulated sources at the test traffic generator TP1 to the simulated

destinations at test traffic generator TP2 through controller's northbound or management interface.

3. Record total number of unique traffic frames received Ndf) at the test traffic generator TP2 within the trail duration (Td).

Measurement:

$$\text{Proactive Path Provisioning Rate Tr1} = \frac{\text{Ndf}}{\text{Td}}$$

$$\text{Average Proactive Path Provisioning Rate} = \frac{\text{Tr1 + Tr2 + Tr3 .. Trn}}{\text{Total Trails}}$$

Reporting Format:

The Proactive Path Provisioning Rate results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Average Proactive Path Provisioning Rate.

The report should capture the following information in addition to the configuration parameters captured in section 5.

- Number of Network Devices in the path

- Offered rate

5.1.8. Network Topology Change Detection Time

Objective:

The amount of time required for the controller to detect any changes in the network topology, defined as the interval starting with the notification message received by the controller(s) at its Southbound interface, ending with the first topology rediscovery messages sent from the controller(s) at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST have successfully discovered the network topology information for the deployed network topology.

2. The periodic network discovery operation should be configured to twice the Trail duration (Td) value.

Procedure:

1. Trigger a topology change event by bringing down an active Network Device in the topology.

2. Record the time when the first topology change notification is sent to the controller (Tcn) at the forwarding plane test emulator interface (I1).

3. Stop the trail when the controller sends the first topology re-discovery message to the Network Device or the expiry of trail duration (Td).

4. Record the time when the first topology re-discovery message is received from the controller (Tcd) at the forwarding plane test emulator interface (I1)

Measurement:

Network Topology Change Detection Time Tr1 = Tcd-Tcn.

$$\text{Average Network Topology Change Detection Time} = \frac{Tr1 + Tr2 + Tr3 .. Trn}{\text{Total Trails}}$$

Reporting Format:

The Network Topology Change Detection Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the average Network Topology Change Time.

5.2. Scalability

5.2.1. Control Session Capacity

Objective:

Measure the maximum number of control sessions the controller can maintain, defined as the number of sessions that the controller can accept from network devices, starting with the first control session, ending with the last control session that the controller(s) accepts at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Procedure:

1. Establish control connection with controller from every Network Device emulated in the forwarding plane test emulator.
2. Stop the trail when the controller starts dropping the control connections.
3. Record the number of successful connections established with the controller (CCn) at the forwarding plane test emulator.

Measurement:

Control Sessions Capacity = CCn.

Reporting Format:

The Control Session Capacity results MUST be reported in addition to the configuration parameters captured in section 5.

5.2.2. Network Discovery Size

Objective:

Measure the network size (number of nodes, links and hosts) that a controller can discover, defined as the size of a network that the controller(s) can discover, starting from a network topology given by the user for discovery, ending with the topology that the controller(s) could successfully discover.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller MUST support automatic network discovery.
2. Tester should be able to retrieve the discovered topology information either through controller's management interface or northbound interface.

Procedure:

1. Establish the network connections between controller and network nodes.
2. Query the controller for the discovered network topology information and compare it with the deployed network topology information.
3. Increase the number of nodes by 1 when the comparison is successful and repeat the trail.
4. Decrease the number of nodes by 1 when the comparison fails and repeat the trail.
5. Continue the trail until the comparison of step 4 is successful.
6. Record the number of nodes for the last trail (Ns) where the topology comparison was successful.

Measurement:

Network Discovery Size = Ns.

Reporting Format:

The Network Discovery Size results MUST be reported in addition to the configuration parameters captured in section 5.

5.2.3. Forwarding Table Capacity

Objective:

Measure the maximum number of flow entries a controller can manage in its Forwarding table.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. The controller Forwarding table should be empty.
2. Flow Idle time MUST be set to higher or infinite value.
3. The controller MUST have successfully completed network topology discovery.
4. Tester should be able to retrieve the forwarding table information either through controller's management interface or northbound interface.

Procedure:

Reactive Flow Provisioning Mode:

1. Send bi-directional traffic continuously with unique source and/or destination addresses from test traffic generators TP1 and TP2 at the asynchronous message processing rate of controller.
2. Query the controller at a regular interval (e.g., 5 seconds) for the number of learnt flow entries from its northbound interface.
3. Stop the trail when the retrieved value is constant for three consecutive iterations and record the value received from the last query (Nrp).

Proactive Flow Provisioning Mode:

1. Install unique flows continuously through controller's northbound or management interface until a failure response is received from the controller.
2. Record the total number of successful responses (Nrp).

Note:

Some controller designs for proactive flow provisioning mode may require the switch to send flow setup requests in order to generate flow setup responses. In such cases, it is recommended to generate bi-directional traffic for the provisioned flows.

Measurement:

Proactive Flow Provisioning Mode:

Max Flow Entries = Total number of flows provisioned (Nrp)

Reactive Flow Provisioning Mode:

Max Flow Entries = Total number of learnt flow entries (Nrp)

Forwarding Table Capacity = Max Flow Entries.

Reporting Format:

The Forwarding Table Capacity results MUST be tabulated with the following information in addition to the configuration parameters captured in section 5.

- Provisioning Type (Proactive/Reactive)

5.3. Security

5.3.1. Exception Handling

Objective:

Determine the effect of handling error packets and notifications on performance tests. The impact MUST be measured for the following performance tests

a. Path Provisioning Rate

b. Path Provisioning Time

c. Network Topology Change Detection Time

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. This test MUST be performed after obtaining the baseline measurement results for the above performance tests.

2. Ensure that the invalid messages are not dropped by the intermediate devices connecting the controller and Network Devices.

Procedure:

1. Perform the above listed performance tests and send 1% of messages from the Asynchronous Message Processing Rate as invalid messages from the connected Network Devices emulated at the forwarding plane test emulator.
2. Perform the above listed performance tests and send 2% of messages from the Asynchronous Message Processing Rate as invalid messages from the connected Network Devices emulated at the forwarding plane test emulator.

Note:

Invalid messages can be frames with incorrect protocol fields or any form of failure notifications sent towards controller.

Measurement:

Measurement MUST be done as per the equation defined in the corresponding performance test measurement section.

Reporting Format:

The Exception Handling results MUST be reported in the format of table with a column for each of the below parameters and row for each of the listed performance tests.

- Without Exceptions

- With 1% Exceptions

- With 2% Exceptions

5.3.2. Denial of Service Handling

Objective:

Determine the effect of handling DoS attacks on performance and scalability tests the impact MUST be measured for the following tests:

a. Path Provisioning Rate

b. Path Provisioning Time

c. Network Topology Change Detection Time

d. Network Discovery Size

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

This test MUST be performed after obtaining the baseline measurement results for the above tests.

Procedure:

1. Perform the listed tests and launch a DoS attack towards controller while the trail is running.

Note:

DoS attacks can be launched on one of the following interfaces.

a. Northbound (e.g., Sending a huge number of requests on northbound interface)
b. Management (e.g., Ping requests to controller’s management interface)
c. Southbound (e.g., TCP SYNC messages on southbound interface)

Measurement:

Measurement MUST be done as per the equation defined in the corresponding test’s measurement section.

Reporting Format:

The DoS Attacks Handling results MUST be reported in the format of table with a column for each of the below parameters and row for each of the listed tests.

- Without any attacks

- With attacks

The report should also specify the nature of attack and the interface.

5.4. Reliability

5.4.1. Controller Failover Time

Objective:

The time taken to switch from an active controller to the backup controller, when the controllers work in redundancy mode and the active controller fails, defined as the interval starting with the active controller bringing down, ending with the first re-discovery message received from the new controller at its Southbound interface.

Reference Test Setup:

The test SHOULD use the test setup described in section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. Master controller election MUST be completed.
2. Nodes are connected to the controller cluster as per the Redundancy Mode (RM).
3. The controller cluster should have successfully completed the network topology discovery.
4. The Network Device MUST send all new flows to the controller when it receives from the test traffic generator.
5. Controller should have learnt the location of destination (D1) at test traffic generator TP2.

Procedure:

1. Send uni-directional traffic continuously with incremental sequence number and source addresses from test traffic generator TP1 at the rate that the controller processes without any drops.
2. Ensure that there are no packet drops observed at the test traffic generator TP2.
3. Bring down the active controller.
4. Stop the trail when a first frame received on TP2 after failover operation.

5. Record the time at which the last valid frame received (T1) at test traffic generator TP2 before sequence error and the first valid frame received (T2) after the sequence error at TP2

Measurement:

Controller Failover Time = (T2 - T1)

Packet Loss = Number of missing packet sequences.

Reporting Format:

The Controller Failover Time results MUST be tabulated with the following information.

- Number of cluster nodes
- Redundancy mode
- Controller Failover Time
- Packet Loss
- Cluster keep-alive interval

5.4.2. Network Re-Provisioning Time

Objective:

The time taken to re-route the traffic by the Controller, when there is a failure in existing traffic paths, defined as the interval starting from the first failure notification message received by the controller, ending with the last flow re-provisioning message sent by the controller at its Southbound interface.

Reference Test Setup:

This test SHOULD use one of the test setup described in section 3.1 or section 3.2 of this document in combination with Appendix A.

Prerequisite:

1. Network with the given number of nodes and redundant paths MUST be deployed.

2. Ensure that the controller MUST have knowledge about the location of test traffic generators TP1 and TP2.
3. Ensure that the controller does not pre-provision the alternate path in the emulated Network Devices at the forwarding plane test emulator.

Procedure:

1. Send bi-directional traffic continuously with unique sequence number from TP1 and TP2.
2. Bring down a link or switch in the traffic path.
3. Stop the trail after receiving first frame after network re-convergence.
4. Record the time of last received frame prior to the frame loss at TP2 (TP2-Tlfr) and the time of first frame received after the frame loss at TP2 (TP2-Tffr). There must be a gap in sequence numbers of these frames
5. Record the time of last received frame prior to the frame loss at TP1 (TP1-Tlfr) and the time of first frame received after the frame loss at TP1 (TP1-Tffr).

Measurement:

Forward Direction Path Re-Provisioning Time (FDRT)
= (TP2-Tffr - TP2-Tlfr)

Reverse Direction Path Re-Provisioning Time (RDRT)
= (TP1-Tffr - TP1-Tlfr)

Network Re-Provisioning Time = (FDRT+RDRT)/2

Forward Direction Packet Loss = Number of missing sequence frames at TP1

Reverse Direction Packet Loss = Number of missing sequence frames at TP2

Reporting Format:

The Network Re-Provisioning Time results MUST be tabulated with the following information.

- Number of nodes in the primary path

- Number of nodes in the alternate path

- Network Re-Provisioning Time

- Forward Direction Packet Loss

- Reverse Direction Packet Loss

6. References

6.1. Normative References

[I-D.sdn-controller-benchmark-term] Bhuvaneswaran.V, Anton Basil, Mark.T, Vishwas Manral, Sarah Banks, "Terminology for Benchmarking SDN Controller Performance", draft-ietf-bmwg-sdn-controller-benchmark-term-05 (Work in progress), October 01, 2017

6.2. Informative References

[OpenFlow Switch Specification] ONF,"OpenFlow Switch Specification" Version 1.4.0 (Wire Protocol 0x05), October 14, 2013.

7. IANA Considerations

This document does not have any IANA requests.

8. Security Considerations

Benchmarking tests described in this document are limited to the performance characterization of controller in lab environment with isolated network.

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the controller.

Special capabilities SHOULD NOT exist in the controller specifically for benchmarking purposes. Any implications for network security arising from the controller SHOULD be identical in the lab and in production networks

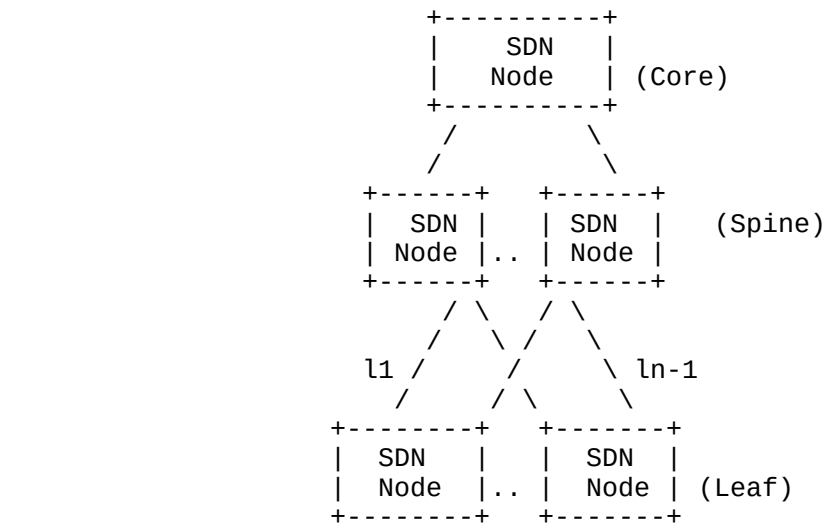
9. Acknowledgments

The authors would like to thank the following individuals for providing their valuable comments to the earlier versions of this document: Al Morton (AT&T), Sandeep Gangadharan (HP), M. Georgescu (NAIST), Andrew McGregor (Google), Scott Bradner (Harvard University), Jay Karthik (Cisco), Ramakrishnan (Dell), Khasanov Boris (Huawei), Brian Castelli (Spirent)

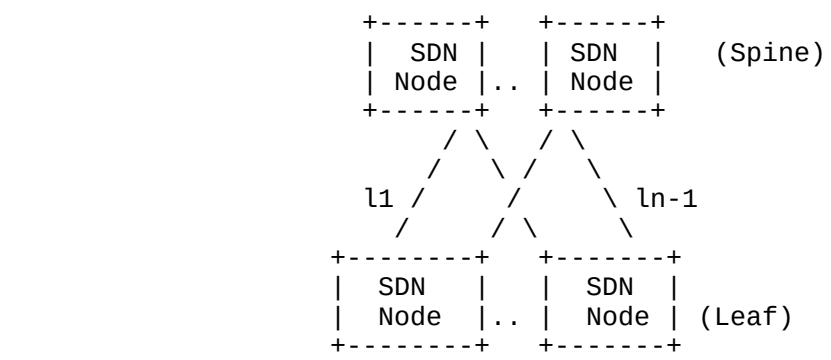
This document was prepared using 2-Word-v2.0.template.dot.

Appendix A. Example Test Topologies

A.1. Leaf-Spine Topology - Three Tier Network Architecture



A.2. Leaf-Spine Topology - Two Tier Network Architecture



Appendix B. Benchmarking Methodology using OpenFlow Controllers

This section gives an overview of OpenFlow protocol and provides test methodology to benchmark SDN controllers supporting OpenFlow southbound protocol.

B.1. Protocol Overview

OpenFlow is an open standard protocol defined by Open Networking Foundation (ONF)[OpenFlow Switch Specification], used for programming the forwarding plane of network switches or routers via a centralized controller.

B.2. Messages Overview

OpenFlow protocol supports three messages types namely controller-to-switch, asynchronous and symmetric.

Controller-to-switch messages are initiated by the controller and used to directly manage or inspect the state of the switch. These messages allow controllers to query/configure the switch (Features, Configuration messages), collect information from switch (Read-State message), send packets on specified port of switch (Packet-out message), and modify switch forwarding plane and state (Modify-State, Role-Request messages etc.).

Asynchronous messages are generated by the switch without a controller soliciting them. These messages allow switches to update controllers to denote an arrival of new flow (Packet-in), switch state change (Flow-Removed, Port-status) and error (Error).

Symmetric messages are generated in either direction without solicitation. These messages allow switches and controllers to set up connection (Hello), verify for liveness (Echo) and offer additional functionalities (Experimenter).

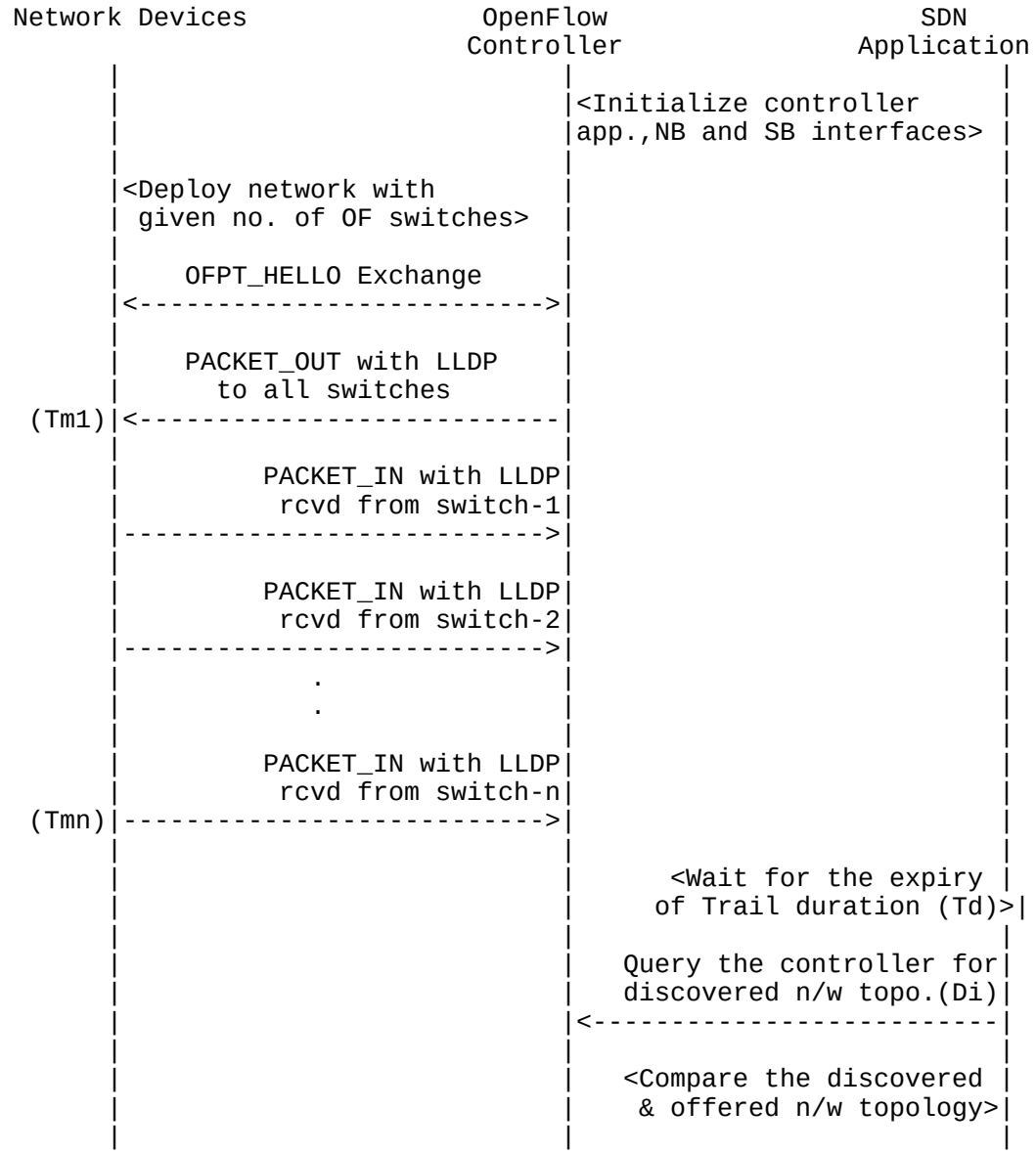
B.3. Connection Overview

OpenFlow channel is used to exchange OpenFlow message between an OpenFlow switch and an OpenFlow controller. The OpenFlow channel connection can be setup using plain TCP or TLS. By default, a switch establishes single connection with SDN controller. A switch may establish multiple parallel connections to single controller (auxiliary connection) or multiple controllers to handle controller failures and load balancing.

B.4. Performance Benchmarking Tests

B.4.1. Network Topology Discovery Time

Procedure:



Legend:

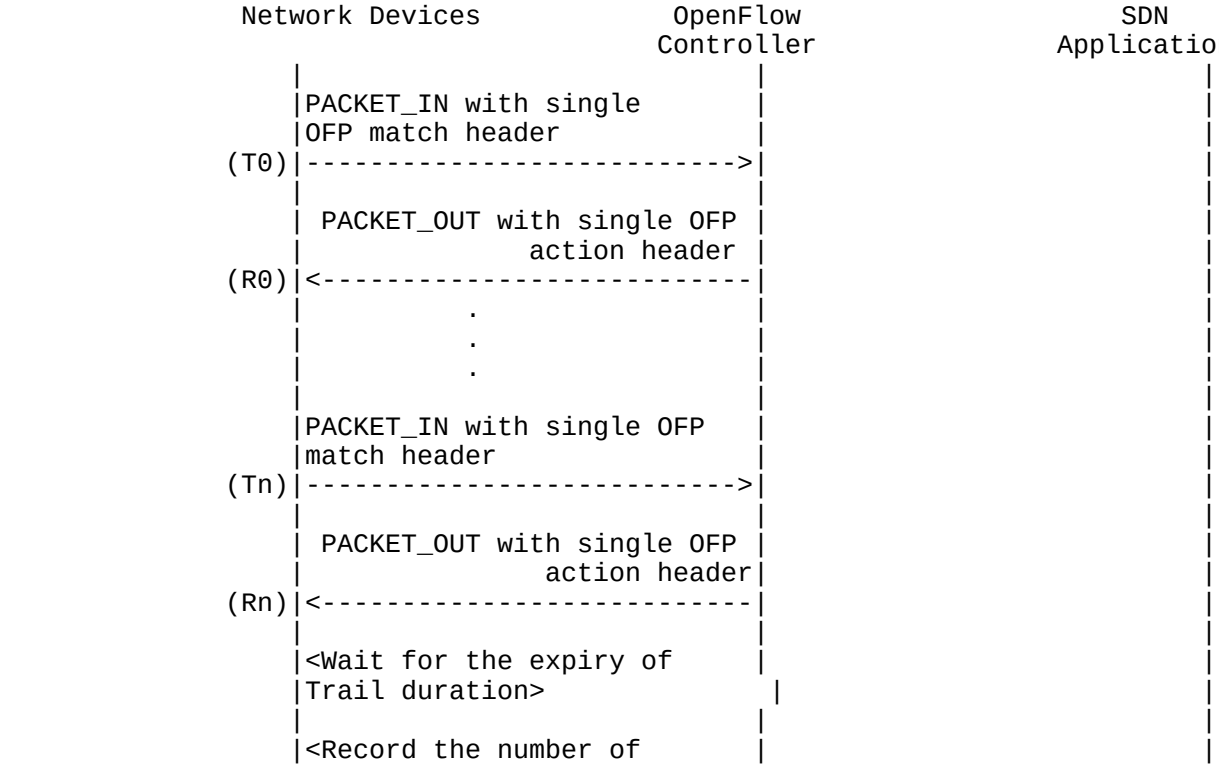
NB: Northbound
SB: Southbound
OF: OpenFlow
Tm1: Time of reception of first LLDP message from controller
Tmn: Time of last LLDP message sent to controller

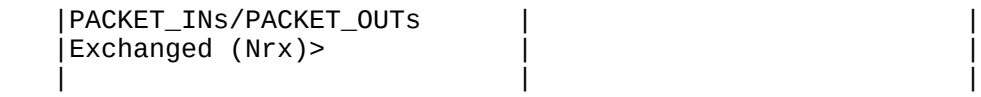
Discussion:

The Network Topology Discovery Time can be obtained by calculating the time difference between the first PACKET_OUT with LLDP message received from the controller (Tm1) and the last PACKET_IN with LLDP message sent to the controller (Tmn) when the comparison is successful.

B.4.2. Asynchronous Message Processing Time

Procedure:





Legend:

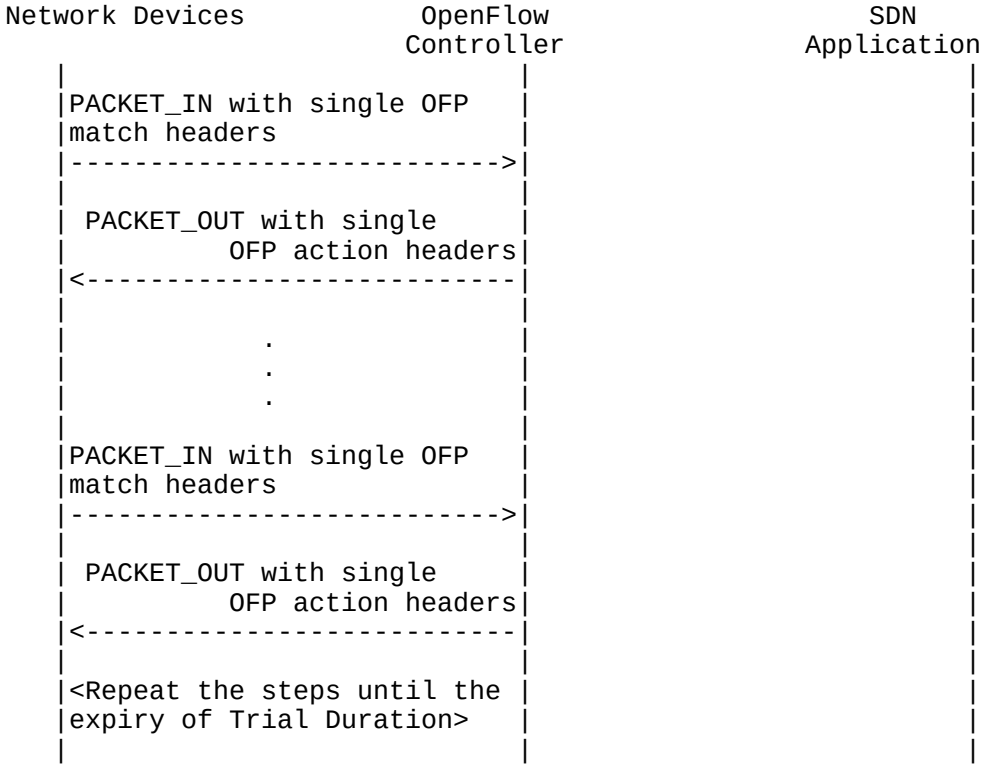
T0,T1, ..Tn are PACKET_IN messages transmit timestamps.
R0,R1, ..Rn are PACKET_OUT messages receive timestamps.
Nrx : Number of successful PACKET_IN/PACKET_OUT message exchanges

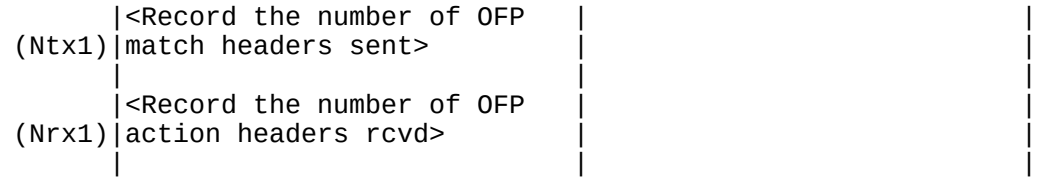
Discussion:

The Asynchronous Message Processing Time will be obtained by sum of ((R0-T0),(R1-T1)..(Rn - Tn))/ Nrx.

B.4.3. Asynchronous Message Processing Rate

Procedure:





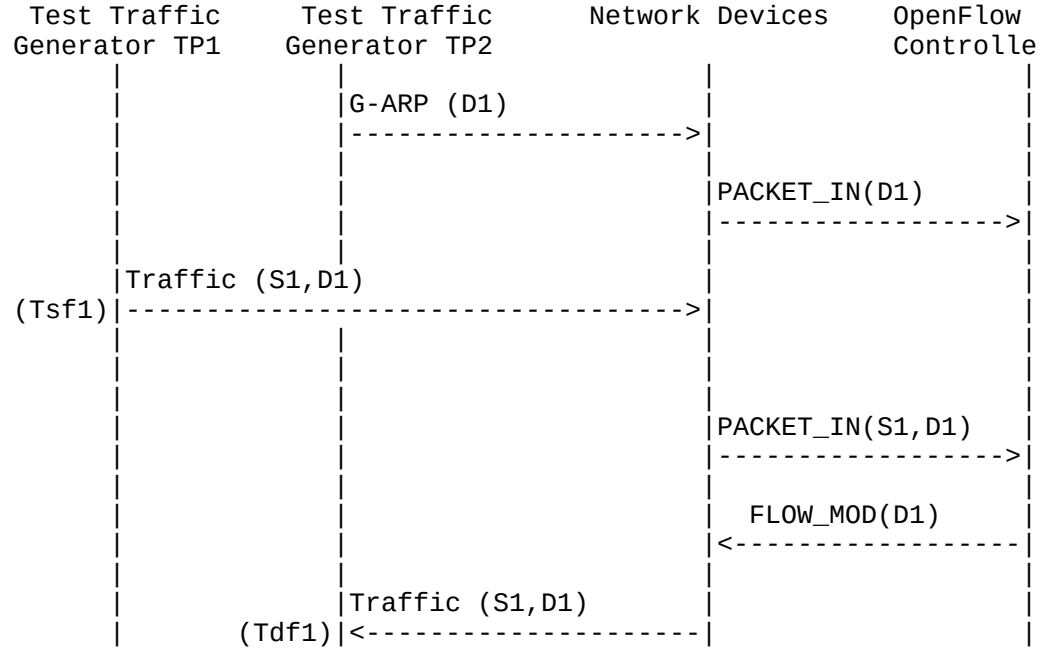
Note: The Ntx1 on initial trials should be greater than Nrx1 and repeat the trials until the Nrxn for two consecutive trials equeal to (+/-P%).

Discussion:

This test will measure two benchmarks using single procedure. 1) The Maximum Asynchronous Message Processing Rate will be obtained by calculating the maximum PACKET OUTs (Nrxn) received from the controller(s) across n trials. 2) The Loss-free Asynchronous Message Processing Rate will be obtained by calculating the maximum PACKET OUTs received from controller (s) when Loss Ratio equals zero. The loss ratio is obtained by 1 - Nrxn/Ntxn

B.4.4. Reactive Path Provisioning Time

Procedure:



Legend:

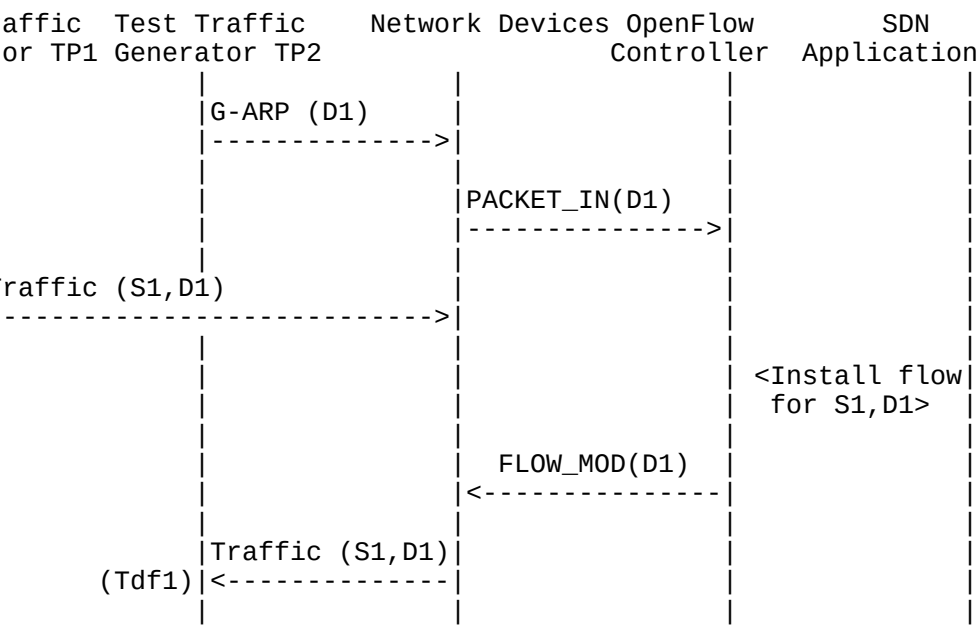
G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

Discussion:

The Reactive Path Provisioning Time can be obtained by finding the time difference between the transmit and receive time of the traffic (Tsf1-Tdf1).

B.4.5. Proactive Path Provisioning Time

Procedure:



Legend:

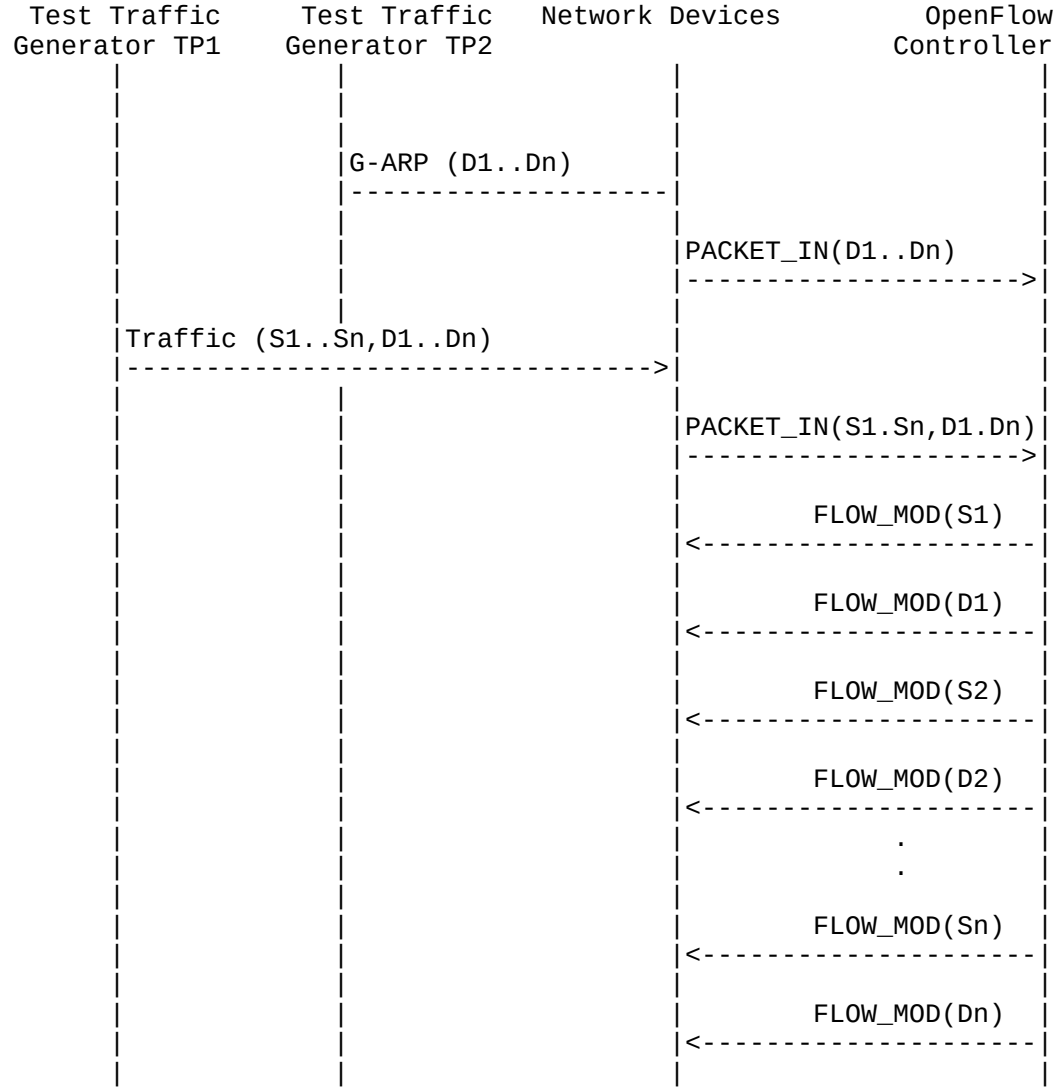
G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

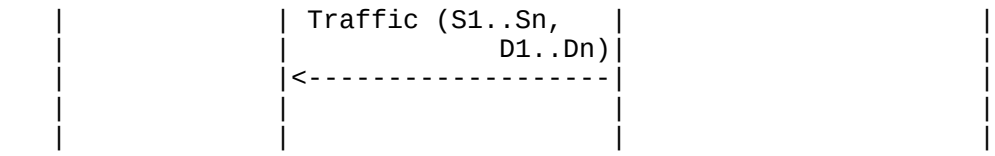
Discussion:

The Proactive Path Provisioning Time can be obtained by finding the time difference between the transmit and receive time of the traffic (Tsf1-Tdf1).

B.4.6. Reactive Path Provisioning Rate

Procedure:





Legend:

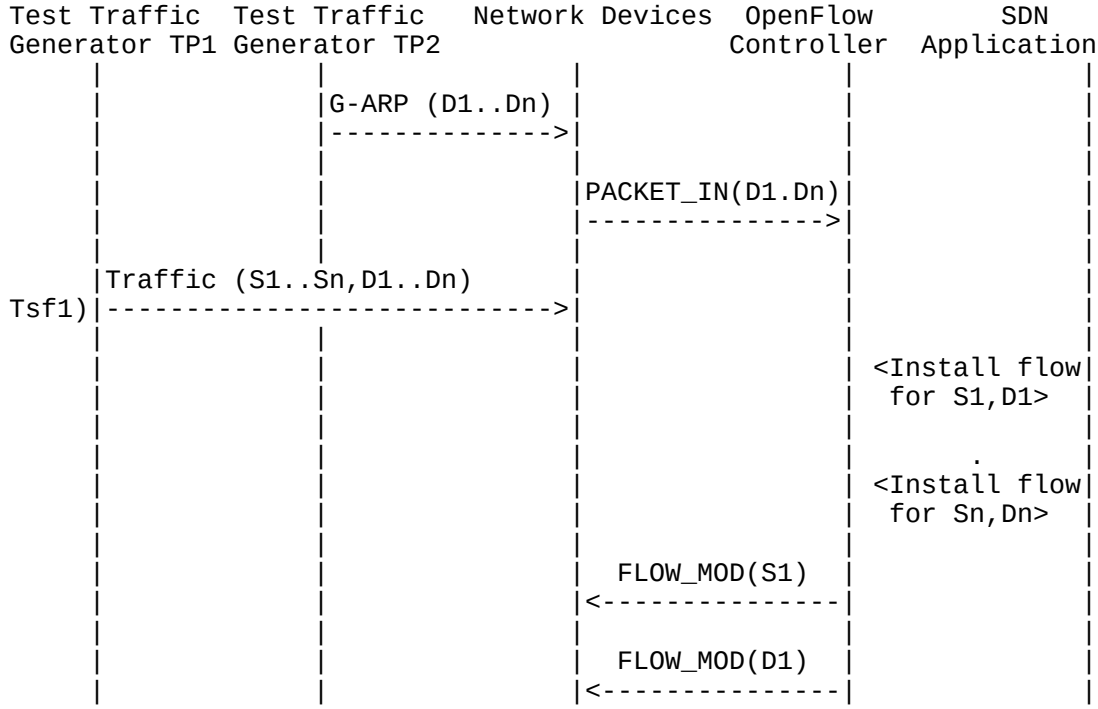
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ....
Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

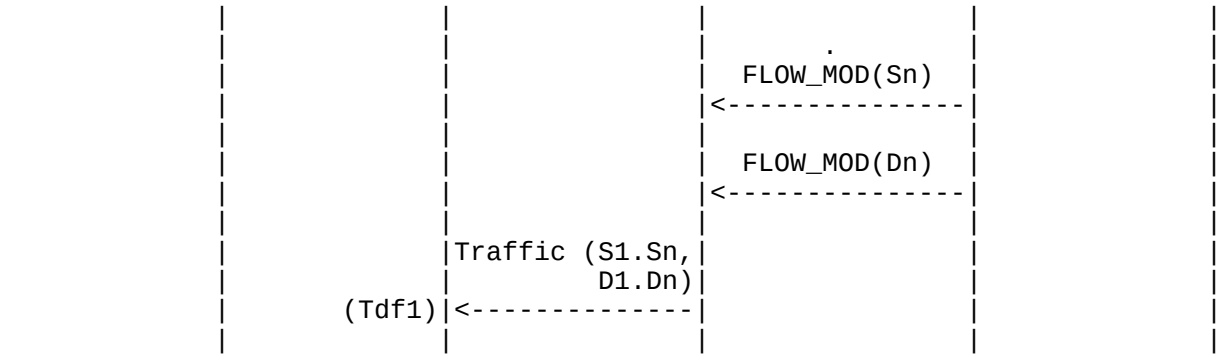
Discussion:

The Reactive Path Provisioning Rate can be obtained by finding the total number of frames received at TP2 after the trail duration.

B.4.7. Proactive Path Provisioning Rate

Procedure:





Legend:

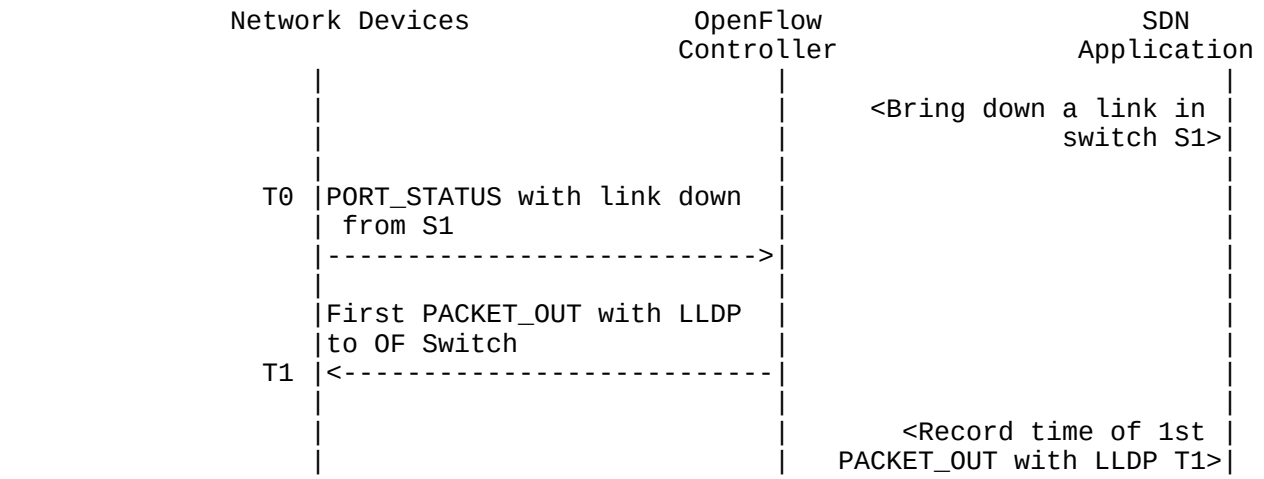
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ....
Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

Discussion:

The Proactive Path Provisioning Rate can be obtained by finding the total number of frames received at TP2 after the trail duration

B.4.8. Network Topology Change Detection Time

Procedure:



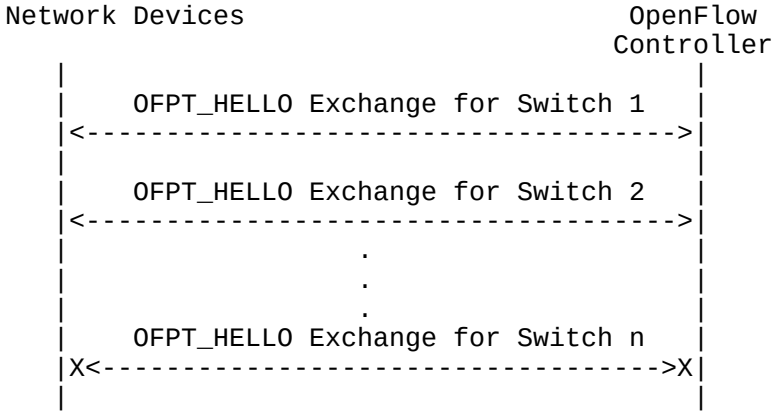
Discussion:

The Network Topology Change Detection Time can be obtained by finding the difference between the time the OpenFlow switch S1 sends the PORT_STATUS message (T0) and the time that the OpenFlow controller sends the first topology re-discovery message (T1) to OpenFlow switches.

B.5. Scalability

B.5.1. Control Sessions Capacity

Procedure:

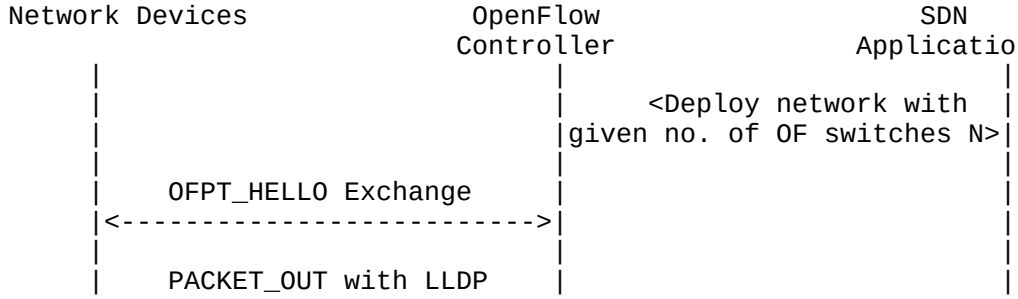


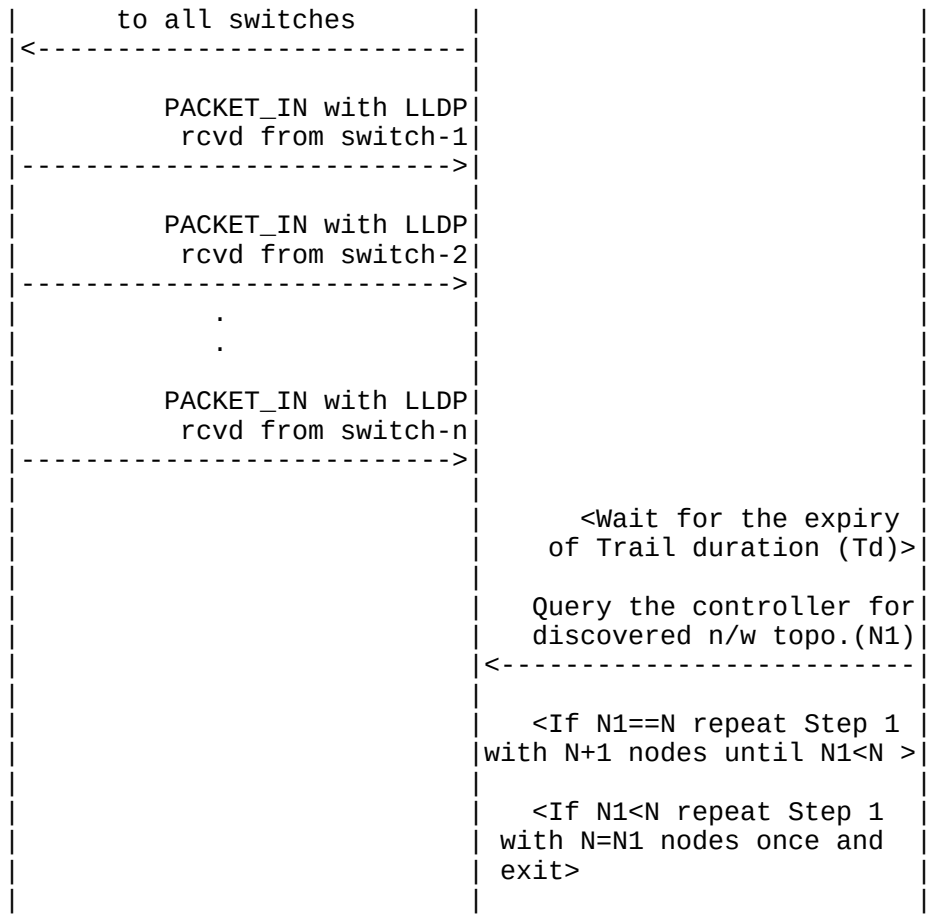
Discussion:

The value of Switch n-1 will provide Control Sessions Capacity.

B.5.2. Network Discovery Size

Procedure:





Legend:

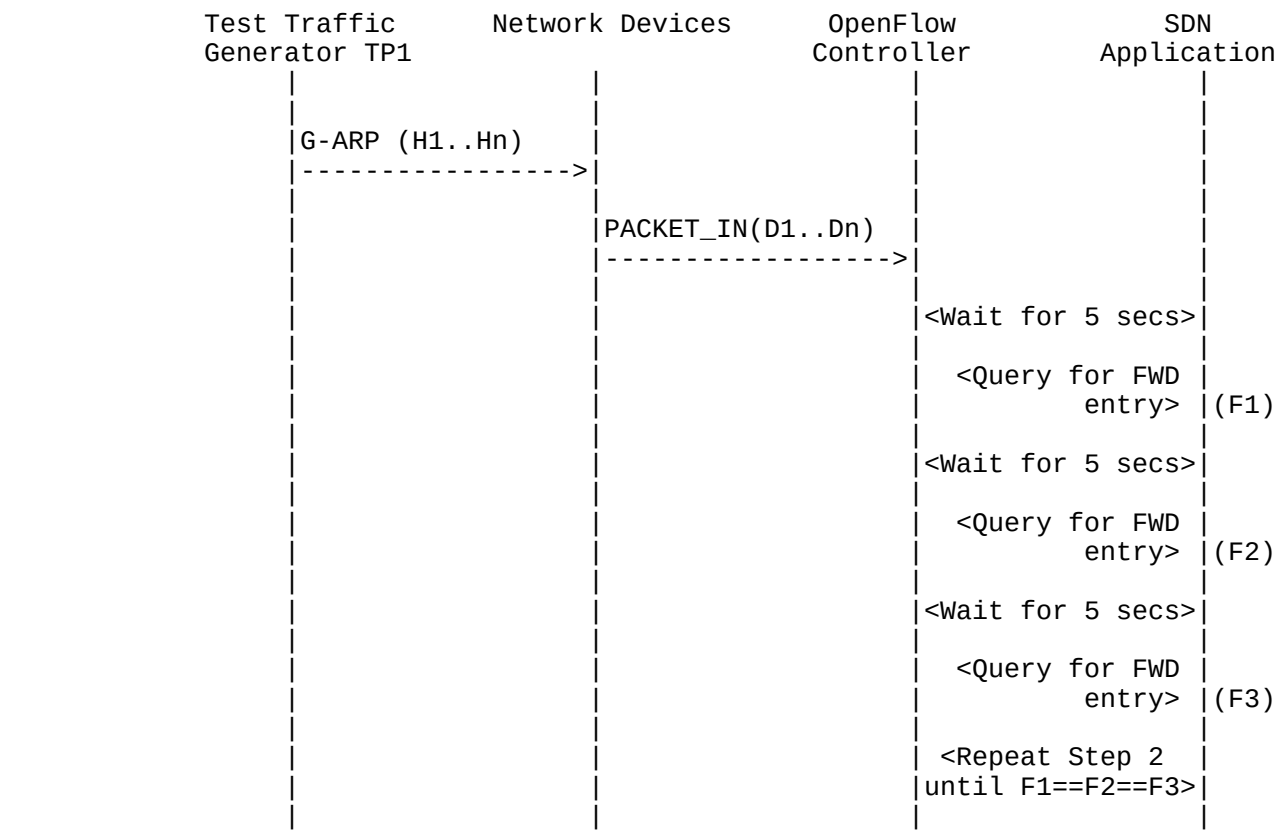
n/w topo: Network Topology
OF: OpenFlow

Discussion:

The value of N1 provides the Network Discovery Size value. The trail duration can be set to the stipulated time within which the user expects the controller to complete the discovery process.

B.5.3. Forwarding Table Capacity

Procedure:



Legend:

G-ARP: Gratuitous ARP
H1..Hn: Host 1 .. Host n
FWD: Forwarding Table

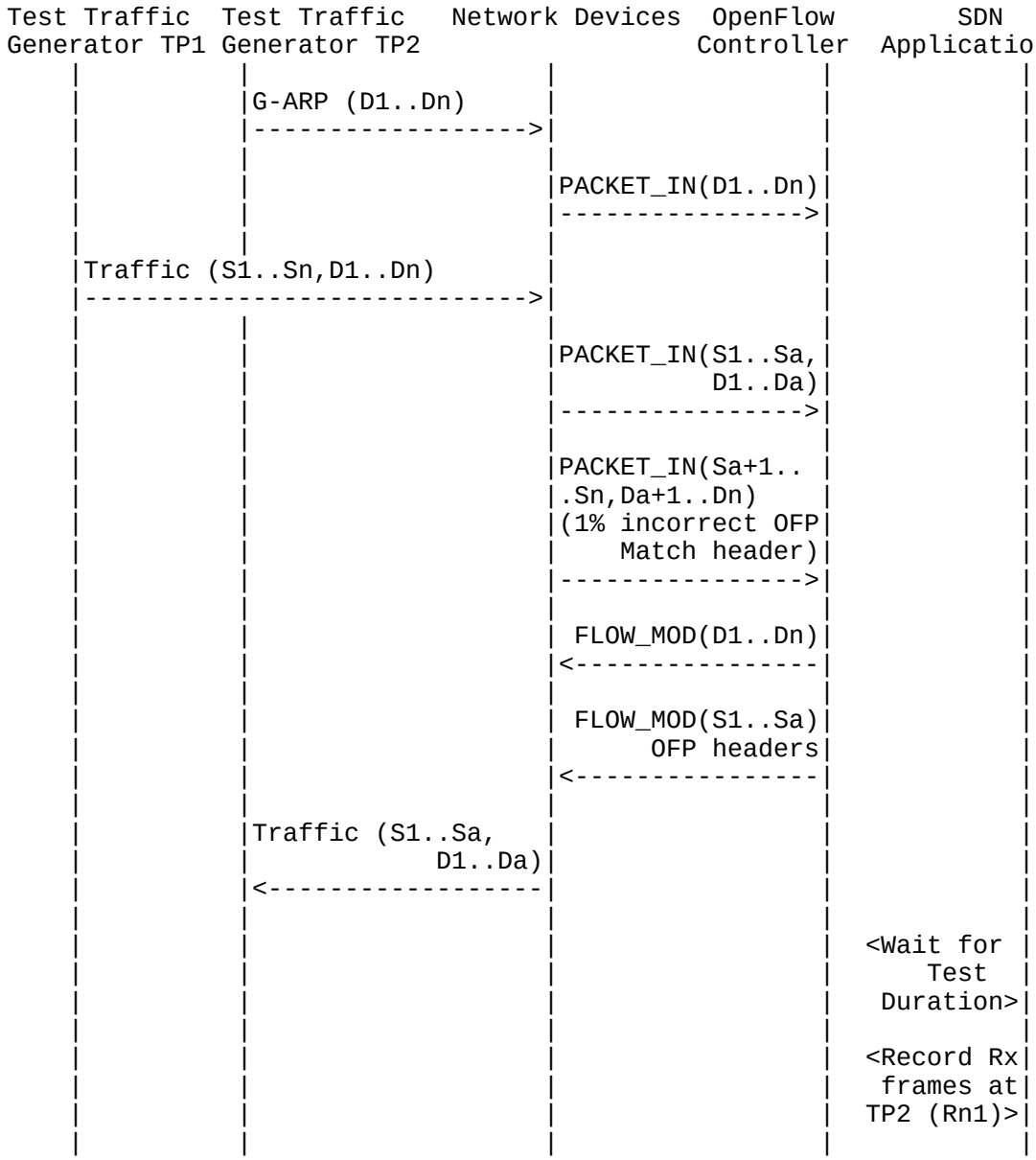
Discussion:

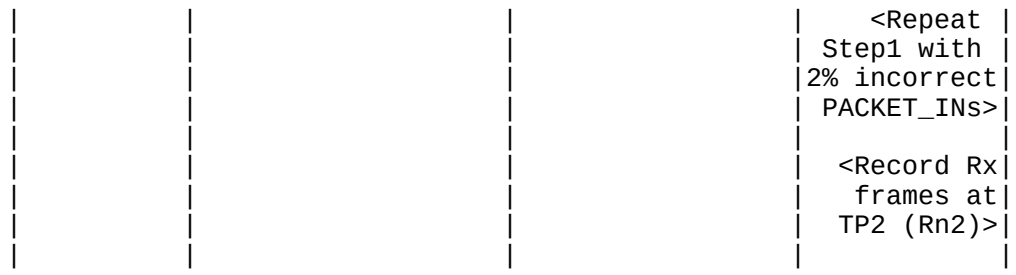
Query the controller forwarding table entries for multiple times until the three consecutive queries return the same value. The last value retrieved from the controller will provide the Forwarding Table Capacity value. The query interval is user configurable. The 5 seconds shown in this example is for representational purpose.

B.6. Security

B.6.1. Exception Handling

Procedure:





Legend:

G-ARP: Gratuitous ARP
PACKET_IN(Sa+1..Sn,Da+1..Dn): OpenFlow PACKET_IN with wrong version number
Rn1: Total number of frames received at Test Port 2 with 1% incorrect frames
Rn2: Total number of frames received at Test Port 2 with 2% incorrect frames

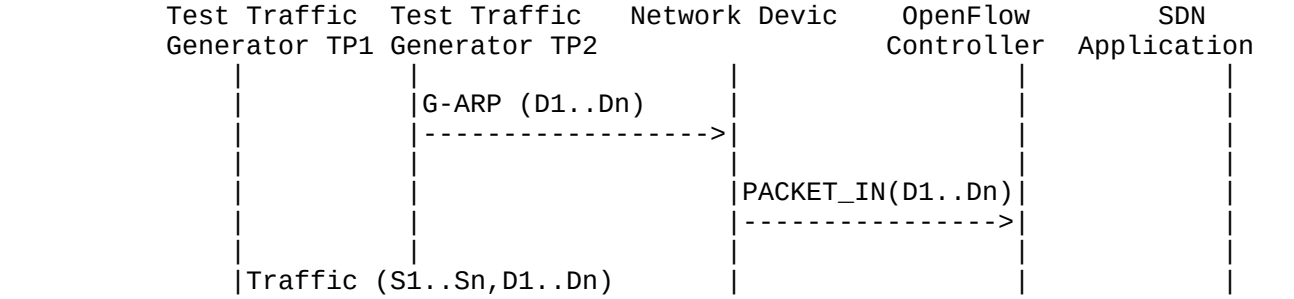
Discussion:

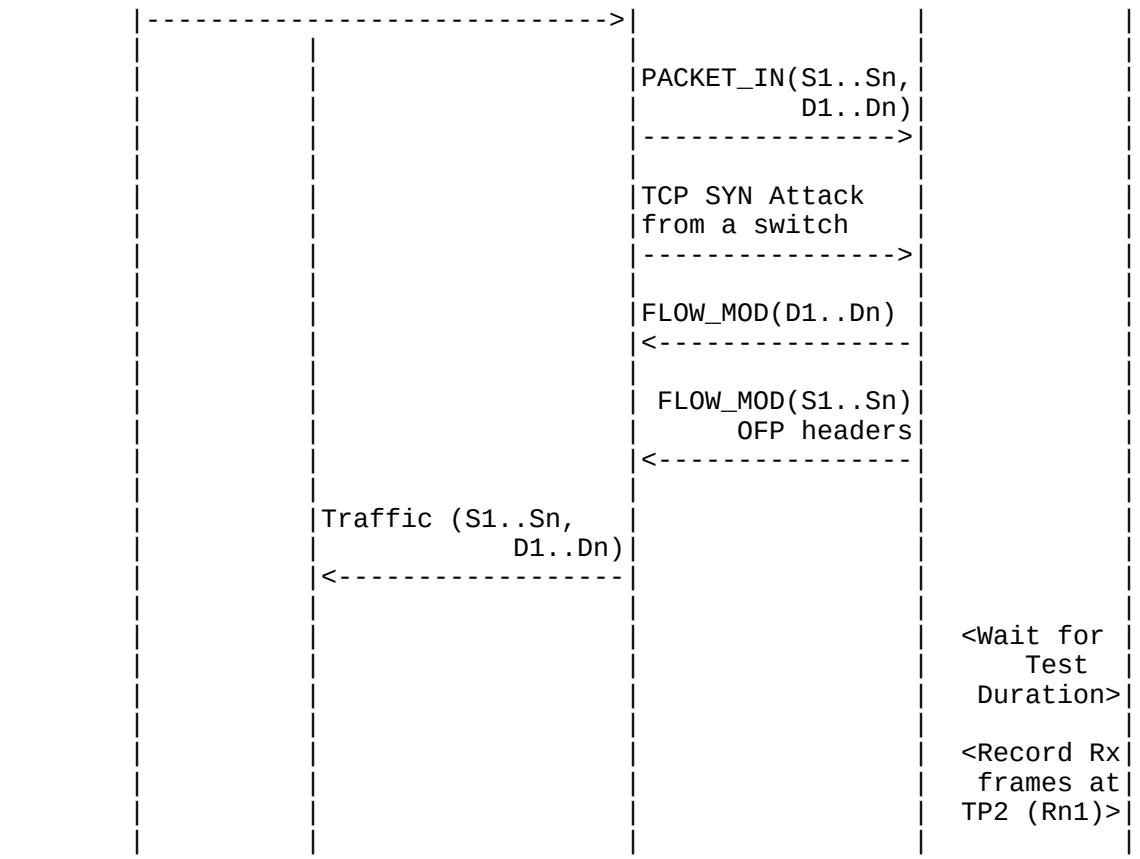
The traffic rate sent towards OpenFlow switch from Test Port 1 should be 1% higher than the Path Programming Rate. Rn1 will provide the Path Provisioning Rate of controller at 1% of incorrect frames handling and Rn2 will provide the Path Provisioning Rate of controller at 2% of incorrect frames handling.

The procedure defined above provides test steps to determine the effect of handling error packets on Path Programming Rate. Same procedure can be adopted to determine the effects on other performance tests listed in this benchmarking tests.

B.6.2. Denial of Service Handling

Procedure:





Legend:

G-ARP: Gratuitous ARP

Discussion:

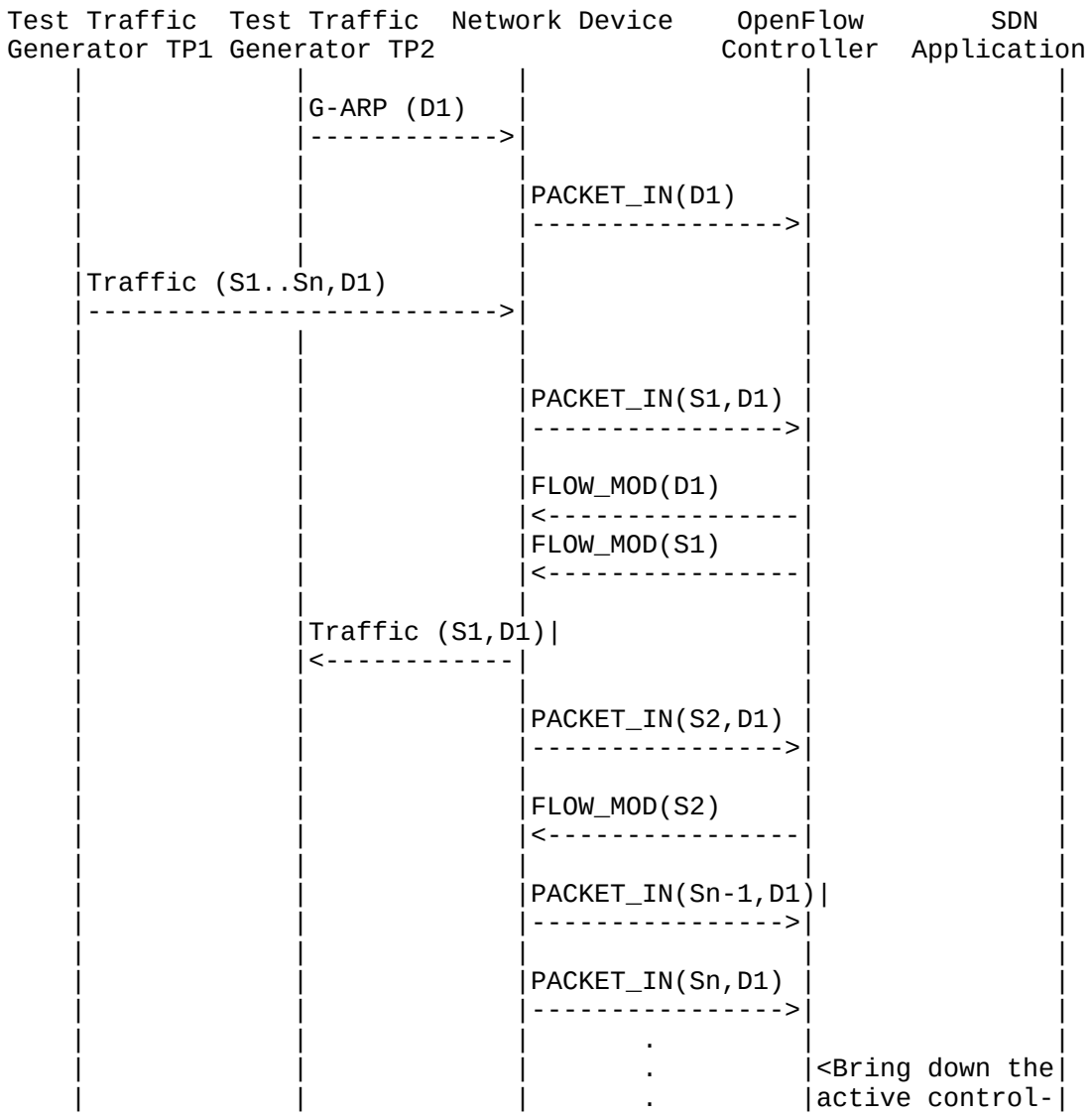
TCP SYN attack should be launched from one of the emulated/simulated OpenFlow Switch. Rn1 provides the Path Programming Rate of controller uponhandling denial of service attack.

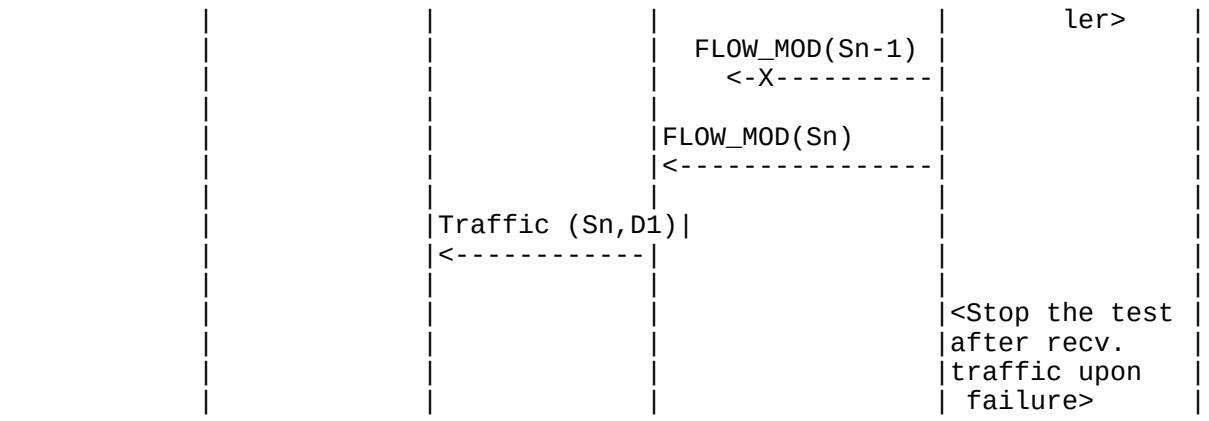
The procedure defined above provides test steps to determine the effect of handling denial of service on Path Programming Rate. Same procedure can be adopted to determine the effects on other performance tests listed in this benchmarking tests.

B.7. Reliability

B.7.1. Controller Failover Time

Procedure:





Legend:

G-ARP: Gratuitous ARP.

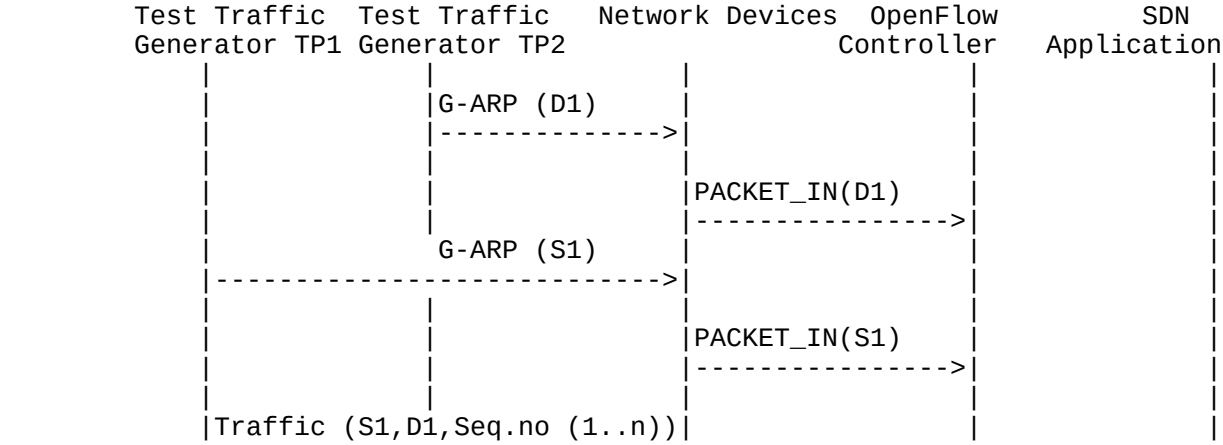
Discussion:

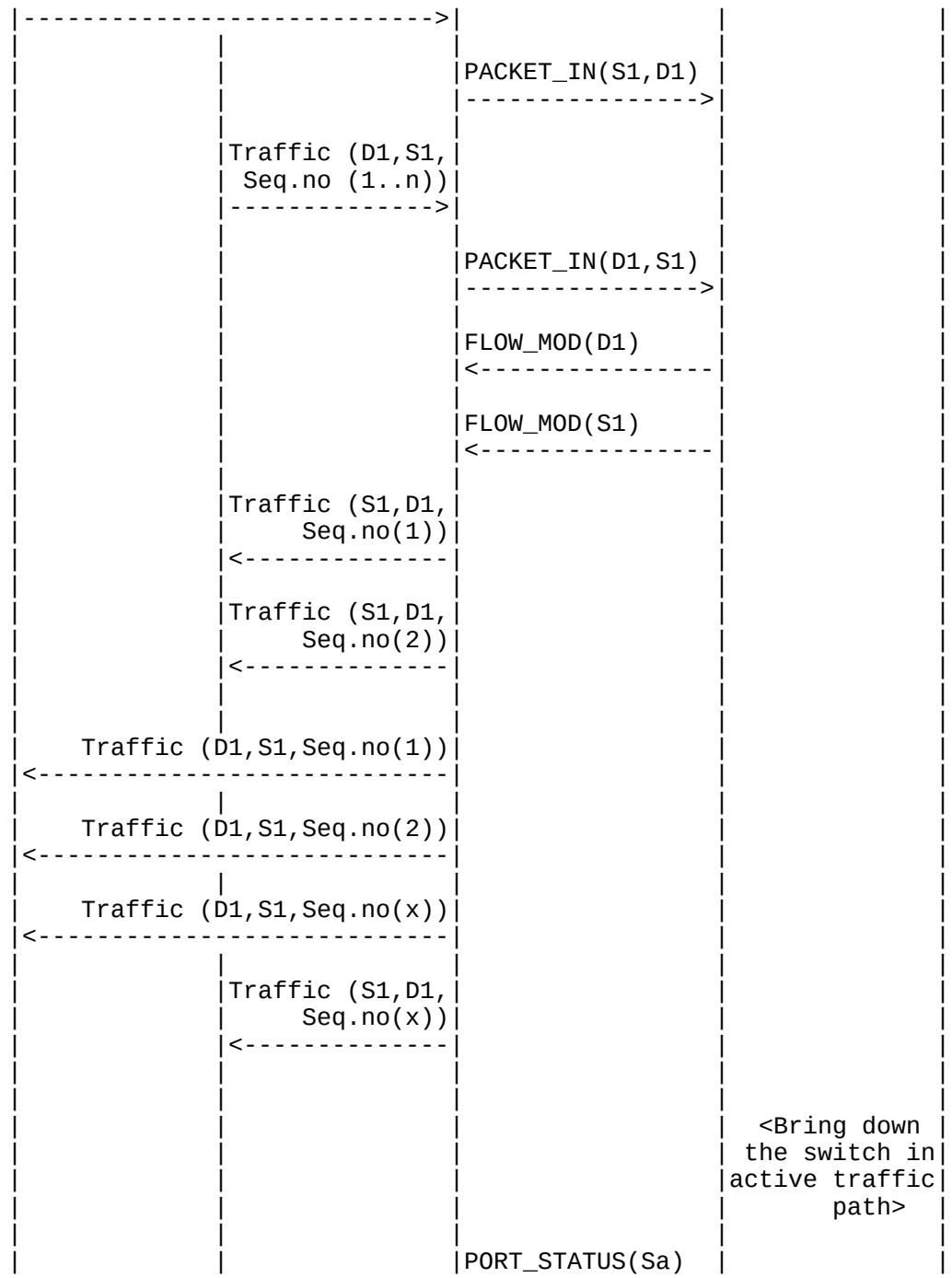
The time difference between the last valid frame received before the traffic loss and the first frame received after the traffic loss will provide the controller failover time.

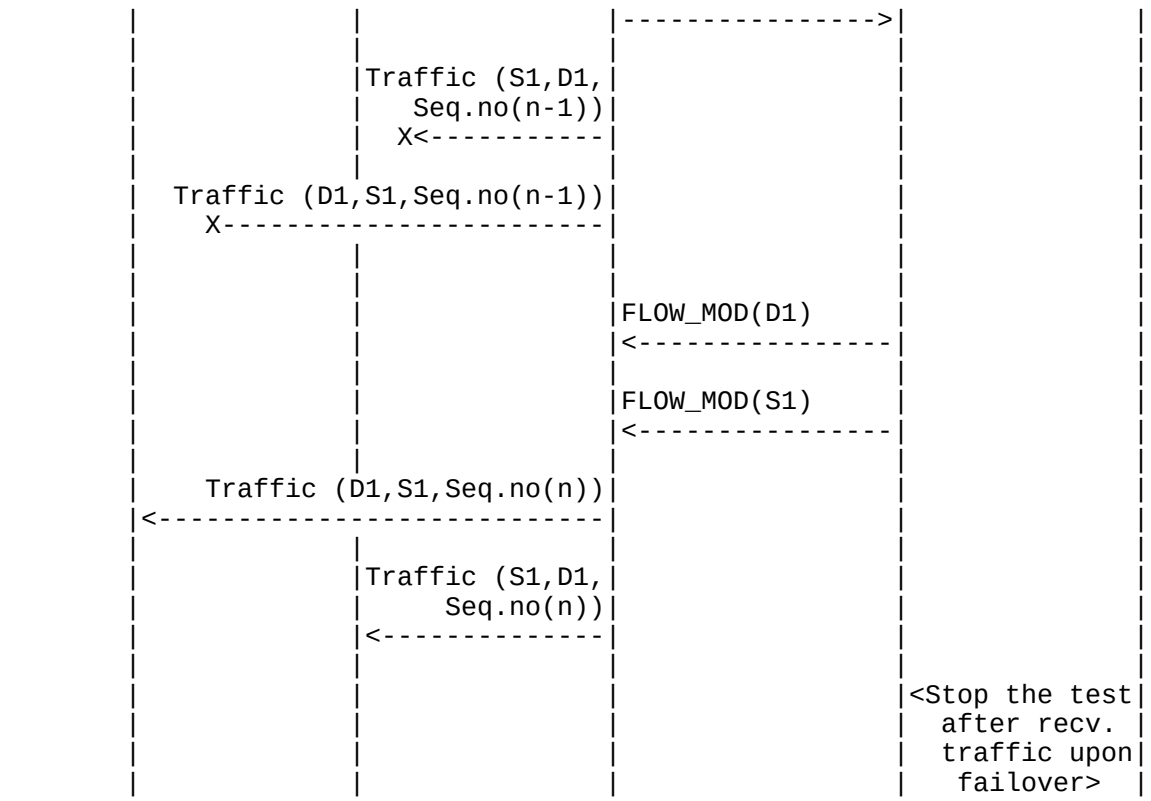
If there is no frame loss during controller failover time, the controller failover time can be deemed negligible.

B.7.2. Network Re-Provisioning Time

Procedure:




PACKET_IN(S1,D1)
Traffic (D1,S1,
Seq.no (1..n))
PACKET_IN(D1,S1)
FLOW_MOD(D1)
FLOW_MOD(S1)
Traffic (S1,D1,
Seq.no(1))
Traffic (S1,D1,
Seq.no(2))
Traffic (D1,S1,Seq.no(1))
Traffic (D1,S1,Seq.no(2))
Traffic (D1,S1,Seq.no(x))
Traffic (S1,D1,
Seq.no(x))
<Bring down
the switch in
active traffic
path>
PORT_STATUS(Sa)



Legend:

G-ARP: Gratuitous ARP message.
Seq.no: Sequence number.
Sa: Neighbor switch of the switch that was brought down.

Discussion:

The time difference between the last valid frame received before the traffic loss (Packet number with sequence number x) and the first frame received after the traffic loss (packet with sequence number n) will provide the network path re-provisioning time.

Note that the trail is valid only when the controller provisions the alternate path upon network failure.

Authors' Addresses

Bhuvaneswaran Vengainathan
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: bhuvaneswaran.vengainathan@veryxtech.com

Anton Basil
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: anton.basil@veryxtech.com

Mark Tassinari
Hewlett-Packard,
8000 Foothills Blvd,
Roseville, CA 95747

Email: mark.tassinari@hpe.com

Vishwas Manral
Nano Sec,
CA

Email: vishwas.manral@gmail.com

Sarah Banks
VSS Monitoring
930 De Guigne Drive,
Sunnyvale, CA

Email: sbanks@encrypted.net

Internet-Draft
Network Working Group
Intended Status: Informational
Expires: November 25, 2018

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Mark Tassinari
Hewlett-Packard
Vishwas Manral
Nano Sec
Sarah Banks
VSS Monitoring
May 25, 2018

Benchmarking Methodology for SDN Controller Performance
draft-ietf-bmwg-sdn-controller-benchmark-meth-09

Abstract

This document defines methodologies for benchmarking control plane performance of SDN controllers. SDN controller is a core component in software-defined networking architecture that controls the network behavior. SDN controllers have been implemented with many varying designs in order to achieve their intended network functionality. Hence, the authors have taken the approach of considering an SDN controller as a black box, defining the methodology in a manner that is agnostic to protocols and network services supported by controllers. The intent of this document is to provide a method to measure the performance of all controller implementations.

Table of Contents

1. Introduction

This document provides generic methodologies for benchmarking SDN controller performance. An SDN controller may support many northbound and southbound protocols, implement a wide range of applications, and work solely, or as a group to achieve the desired functionality. This document considers an SDN controller as a black box, regardless of design and implementation. The tests defined in the document can be used to benchmark SDN controller for performance, scalability, reliability and security independent of northbound and southbound protocols. Terminology related to benchmarking SDN controllers is described in the companion terminology document [I-D.sdn-controller-benchmark-term]. These tests can be performed on an SDN controller running as a virtual machine (VM) instance or on a bare metal server. This document is intended for those who want to measure the SDN controller performance as well as compare various SDN controllers performance.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Scope

This document defines methodology to measure the networking metrics of SDN controllers. For the purpose of this memo, the SDN controller is a function that manages and controls Network Devices. Any SDN controller without a control capability is out of scope for this memo. The tests defined in this document enable benchmarking of SDN Controllers in two ways; as a standalone controller and as a cluster of homogeneous controllers. These tests are recommended for execution in lab environments rather than in live network deployments. Performance benchmarking of a federation of controllers, set of SDN controllers managing different domains, is beyond the scope of this document.

3. Test Setup

The tests defined in this document enable measurement of an SDN controller’s performance in standalone mode and cluster mode. This section defines common reference topologies that are later referred to in individual tests.

3.1. Test setup - Controller working in Standalone Mode

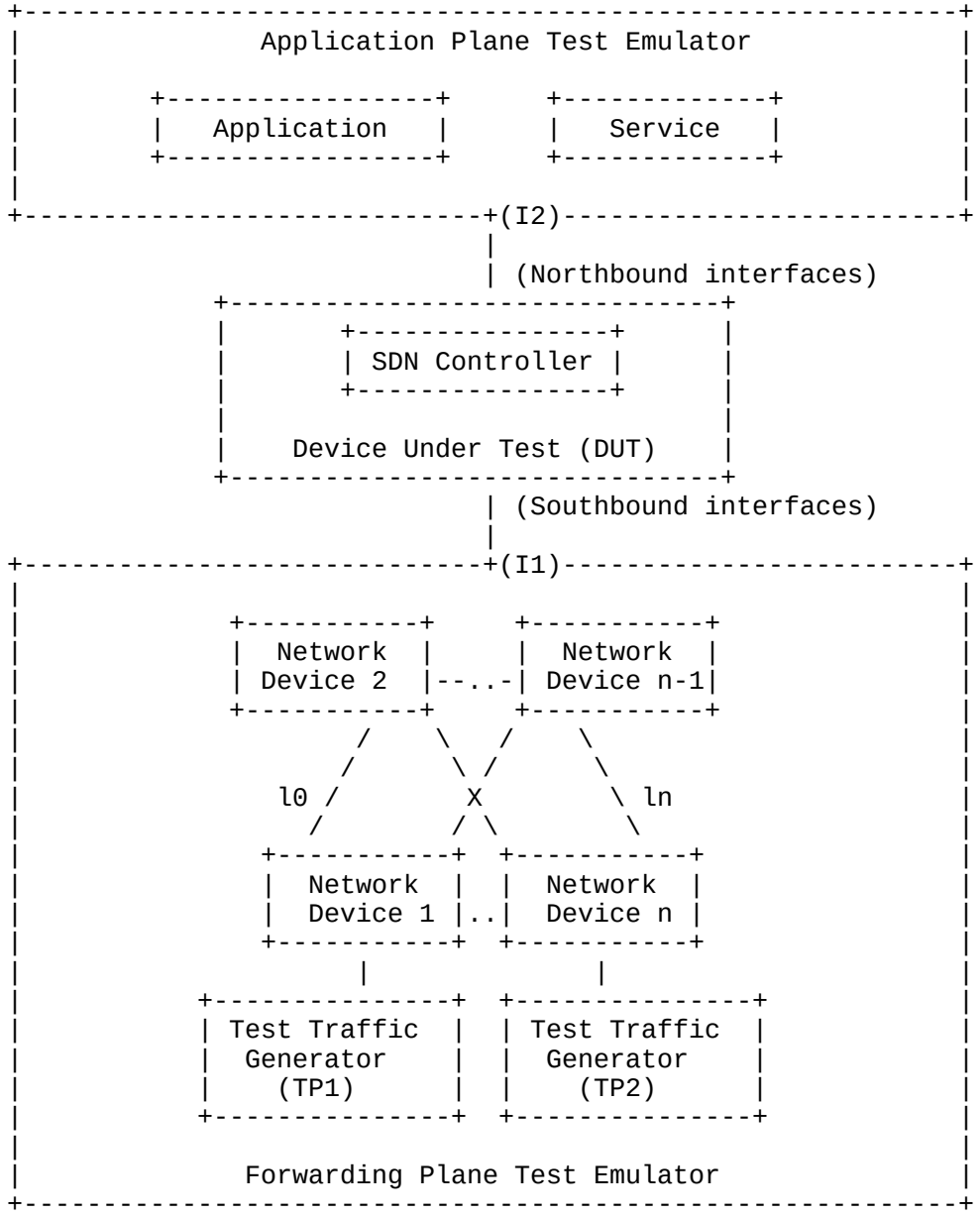


Figure 1

3.2. Test setup - Controller working in Cluster Mode

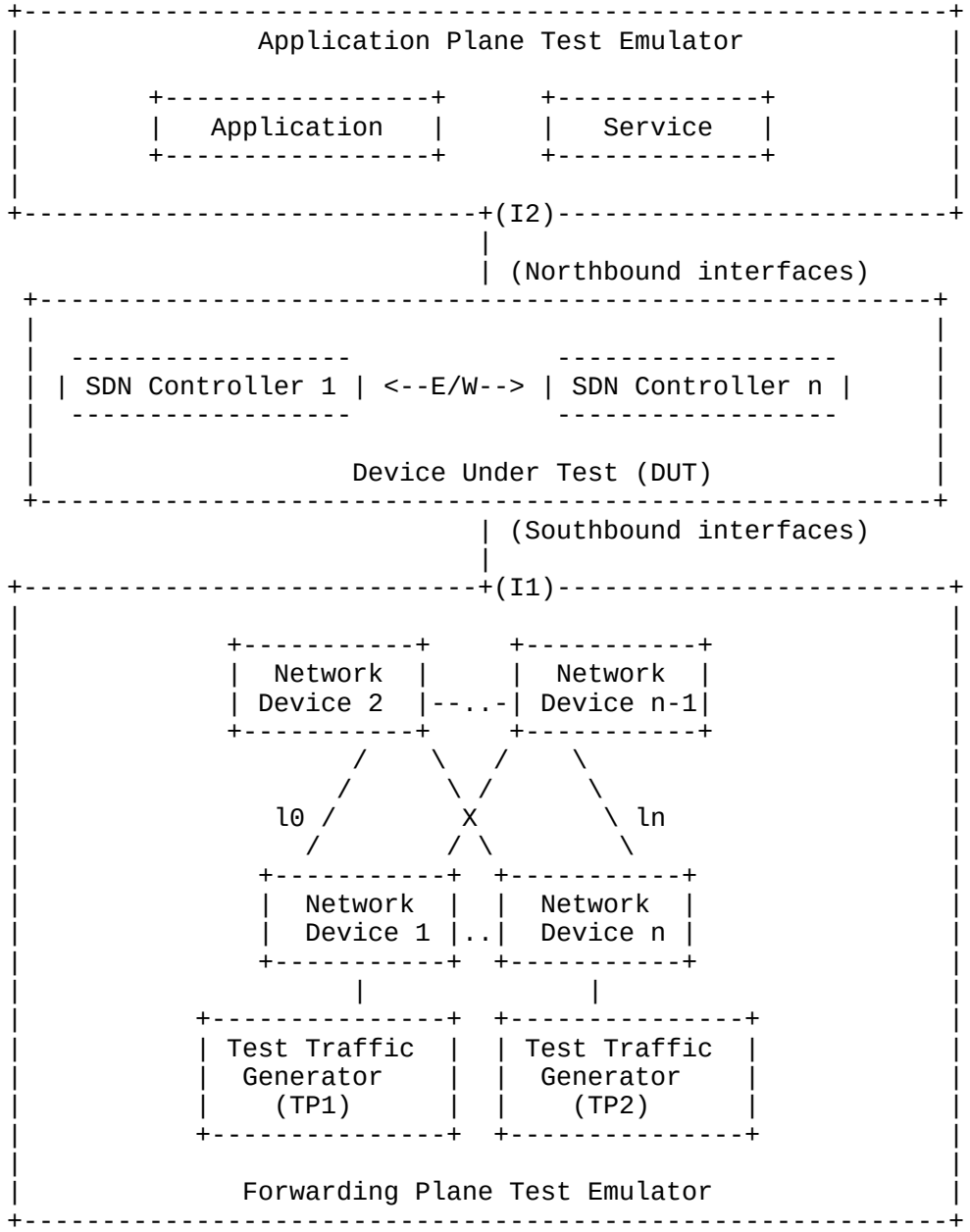


Figure 2

4. Test Considerations

4.1. Network Topology

The test cases SHOULD use Leaf-Spine topology with at least 2 Network Devices in the topology for benchmarking. The test traffic generators TP1 and TP2 SHOULD be connected to the leaf Network Device 1 and the leaf Network Device n. To achieve a complete performance characterization of the SDN controller, it is recommended that the controller be benchmarked for many network topologies and a varying number of Network Devices. Further, care should be taken to make sure that a loop prevention mechanism is enabled either in the SDN controller, or in the network when the topology contains redundant network paths.

4.2. Test Traffic

Test traffic is used to notify the controller about the asynchronous arrival of new flows. The test cases SHOULD use frame sizes of 128, 512 and 1508 bytes for benchmarking. Tests using jumbo frames are optional.

4.3. Test Emulator Requirements

The Test Emulator SHOULD time stamp the transmitted and received control messages to/from the controller on the established network connections. The test cases use these values to compute the controller processing time.

4.4. Connection Setup

There may be controller implementations that support unencrypted and encrypted network connections with Network Devices. Further, the controller may have backward compatibility with Network Devices running older versions of southbound protocols. It may be useful to measure the controller performance with one or more applicable connection setup methods defined below. For cases with encrypted communications between the controller and the switch, key management and key exchange MUST take place before any performance or benchmark measurements.

1. Unencrypted connection with Network Devices, running same protocol version.
2. Unencrypted connection with Network Devices, running different protocol versions.
   Example:

   a. Controller running current protocol version and switch running older protocol version
   b. Controller running older protocol version and switch running current protocol version
3. Encrypted connection with Network Devices, running same protocol version
4. Encrypted connection with Network Devices, running different protocol versions.
   Example:
   a. Controller running current protocol version and switch running older protocol version
   b. Controller running older protocol version and switch running current protocol version

4.5. Measurement Point Specification and Recommendation

The measurement accuracy depends on several factors including the point of observation where the indications are captured. For example, the notification can be observed at the controller or test emulator. The test operator SHOULD make the observations/ measurements at the interfaces of test emulator unless it is explicitly mentioned otherwise in the individual test. In any case, the locations of measurement points MUST be reported.

4.6. Connectivity Recommendation

The SDN controller in the test setup SHOULD be connected directly with the forwarding and the management plane test emulators to avoid any delays or failure introduced by the intermediate devices during benchmarking tests. When the controller is implemented as a virtual machine, details of the physical and logical connectivity MUST be reported.

4.7. Test Repeatability

To increase the confidence in measured result, it is recommended that each test RECOMMENDED be repeated a minimum of 10 times.

4.8. Test Reporting

Each test has a reporting format that contains some global and identical reporting components, and some individual components that are specific to individual tests. The following test configuration parameters and controller settings parameters MUST be reflected in the test report.

Test Configuration Parameters:

1. Controller name and version
2. Northbound protocols and versions
3. Southbound protocols and versions
4. Controller redundancy mode (Standalone or Cluster Mode)
5. Connection setup (Unencrypted or Encrypted)
6. Network Device Type (Physical or Virtual or Emulated)
7. Number of Nodes
8. Number of Links
9. Dataplane Test Traffic Type
10. Controller System Configuration (e.g., Physical or Virtual Machine, CPU, Memory, Caches, Operating System, Interface Speed, Storage)
11. Reference Test Setup (e.g., Section 3.1 etc.,)

Controller Settings Parameters:

1. Topology re-discovery timeout
2. Controller redundancy mode (e.g., active-standby etc.,)
3. Controller state persistence enabled/disabled

To ensure the repeatability of test, the following capabilities of test emulator SHOULD be reported

1. Maximum number of Network Devices that the forwarding plane emulates
2. Control message processing time (e.g., Topology Discovery Messages)

One way to determine the above two values are to simulate the required control sessions and messages from the control plane.

5. Benchmarking Tests

5.1. Performance

5.1.1. Network Topology Discovery Time

Objective:

The time taken by controller(s) to determine the complete network topology, defined as the interval starting with the first discovery message from the controller(s) at its Southbound interface, ending with all features of the static topology determined.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST support network discovery.
2. Tester should be able to retrieve the discovered topology information either through the controller's management interface, or northbound interface to determine if the discovery was successful and complete.
3. Ensure that the controller's topology re-discovery timeout has been set to the maximum value to avoid initiation of re-discovery process in the middle of the test.

Procedure:

1. Ensure that the controller is operational, its network applications, northbound and southbound interfaces are up and running.
2. Establish the network connections between controller and Network Devices.
3. Record the time for the first discovery message (Tm1) received from the controller at forwarding plane test emulator interface I1.
4. Query the controller every t seconds (RECOMMENDED value for t is 3) to obtain the discovered network topology information through the northbound interface or the management interface and compare it with the deployed network topology information.
5. Stop the trial when the discovered topology information matches the deployed network topology, or when the discovered topology information return the same details for 3 consecutive queries.
6. Record the time last discovery message (Tmn) sent to controller from the forwarding plane test emulator interface (I1) when the trial completed successfully. (e.g., the topology matches).

Measurement:

Topology Discovery Time Tr1 = Tmn-Tm1.

$$\text{Average Topology Discovery Time (TDm)} = \frac{Tr1 + Tr2 + Tr3 \,..\, Trn}{\text{Total Trials}}$$

$$\text{Topology Discovery Time Variance (TDv)} \quad \frac{SUM[SQUAREOF(Tri-TDm)]}{\text{Total Trials} - 1}$$

Reporting Format:

The Topology Discovery Time results MUST be reported in the format of a table, with a row for each successful iteration. The last row of the table indicates the Topology Discovery Time variance and the previous row indicates the average Topology Discovery Time.

If this test is repeated with varying number of nodes over the same topology, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the Number of nodes (N), the Y coordinate SHOULD be the average Topology Discovery Time.

5.1.2. Asynchronous Message Processing Time

Objective:

The time taken by controller(s) to process an asynchronous message, defined as the interval starting with an asynchronous message from a network device after the discovery of all the devices by the controller(s), ending with a response message from the controller(s) at its Southbound interface.

Reference Test Setup:

This test SHOULD use one of the test setup described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST have successfully completed the network topology discovery for the connected Network Devices.

Procedure:

1. Generate asynchronous messages from every connected Network Device, to the SDN controller, one at a time in series from the forwarding plane test emulator for the trial duration.
2. Record every request transmit time (T1) and the corresponding response received time (R1) at the forwarding plane test emulator interface (I1) for every successful message exchange.

Measurement:

$$\text{Asynchronous Message Processing Time } Tr1 = \frac{SUM\{Ri\} - SUM\{Ti\}}{Nrx}$$

Where Nrx is the total number of successful messages exchanged

$$\text{Average Asynchronous Message Processing Time} = \frac{Tr1 + Tr2 + Tr3..Trn}{\text{Total Trials}}$$

Asynchronous Message Processing Time Variance (TAMv) =

$$\frac{SUM[SQUAREOF(Tri\text{-}TAMm)]}{\text{Total Trials} - 1}$$

Where TAMm is the Average Asynchronous Message Processing Time.

Reporting Format:

The Asynchronous Message Processing Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Asynchronous Message Processing Time variance and the previous row indicates the average Asynchronous Message Processing Time.

The report SHOULD capture the following information in addition to the configuration parameters captured in section 4.8.

- Successful messages exchanged (Nrx)

- Percentage of unsuccessful messages exchanged, computed using the formula (1 - Nrx/Ntx) * 100), Where Ntx is the total number of messages transmitted to the controller.

If this test is repeated with varying number of nodes with same topology, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the Number of nodes (N), the Y coordinate SHOULD be the average Asynchronous Message Processing Time.

5.1.3. Asynchronous Message Processing Rate

Objective:

Measure the number of responses to asynchronous messages (such as new flow arrival notification message, link down, etc.) for which

the controller(s) performed processing and replied with a valid and productive (non-trivial) response message

This test will measure two benchmarks on Asynchronous Message Processing Rate using a single procedure. The two benchmarks are (see section 2.3.1.3 of [I-D.sdn-controller-benchmark-term]):

1. Loss-free Asynchronous Message Processing Rate

2. Maximum Asynchronous Message Processing Rate

Here two benchmarks are determined through a series of trials where the number of messages are sent to the controller(s), and the responses from the controller(s) are counted over the trial duration. The message response rate and the message loss ratio are calculated for each trial.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller(s) MUST have successfully completed the network topology discovery for the connected Network Devices.
2. Choose and record the Trial Duration (Td), the sending rate step-size (STEP), the tolerance on equality for two consecutive trials (P%),and the maximum possible message sending rate (Ntx1/Td).

Procedure:

1. Generate asynchronous messages continuously at the maximum possible rate on the established connections from all the emulated/simulated Network Devices for the given trial Duration (Td).
2. Record the total number of responses received from the controller (Nrx1) as well as the number of messages sent (Ntx1) to the controller within the trial duration (Td).
3. Calculate the Asynchronous Message Processing Rate (Tr1) and the Message Loss Ratio (Lr1). Ensure that the controller(s) have returned to normal operation.
4. Repeat the trial by reducing the asynchronous message sending rate used in last trial by the STEP size.
5. Continue repeating the trials and reducing the sending rate until both the maximum value of Nrxn (number of responses received from

the controller) and the Nrxn corresponding to zero loss ratio have been found.

6. The trials corresponding to the benchmark levels MUST be repeated using the same asynchronous message rates until the responses received from the controller are equal (+/-P%) for two consecutive trials.
7. Record the number of responses received from the controller (Nrxn) as well as the number of messages sent (Ntxn) to the controller in the last trial.

Measurement:

$$\text{Asynchronous Message Processing Rate Trn} = \frac{Nrxn}{Td}$$

Maximum Asynchronous Message Processing Rate = MAX(Trn) for all n

$$\text{Asynchronous Message Loss Ratio Lrn} = 1 - \frac{Nrxn}{Ntxn}$$

Loss-free Asynchronous Message Processing Rate = MAX(Trn) given Lrn=0

Reporting Format:

The Asynchronous Message Processing Rate results MUST be reported in the format of a table with a row for each trial.

The table should report the following information in addition to the configuration parameters captured in section 4.8, with columns:

- Offered rate (Ntxn/Td)

- Asynchronous Message Processing Rate (Trn)

- Loss Ratio (Lr)

- Benchmark at this iteration (blank for none, Maximum, Loss-Free)

The results MAY be presented in the form of a graph. The X axis SHOULD be the Offered rate, and dual Y axes would represent Asynchronous Message Processing Rate and Loss Ratio, respectively.

If this test is repeated with varying number of nodes over same topology, the results SHOULD be reported in the form of a graph. The

X axis SHOULD be the Number of nodes (N), the Y axis SHOULD be the Asynchronous Message Processing Rate. Both the Maximum and the Loss-Free Rates should be plotted for each N.

5.1.4. Reactive Path Provisioning Time

Objective:

The time taken by the controller to setup a path reactively between source and destination node, defined as the interval starting with the first flow provisioning request message received by the controller(s) at its Southbound interface, ending with the last flow provisioning response message sent from the controller(s) at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document. The number of Network Devices in the path is a parameter of the test that may be varied from 2 to maximum discovery size in repetitions of this test.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should have the knowledge about the location of destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for ’flow miss’ in Network Device is configured to ’send to controller’.
4. Ensure that each Network Device in a path requires the controller to make the forwarding decision while paving the entire path.

Procedure:

1. Send a single traffic stream from the test traffic generator TP1 to test traffic generator TP2.
2. Record the time of the first flow provisioning request message sent to the controller (Tsf1) from the Network Device at the forwarding plane test emulator interface (I1).
3. Wait for the arrival of first traffic frame at the Traffic Endpoint TP2 or the expiry of trial duration (Td).
4. Record the time of the last flow provisioning response message received from the controller (Tdf1) to the Network Device at the forwarding plane test emulator interface (I1).

Measurement:

Reactive Path Provisioning Time Tr1 = Tdf1-Tsf1.

$$\text{Average Reactive Path Provisioning Time} = \frac{Tr1 + Tr2 + Tr3 \,..\, Trn}{\text{Total Trials}}$$

$$\text{Reactive Path Provisioning Time Variance(TRPv)} \frac{\text{SUM[SQUAREOF(Tri-TRPm)]}}{\text{Total Trials -1}}$$

Where TRPm is the Average Reactive Path Provisioning Time.

Reporting Format:

The Reactive Path Provisioning Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Reactive Path Provisioning Time variance and the previous row indicates the Average Reactive Path Provisioning Time.

The report should capture the following information in addition to the configuration parameters captured in section 4.8.

- Number of Network Devices in the path

5.1.5. Proactive Path Provisioning Time

Objective:

The time taken by the controller to setup a path proactively between source and destination node, defined as the interval starting with the first proactive flow provisioned in the controller(s) at its Northbound interface, ending with the last flow provisioning response message sent from the controller(s) at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should have the knowledge about the location of destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for flow miss in Network Device is 'drop'.

Procedure:

1. Send a single traffic stream from test traffic generator TP1 to TP2.
2. Install the flow entries to reach from test traffic generator TP1 to the test traffic generator TP2 through controller's northbound or management interface.
3. Wait for the arrival of first traffic frame at the test traffic generator TP2 or the expiry of trial duration (Td).
4. Record the time when the proactive flow is provisioned in the Controller (Tsf1) at the management plane test emulator interface I2.
5. Record the time of the last flow provisioning message received from the controller (Tdf1) at the forwarding plane test emulator interface I1.

Measurement:

Proactive Flow Provisioning Time Tr1 = Tdf1-Tsf1.

$$\text{Average Proactive Path Provisioning Time} = \frac{Tr1 + Tr2 + Tr3 \,..\, Trn}{\text{Total Trials}}$$

$$\text{Proactive Path Provisioning Time Variance(TPPv)} = \frac{\text{SUM[SQUAREOF(Tri-TPPm)]}}{\text{Total Trials -1}}$$

Where TPPm is the Average Proactive Path Provisioning Time.

Reporting Format:

The Proactive Path Provisioning Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Proactive Path Provisioning Time variance and

the previous row indicates the Average Proactive Path Provisioning Time.

The report should capture the following information in addition to the configuration parameters captured in section 4.8.

- Number of Network Devices in the path

5.1.6. Reactive Path Provisioning Rate

Objective:

The maximum number of independent paths a controller can concurrently establish per second between source and destination nodes reactively, defined as the number of paths provisioned per second by the controller(s) at its Southbound interface for the flow provisioning requests received for path provisioning at its Southbound interface between the start of the test and the expiry of given trial duration.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should have the knowledge about the location of destination addresses for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for ’flow miss’ in Network Device is configured to ’send to controller’.
4. Ensure that each Network Device in a path requires the controller to make the forwarding decision while provisioning the entire path.

Procedure:

1. Send traffic with unique source and destination addresses from test traffic generator TP1.
2. Record total number of unique traffic frames (Ndf) received at the test traffic generator TP2 within the trial duration (Td).

Measurement:

$$\text{Reactive Path Provisioning Rate Tr1} = \frac{\text{Ndf}}{\text{Td}}$$

$$\text{Average Reactive Path Provisioning Rate} = \frac{\text{Tr1 + Tr2 + Tr3 .. Trn}}{\text{Total Trials}}$$

$$\text{Reactive Path Provisioning Rate Variance(RPPv)} \frac{\text{SUM[SQUAREOF(Tri-RPPm)]}}{\text{Total Trials -1}}$$

Where RPPm is the Average Reactive Path Provisioning Rate.

Reporting Format:

The Reactive Path Provisioning Rate results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Reactive Path Provisioning Rate variance and the previous row indicates the Average Reactive Path Provisioning Rate.

The report should capture the following information in addition to the configuration parameters captured in section 4.8.

- Number of Network Devices in the path

- Offered rate

5.1.7. Proactive Path Provisioning Rate

Objective:

Measure the maximum number of independent paths a controller can concurrently establish per second between source and destination nodes proactively, defined as the number of paths provisioned per second by the controller(s) at its Southbound interface for the paths requested in its Northbound interface between the start of the test and the expiry of given trial duration. The measurement is based on dataplane observations of successful path activation

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST contain the network topology information for the deployed network topology.

2. The controller should have the knowledge about the location of destination addresses for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.

3. Ensure that the default action for flow miss in Network Device is 'drop'.

Procedure:

1. Send traffic continuously with unique source and destination addresses from test traffic generator TP1.

2. Install corresponding flow entries to reach from simulated sources at the test traffic generator TP1 to the simulated destinations at test traffic generator TP2 through controller's northbound or management interface.

3. Record total number of unique traffic frames received Ndf) at the test traffic generator TP2 within the trial duration (Td).

Measurement:

$$\text{Proactive Path Provisioning Rate Tr1} = \frac{\text{Ndf}}{\text{Td}}$$

$$\text{Average Proactive Path Provisioning Rate} = \frac{\text{Tr1} + \text{Tr2} + \text{Tr3} .. \text{Trn}}{\text{Total Trials}}$$

$$\text{Proactive Path Provisioning Rate Variance(PPPv)} \frac{\text{SUM[SQUAREOF(Tri-PPPm)]}}{\text{Total Trials} -1}$$

Where PPPm is the Average Proactive Path Provisioning Rate.

Reporting Format:

The Proactive Path Provisioning Rate results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Proactive Path Provisioning Rate variance and the previous row indicates the Average Proactive Path Provisioning Rate.

The report should capture the following information in addition to the configuration parameters captured in section 4.8.

- Number of Network Devices in the path

- Offered rate

5.1.8. Network Topology Change Detection Time

Objective:

The amount of time required for the controller to detect any changes in the network topology, defined as the interval starting with the notification message received by the controller(s) at its Southbound interface, ending with the first topology rediscovery messages sent from the controller(s) at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST have successfully discovered the network topology information for the deployed network topology.

2. The periodic network discovery operation should be configured to twice the Trial duration (Td) value.

Procedure:

1. Trigger a topology change event by bringing down an active Network Device in the topology.

2. Record the time when the first topology change notification is sent to the controller (Tcn) at the forwarding plane test emulator interface (I1).

3. Stop the trial when the controller sends the first topology re-discovery message to the Network Device or the expiry of trial duration (Td).

4. Record the time when the first topology re-discovery message is received from the controller (Tcd) at the forwarding plane test emulator interface (I1)

Measurement:

Network Topology Change Detection Time Tr1 = Tcd-Tcn.

$$\text{Average Network Topology Change Detection Time} = \frac{Tr1 + Tr2 + Tr3 .. Trn}{\text{Total Trials}}$$

Network Topology Change Detection Time Variance(NTDv) =

$$\frac{\text{SUM[SQUAREOF(Tri-NTDm)]}}{\text{Total Trials -1}}$$

Where NTDm is the Average Network Topology Change Detection Time.

Reporting Format:

The Network Topology Change Detection Time results MUST be reported in the format of a table with a row for each iteration. The last row of the table indicates the Network Topology Change Detection Time variance and the previous row indicates the average Network Topology Change Time.

5.2. Scalability

5.2.1. Control Session Capacity

Objective:

Measure the maximum number of control sessions the controller can maintain, defined as the number of sessions that the controller can accept from network devices, starting with the first control

session, ending with the last control session that the controller(s) accepts at its Southbound interface.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Procedure:

1. Establish control connection with controller from every Network Device emulated in the forwarding plane test emulator.
2. Stop the trial when the controller starts dropping the control connections.
3. Record the number of successful connections established with the controller (CCn) at the forwarding plane test emulator.

Measurement:

Control Sessions Capacity = CCn.

Reporting Format:

The Control Session Capacity results MUST be reported in addition to the configuration parameters captured in section 4.8.

5.2.2. Network Discovery Size

Objective:

Measure the network size (number of nodes, links and hosts) that a controller can discover, defined as the size of a network that the controller(s) can discover, starting from a network topology given by the user for discovery, ending with the topology that the controller(s) could successfully discover.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller MUST support automatic network discovery.

2. Tester should be able to retrieve the discovered topology information either through controller's management interface or northbound interface.

Procedure:

1. Establish the network connections between controller and network nodes.
2. Query the controller every t seconds (RECOMMENDED value for t is 30) to obtain the discovered network topology information through the northbound interface or the management interface.
3. Stop the trial when the discovered network topology information remains the same as that of last two query responses.
4. Compare the obtained network topology information with the deployed network topology information.
5. If the comparison is successful, increase the number of nodes by 1 and repeat the trial.
   If the comparison is unsuccessful, decrease the number of nodes by 1 and repeat the trial.
6. Continue the trial until the comparison of step 5 is successful.
7. Record the number of nodes for the last trial run (Ns) where the topology comparison was successful.

Measurement:

Network Discovery Size = Ns.

Reporting Format:

The Network Discovery Size results MUST be reported in addition to the configuration parameters captured in section 4.8.

5.2.3. Forwarding Table Capacity

Objective:

Measure the maximum number of flow entries a controller can manage in its Forwarding table.

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. The controller Forwarding table should be empty.
2. Flow Idle time MUST be set to higher or infinite value.
3. The controller MUST have successfully completed network topology discovery.
4. Tester should be able to retrieve the forwarding table information either through controller's management interface or northbound interface.

Procedure:

Reactive Flow Provisioning Mode:

1. Send bi-directional traffic continuously with unique source and destination addresses from test traffic generators TP1 and TP2 at the asynchronous message processing rate of controller.
2. Query the controller at a regular interval (e.g., 5 seconds) for the number of learned flow entries from its northbound interface.
3. Stop the trial when the retrieved value is constant for three consecutive iterations and record the value received from the last query (Nrp).

Proactive Flow Provisioning Mode:

1. Install unique flows continuously through controller's northbound or management interface until a failure response is received from the controller.
2. Record the total number of successful responses (Nrp).

Note:

Some controller designs for proactive flow provisioning mode may require the switch to send flow setup requests in order to generate flow setup responses. In such cases, it is recommended to generate bi-directional traffic for the provisioned flows.

Measurement:

Proactive Flow Provisioning Mode:

Max Flow Entries = Total number of flows provisioned (Nrp)

Reactive Flow Provisioning Mode:

Max Flow Entries = Total number of learned flow entries (Nrp)

Forwarding Table Capacity = Max Flow Entries.

Reporting Format:

The Forwarding Table Capacity results MUST be tabulated with the following information in addition to the configuration parameters captured in section 4.8.

- Provisioning Type (Proactive/Reactive)

5.3. Security

5.3.1. Exception Handling

Objective:

Determine the effect of handling error packets and notifications on performance tests. The impact MUST be measured for the following performance tests

a. Path Provisioning Rate

b. Path Provisioning Time

c. Network Topology Change Detection Time

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. This test MUST be performed after obtaining the baseline measurement results for the above performance tests.
2. Ensure that the invalid messages are not dropped by the intermediate devices connecting the controller and Network Devices.

Procedure:

1. Perform the above listed performance tests and send 1% of messages from the Asynchronous Message Processing Rate as invalid messages from the connected Network Devices emulated at the forwarding plane test emulator.
2. Perform the above listed performance tests and send 2% of messages from the Asynchronous Message Processing Rate as invalid messages from the connected Network Devices emulated at the forwarding plane test emulator.

Note:

Invalid messages can be frames with incorrect protocol fields or any form of failure notifications sent towards controller.

Measurement:

Measurement MUST be done as per the equation defined in the corresponding performance test measurement section.

Reporting Format:

The Exception Handling results MUST be reported in the format of table with a column for each of the below parameters and row for each of the listed performance tests.

- Without Exceptions

- With 1% Exceptions

- With 2% Exceptions

5.3.2. Denial of Service Handling

Objective:

Determine the effect of handling DoS attacks on performance and scalability tests the impact MUST be measured for the following tests:

a. Path Provisioning Rate

b. Path Provisioning Time

c. Network Topology Change Detection Time

d. Network Discovery Size

Reference Test Setup:

The test SHOULD use one of the test setups described in section 3.1 or section 3.2 of this document.

Prerequisite:

This test MUST be performed after obtaining the baseline measurement results for the above tests.

Procedure:

1. Perform the listed tests and launch a DoS attack towards controller while the trial is running.

Note:

DoS attacks can be launched on one of the following interfaces.

a. Northbound (e.g., Query for flow entries continuously on northbound interface)
b. Management (e.g., Ping requests to controller’s management interface)
c. Southbound (e.g., TCP SYN messages on southbound interface)

Measurement:

Measurement MUST be done as per the equation defined in the corresponding test’s measurement section.

Reporting Format:

The DoS Attacks Handling results MUST be reported in the format of table with a column for each of the below parameters and row for each of the listed tests.

- Without any attacks

- With attacks

The report should also specify the nature of attack and the interface.

5.4. Reliability

5.4.1. Controller Failover Time

Objective:

The time taken to switch from an active controller to the backup controller, when the controllers work in redundancy mode and the active controller fails, defined as the interval starting with the active controller bringing down, ending with the first re-discovery message received from the new controller at its Southbound interface.

Reference Test Setup:

The test SHOULD use the test setup described in section 3.2 of this document.

Prerequisite:

1. Master controller election MUST be completed.
2. Nodes are connected to the controller cluster as per the Redundancy Mode (RM).
3. The controller cluster should have successfully completed the network topology discovery.
4. The Network Device MUST send all new flows to the controller when it receives from the test traffic generator.
5. Controller should have learned the location of destination (D1) at TP2.

Procedure:

1. Send uni-directional traffic continuously with incremental sequence number and source addresses from test traffic generator TP1 at the rate that the controller processes without any drops.
2. Ensure that there are no packet drops observed at TP2.
3. Bring down the active controller.
4. Stop the trial when a first frame received on TP2 after failover operation.
5. Record the time at which the last valid frame received (T1) at test traffic generator TP2 before sequence error and the first valid frame received (T2) after the sequence error at TP2

Measurement:

Controller Failover Time = (T2 - T1)

Packet Loss = Number of missing packet sequences.

Reporting Format:

The Controller Failover Time results MUST be tabulated with the following information.

- Number of cluster nodes
- Redundancy mode
- Controller Failover Time
- Packet Loss
- Cluster keep-alive interval

5.4.2. Network Re-Provisioning Time

Objective:

The time taken to re-route the traffic by the Controller, when there is a failure in existing traffic paths, defined as the interval starting from the first failure notification message received by the controller, ending with the last flow re-provisioning message sent by the controller at its Southbound interface.

Reference Test Setup:

This test SHOULD use one of the test setup described in section 3.1 or section 3.2 of this document.

Prerequisite:

1. Network with the given number of nodes and redundant paths MUST be deployed.
2. Ensure that the controller MUST have knowledge about the location of test traffic generators TP1 and TP2.

3. Ensure that the controller does not pre-provision the alternate path in the emulated Network Devices at the forwarding plane test emulator.

Procedure:

1. Send bi-directional traffic continuously with unique sequence number from TP1 and TP2.
2. Bring down a link or switch in the traffic path.
3. Stop the trial after receiving first frame after network re-convergence.
4. Record the time of last received frame prior to the frame loss at TP2 (TP2-Tlfr) and the time of first frame received after the frame loss at TP2 (TP2-Tffr). There must be a gap in sequence numbers of these frames
5. Record the time of last received frame prior to the frame loss at TP1 (TP1-Tlfr) and the time of first frame received after the frame loss at TP1 (TP1-Tffr).

Measurement:

Forward Direction Path Re-Provisioning Time (FDRT)
= (TP2-Tffr - TP2-Tlfr)

Reverse Direction Path Re-Provisioning Time (RDRT)
= (TP1-Tffr - TP1-Tlfr)

Network Re-Provisioning Time = (FDRT+RDRT)/2

Forward Direction Packet Loss = Number of missing sequence frames at TP1

Reverse Direction Packet Loss = Number of missing sequence frames at TP2

Reporting Format:

The Network Re-Provisioning Time results MUST be tabulated with the following information.

- Number of nodes in the primary path

- Number of nodes in the alternate path

- Network Re-Provisioning Time

- Forward Direction Packet Loss

- Reverse Direction Packet Loss

6. References

6.1. Normative References

[RFC2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC8174] B. Leiba, "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.

[I-D.sdn-controller-benchmark-term] Bhuvaneswaran.V, Anton Basil, Mark.T, Vishwas Manral, Sarah Banks, "Terminology for Benchmarking SDN Controller Performance", draft-ietf-bmwg-sdn-controller-benchmark-term-10 (Work in progress), May 25, 2018

6.2. Informative References

[OpenFlow Switch Specification] ONF,"OpenFlow Switch Specification" Version 1.4.0 (Wire Protocol 0x05), October 14, 2013.

7. IANA Considerations

This document does not have any IANA requests.

8. Security Considerations

Benchmarking tests described in this document are limited to the performance characterization of controllers in a lab environment with isolated network.

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the controller.

Special capabilities SHOULD NOT exist in the controller specifically for benchmarking purposes. Any implications for network security arising from the controller SHOULD be identical in the lab and in production networks.

9. Acknowledgments

The authors would like to thank the following individuals for providing their valuable comments to the earlier versions of this document: Al Morton (AT&T), Sandeep Gangadharan (HP), M. Georgescu (NAIST), Andrew McGregor (Google), Scott Bradner , Jay Karthik (Cisco), Ramakrishnan (Dell), Khasanov Boris (Huawei), Brian Castelli (Spirent)

This document was prepared using 2-Word-v2.0.template.dot.

Appendix A Benchmarking Methodology using OpenFlow Controllers

This section gives an overview of OpenFlow protocol and provides test methodology to benchmark SDN controllers supporting OpenFlow southbound protocol. OpenFlow protocol is used as an example to illustrate the methodologies defined in this document.

A.1. Protocol Overview

OpenFlow is an open standard protocol defined by Open Networking Foundation (ONF)[OpenFlow Switch Specification], used for programming the forwarding plane of network switches or routers via a centralized controller.

A.2. Messages Overview

OpenFlow protocol supports three messages types namely controller-to-switch, asynchronous and symmetric.

Controller-to-switch messages are initiated by the controller and used to directly manage or inspect the state of the switch. These messages allow controllers to query/configure the switch (Features, Configuration messages), collect information from switch (Read-State message), send packets on specified port of switch (Packet-out message), and modify switch forwarding plane and state (Modify-State, Role-Request messages etc.).

Asynchronous messages are generated by the switch without a controller soliciting them. These messages allow switches to update controllers to denote an arrival of new flow (Packet-in), switch state change (Flow-Removed, Port-status) and error (Error).

Symmetric messages are generated in either direction without solicitation. These messages allow switches and controllers to set up connection (Hello), verify for liveness (Echo) and offer additional functionalities (Experimenter).

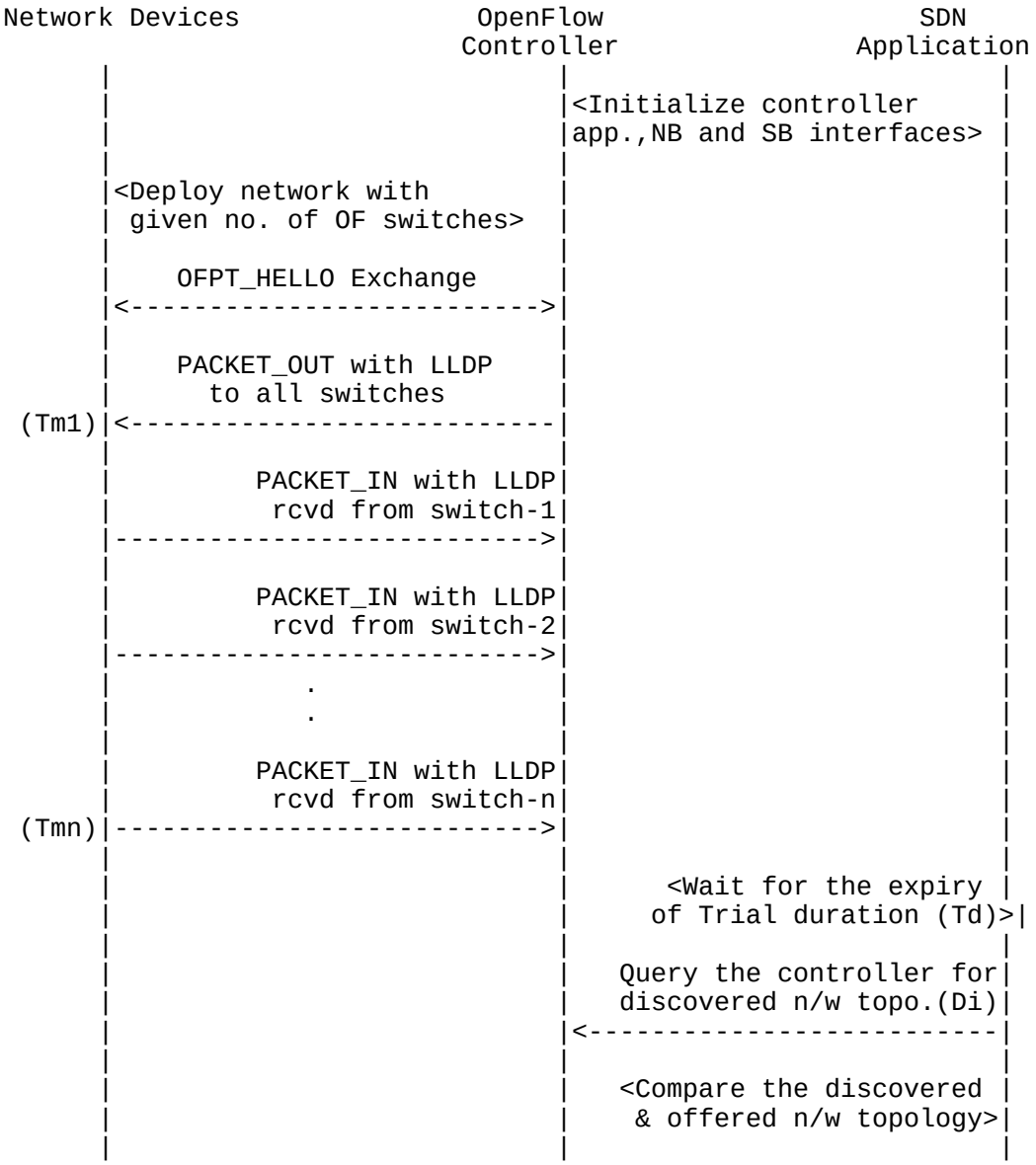
A.3. Connection Overview

OpenFlow channel is used to exchange OpenFlow message between an OpenFlow switch and an OpenFlow controller. The OpenFlow channel connection can be setup using plain TCP or TLS. By default, a switch establishes single connection with SDN controller. A switch may establish multiple parallel connections to single controller (auxiliary connection) or multiple controllers to handle controller failures and load balancing.

A.4. Performance Benchmarking Tests

A.4.1. Network Topology Discovery Time

Procedure:



Legend:

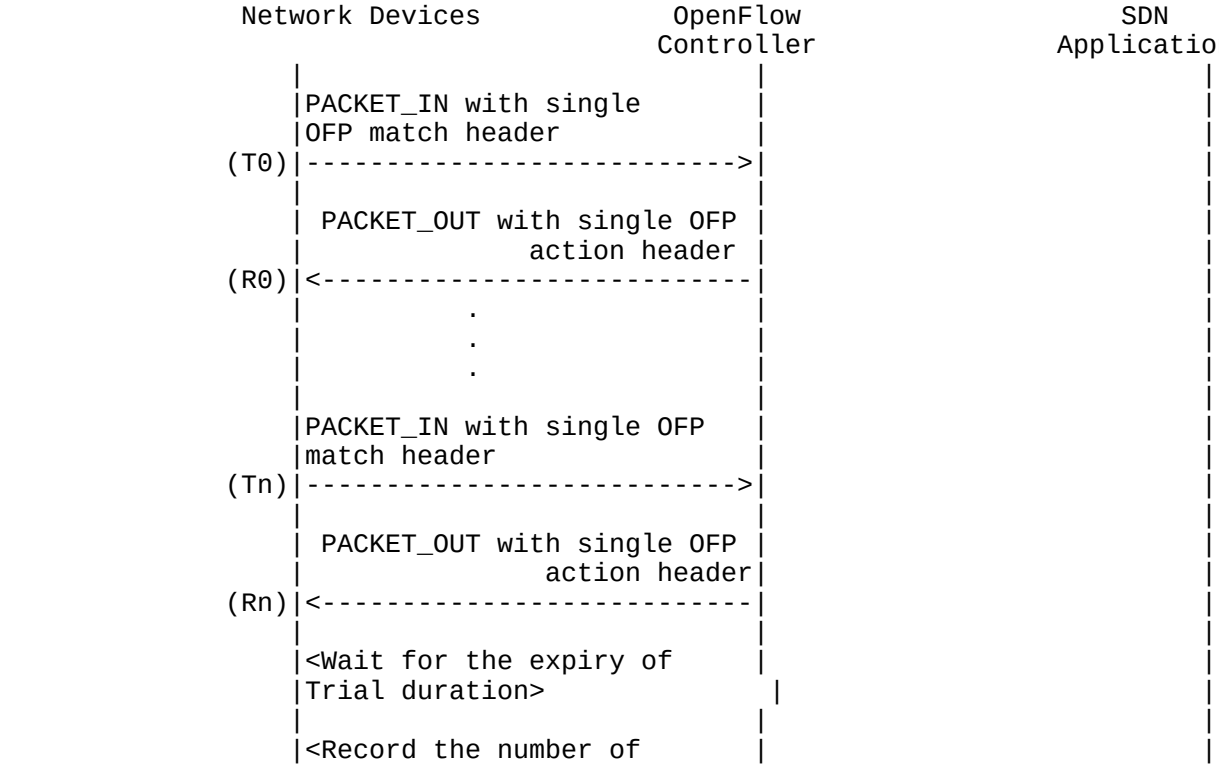
NB: Northbound
SB: Southbound
OF: OpenFlow
Tm1: Time of reception of first LLDP message from controller
Tmn: Time of last LLDP message sent to controller

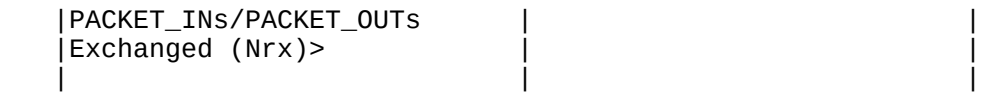
Discussion:

The Network Topology Discovery Time can be obtained by calculating the time difference between the first PACKET_OUT with LLDP message received from the controller (Tm1) and the last PACKET_IN with LLDP message sent to the controller (Tmn) when the comparison is successful.

A.4.2. Asynchronous Message Processing Time

Procedure:





Legend:

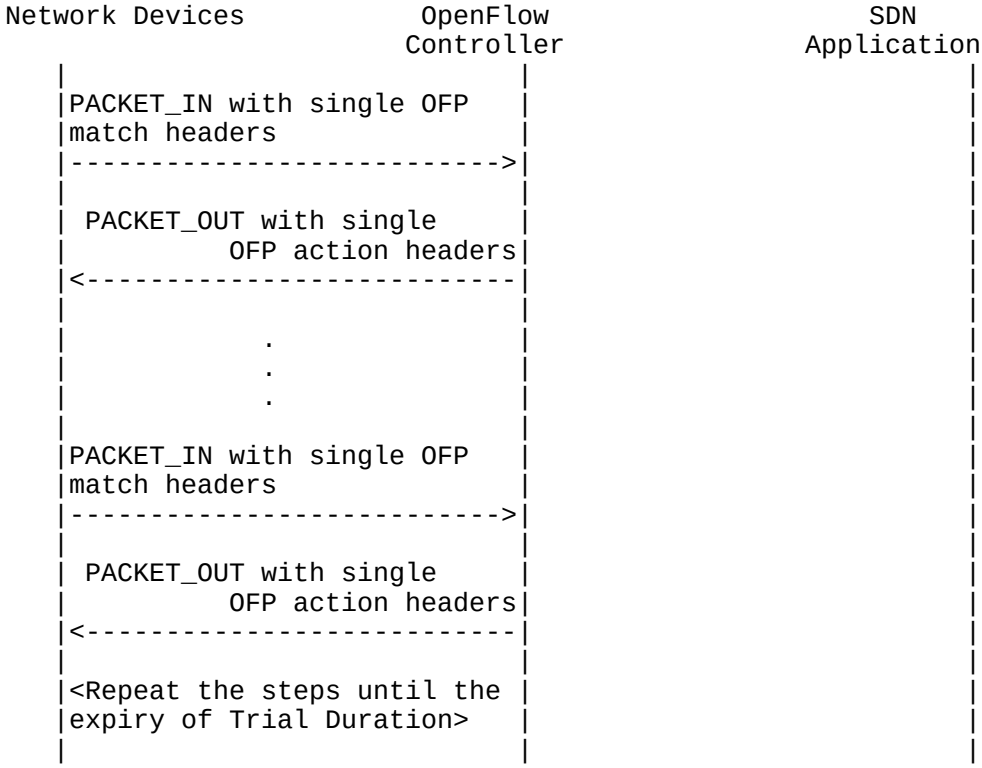
T0,T1, ..Tn are PACKET_IN messages transmit timestamps.
R0,R1, ..Rn are PACKET_OUT messages receive timestamps.
Nrx : Number of successful PACKET_IN/PACKET_OUT message exchanges

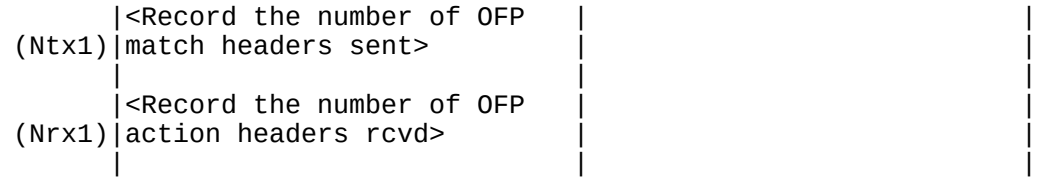
Discussion:

The Asynchronous Message Processing Time will be obtained by sum of ((R0-T0),(R1-T1)..(Rn - Tn))/ Nrx.

A.4.3. Asynchronous Message Processing Rate

Procedure:





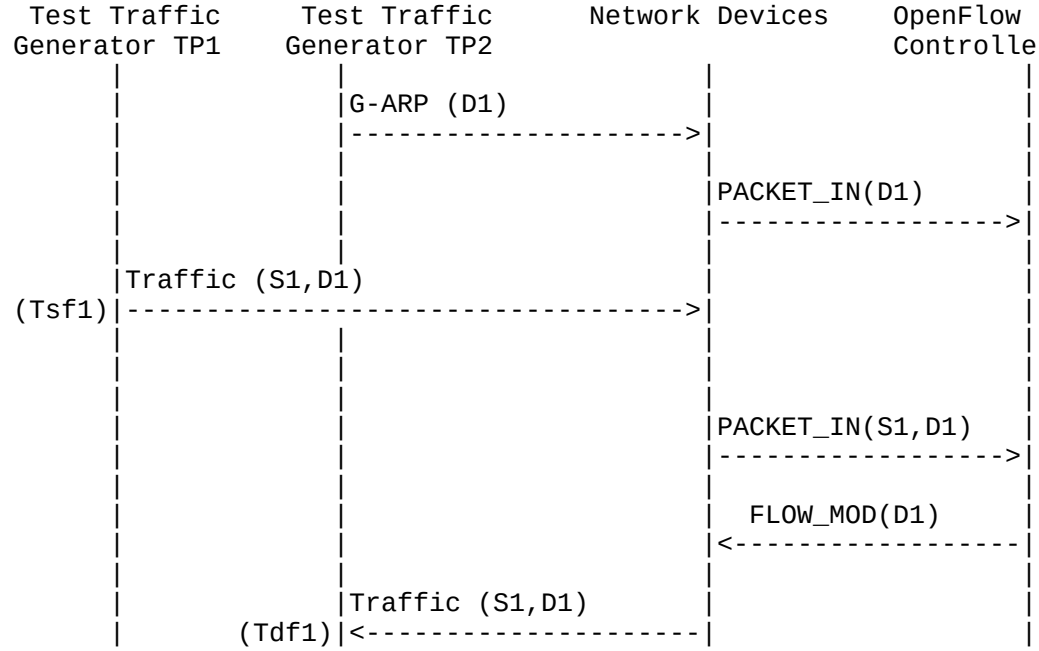
Note: The Ntx1 on initial trials should be greater than Nrx1 and repeat the trials until the Nrxn for two consecutive trials equeal to (+/-P%).

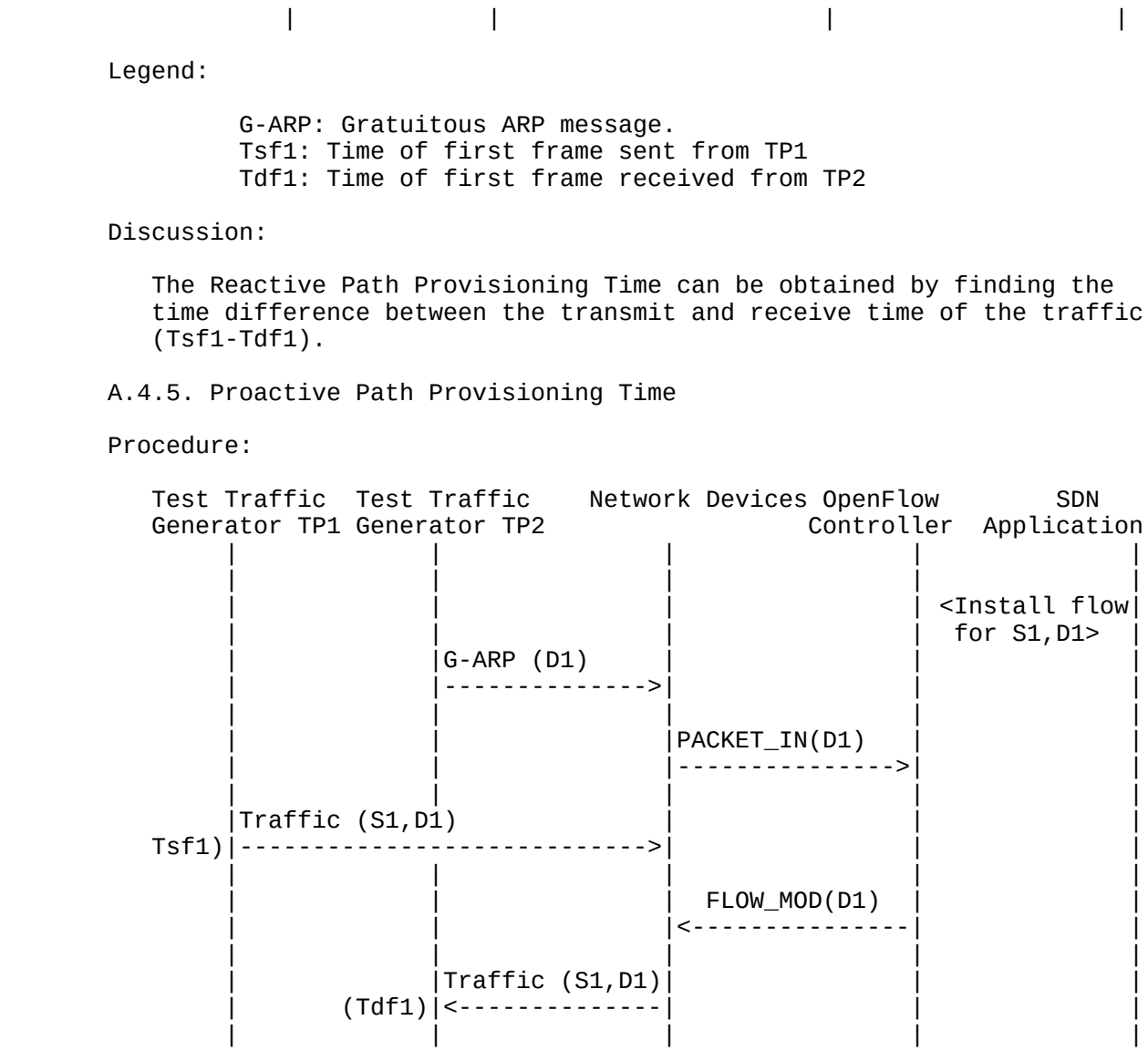
Discussion:

This test will measure two benchmarks using single procedure. 1) The Maximum Asynchronous Message Processing Rate will be obtained by calculating the maximum PACKET OUTs (Nrxn) received from the controller(s) across n trials. 2) The Loss-free Asynchronous Message Processing Rate will be obtained by calculating the maximum PACKET OUTs received from controller (s) when Loss Ratio equals zero. The loss ratio is obtained by 1 - Nrxn/Ntxn

A.4.4. Reactive Path Provisioning Time

Procedure:





Legend:

G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

Discussion:

The Reactive Path Provisioning Time can be obtained by finding the time difference between the transmit and receive time of the traffic (Tsf1-Tdf1).

A.4.5. Proactive Path Provisioning Time

Procedure:

Legend:

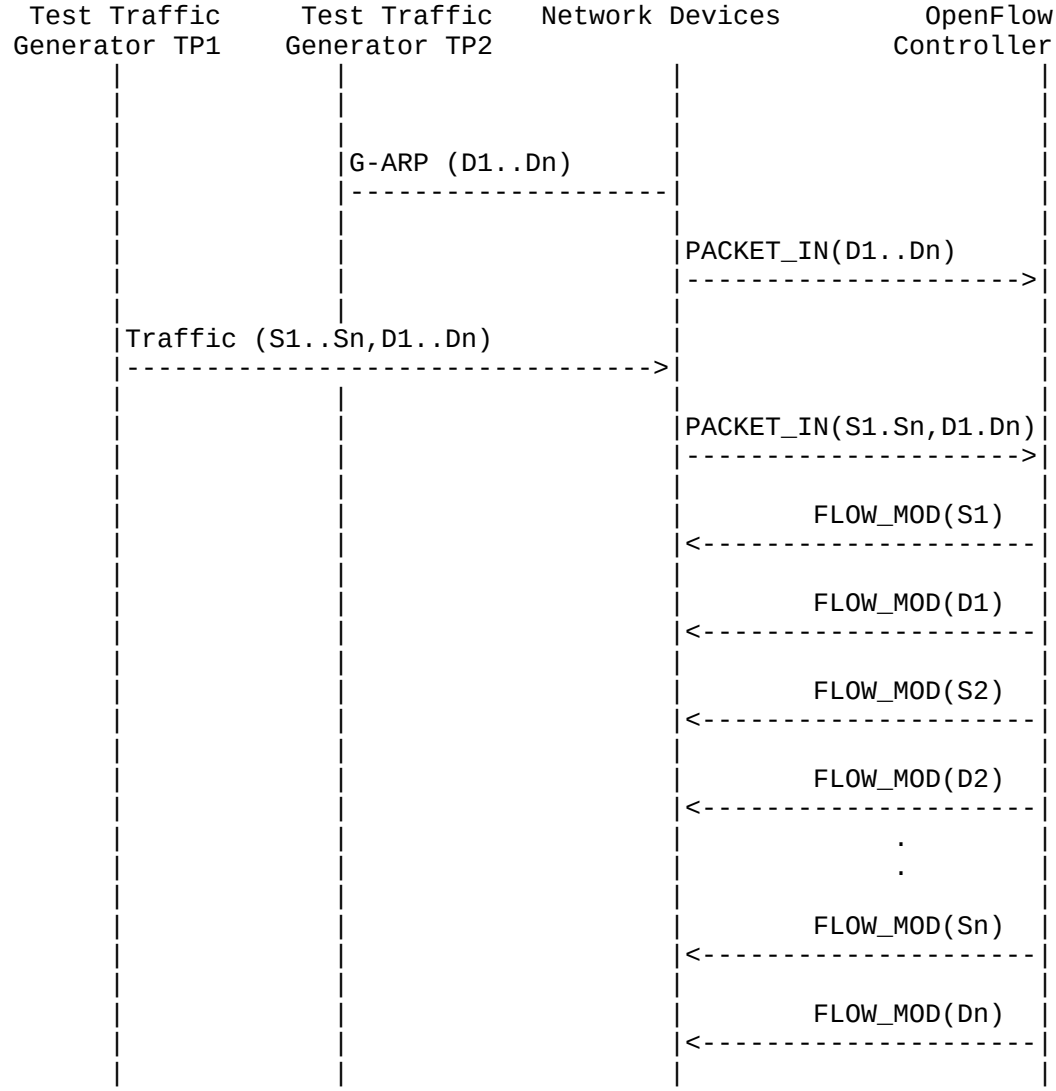
G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

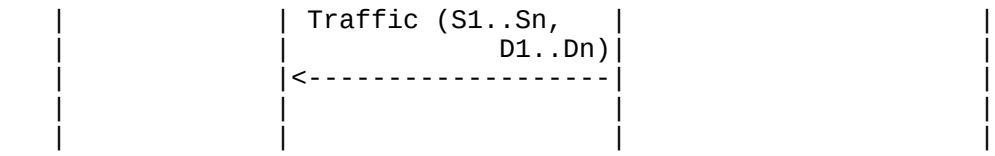
Discussion:

The Proactive Path Provisioning Time can be obtained by finding the time difference between the transmit and receive time of the traffic (Tsf1-Tdf1).

A.4.6. Reactive Path Provisioning Rate

Procedure:





Legend:

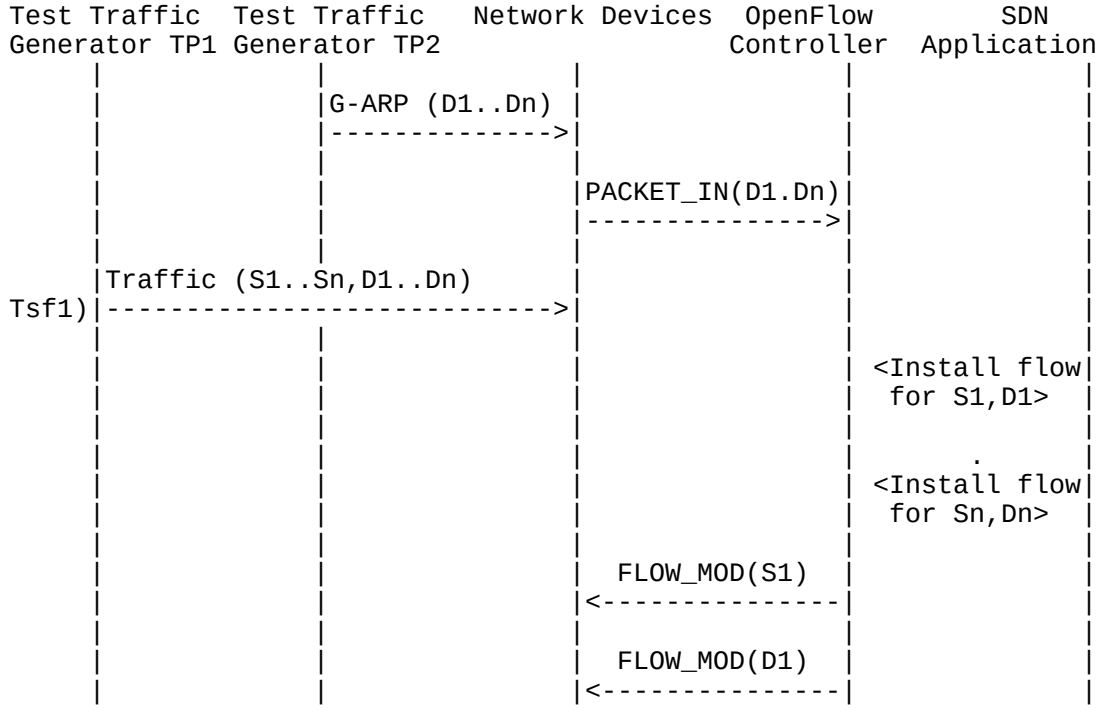
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ....
Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

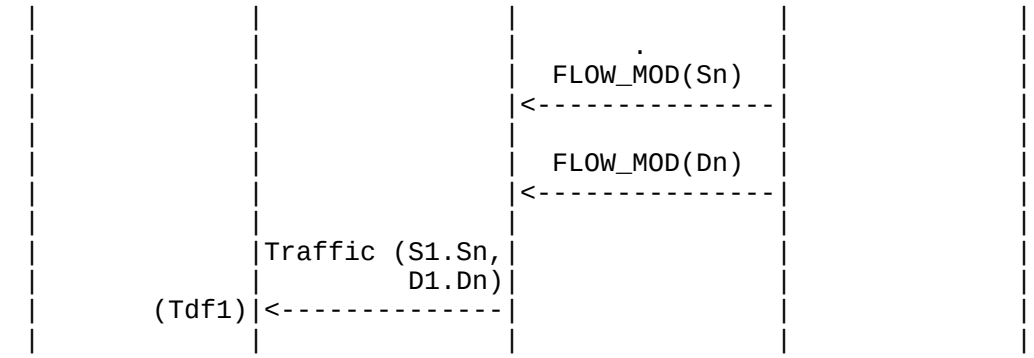
Discussion:

The Reactive Path Provisioning Rate can be obtained by finding the total number of frames received at TP2 after the trial duration.

A.4.7. Proactive Path Provisioning Rate

Procedure:





Legend:

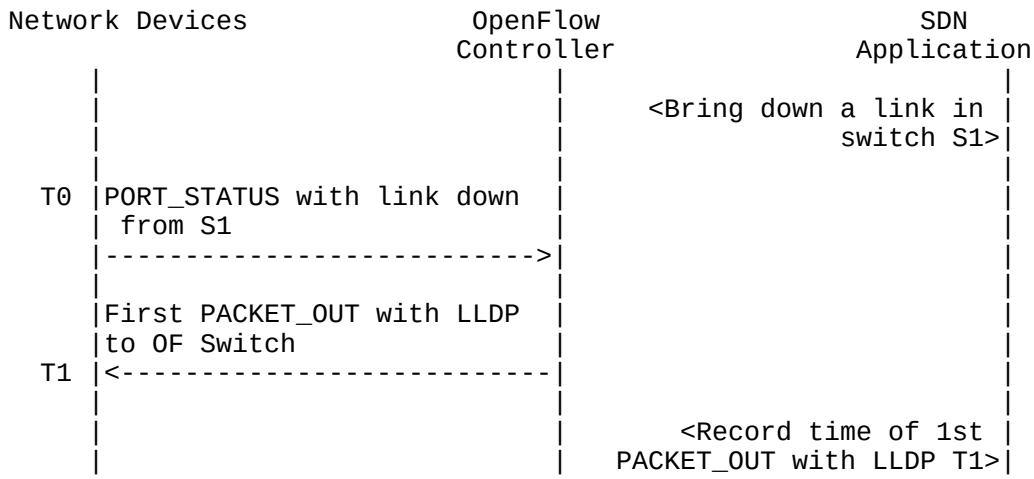
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ....
Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

Discussion:

The Proactive Path Provisioning Rate can be obtained by finding the total number of frames received at TP2 after the trial duration

A.4.8. Network Topology Change Detection Time

Procedure:



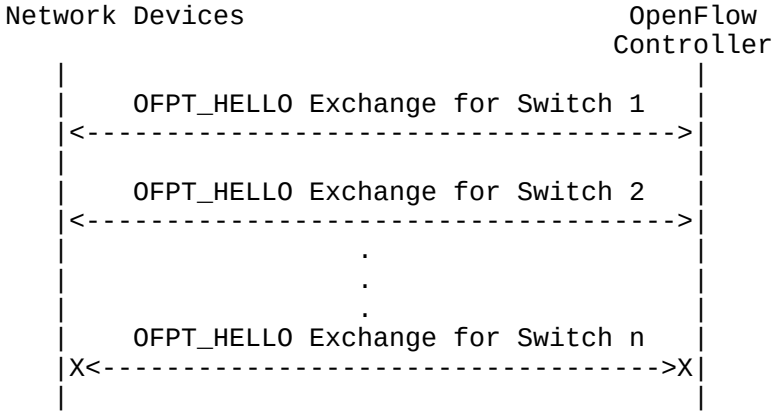
Discussion:

The Network Topology Change Detection Time can be obtained by finding the difference between the time the OpenFlow switch S1 sends the PORT_STATUS message (T0) and the time that the OpenFlow controller sends the first topology re-discovery message (T1) to OpenFlow switches.

A.5. Scalability

A.5.1. Control Sessions Capacity

Procedure:

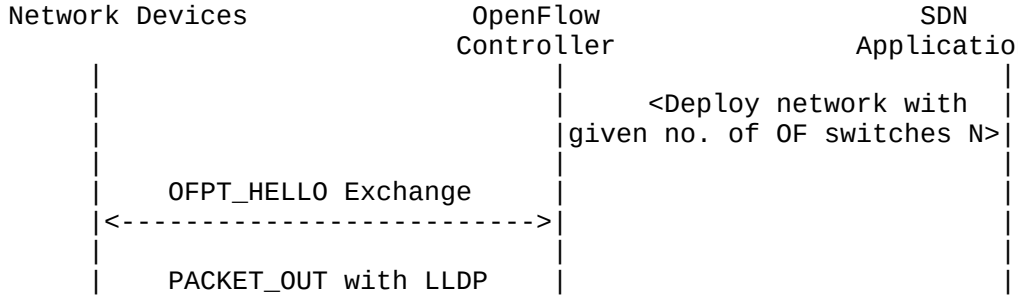


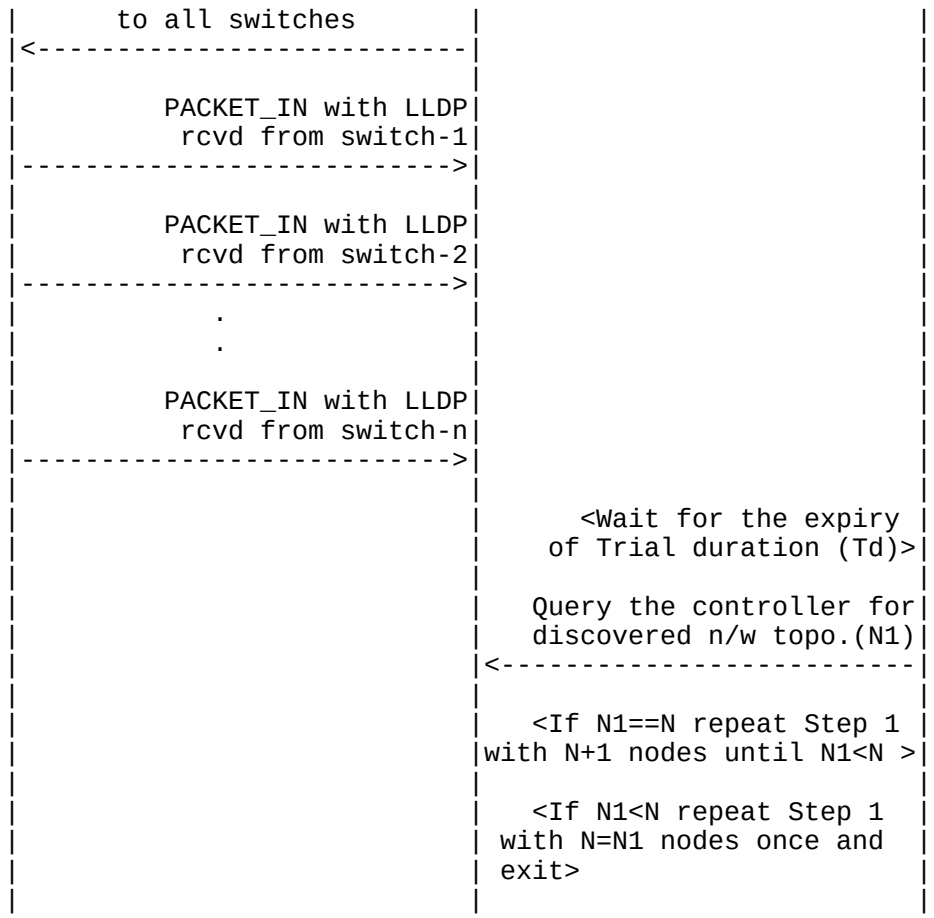
Discussion:

The value of Switch n-1 will provide Control Sessions Capacity.

A.5.2. Network Discovery Size

Procedure:





Legend:

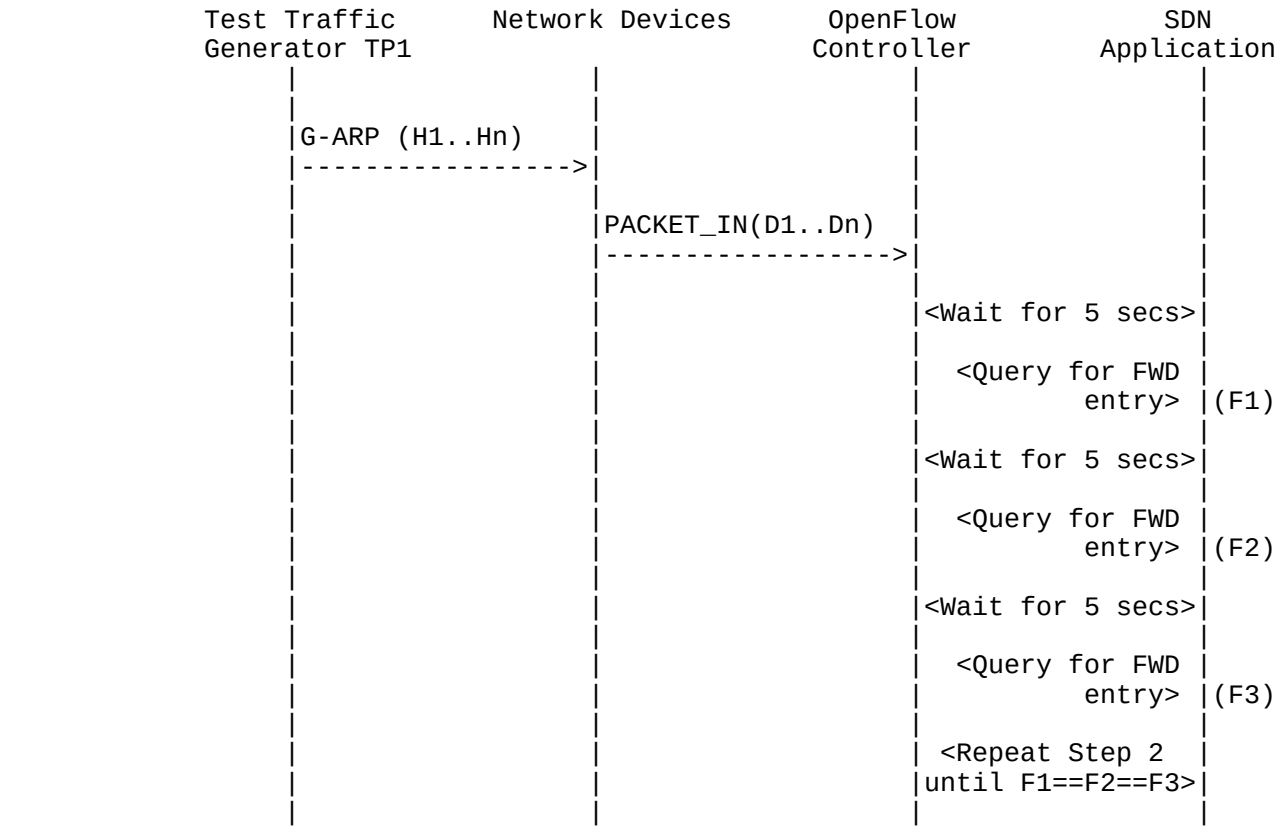
n/w topo: Network Topology
OF: OpenFlow

Discussion:

The value of N1 provides the Network Discovery Size value. The trial duration can be set to the stipulated time within which the user expects the controller to complete the discovery process.

A.5.3. Forwarding Table Capacity

Procedure:



Legend:

G-ARP: Gratuitous ARP
H1..Hn: Host 1 .. Host n
FWD: Forwarding Table

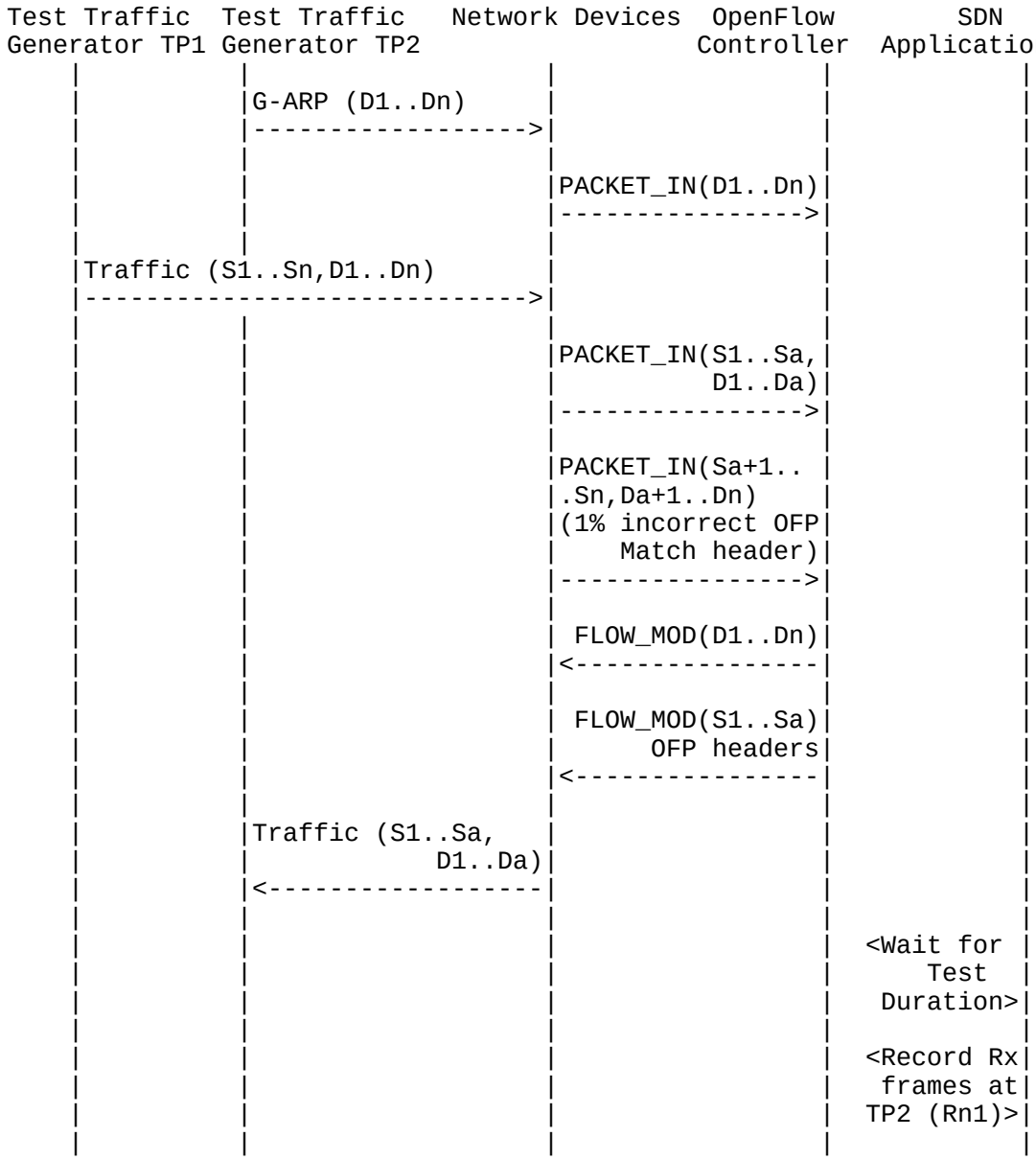
Discussion:

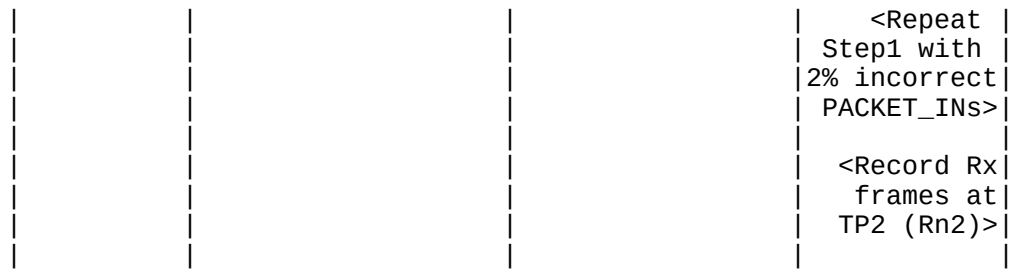
Query the controller forwarding table entries for multiple times until the three consecutive queries return the same value. The last value retrieved from the controller will provide the Forwarding Table Capacity value. The query interval is user configurable. The 5 seconds shown in this example is for representational purpose.

A.6. Security

A.6.1. Exception Handling

Procedure:





Legend:

G-ARP: Gratuitous ARP
PACKET_IN(Sa+1..Sn,Da+1..Dn): OpenFlow PACKET_IN with wrong version number
Rn1: Total number of frames received at Test Port 2 with 1% incorrect frames
Rn2: Total number of frames received at Test Port 2 with 2% incorrect frames

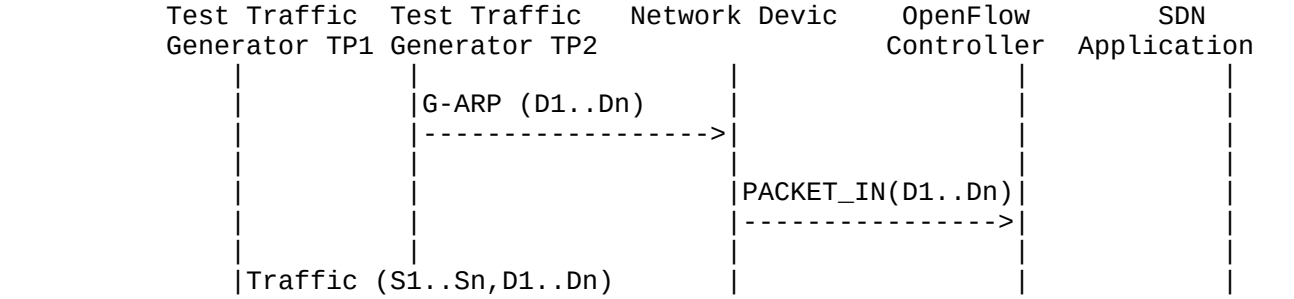
Discussion:

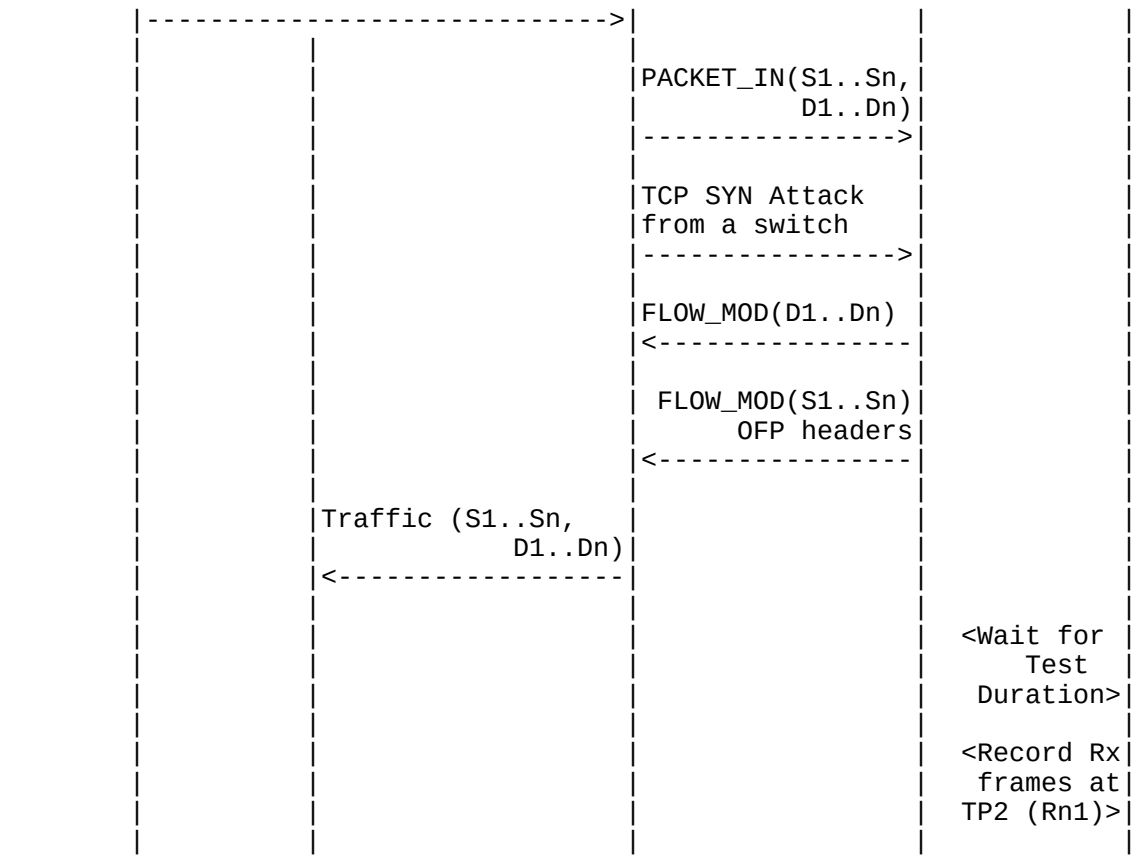
The traffic rate sent towards OpenFlow switch from Test Port 1 should be 1% higher than the Path Programming Rate. Rn1 will provide the Path Provisioning Rate of controller at 1% of incorrect frames handling and Rn2 will provide the Path Provisioning Rate of controller at 2% of incorrect frames handling.

The procedure defined above provides test steps to determine the effect of handling error packets on Path Programming Rate. Same procedure can be adopted to determine the effects on other performance tests listed in this benchmarking tests.

A.6.2. Denial of Service Handling

Procedure:





Legend:

G-ARP: Gratuitous ARP

Discussion:

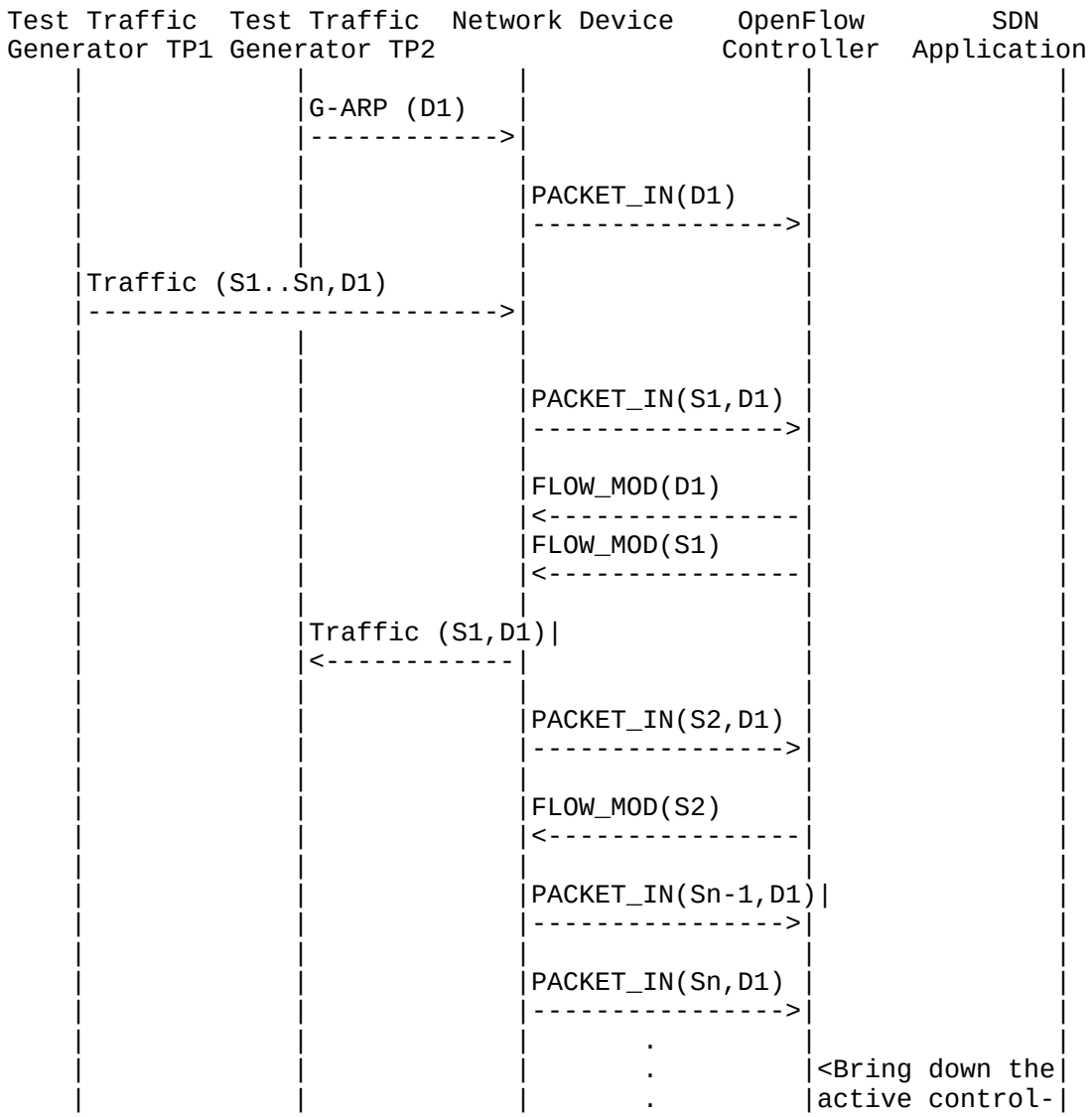
TCP SYN attack should be launched from one of the emulated/simulated OpenFlow Switch. Rn1 provides the Path Programming Rate of controller uponhandling denial of service attack.

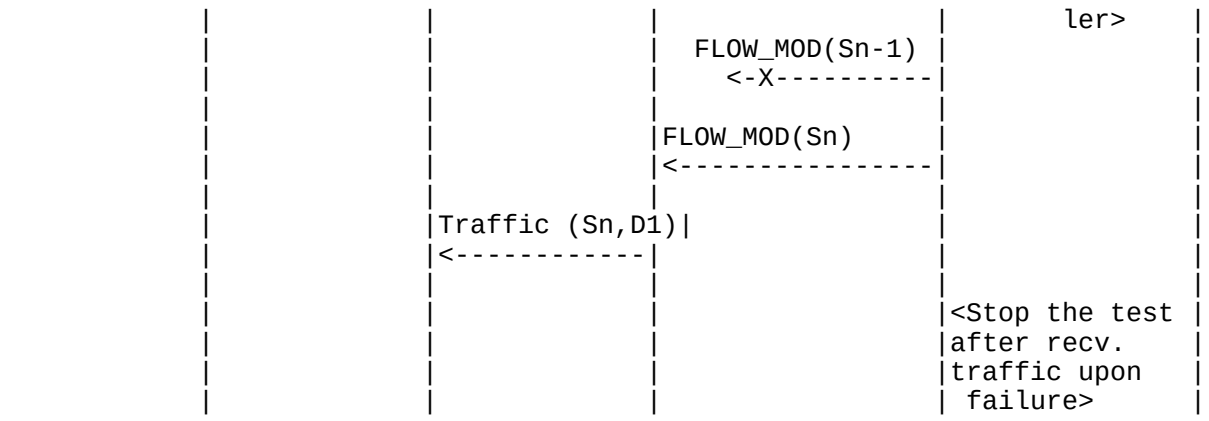
The procedure defined above provides test steps to determine the effect of handling denial of service on Path Programming Rate. Same procedure can be adopted to determine the effects on other performance tests listed in this benchmarking tests.

A.7. Reliability

A.7.1. Controller Failover Time

Procedure:





Legend:

G-ARP: Gratuitous ARP.

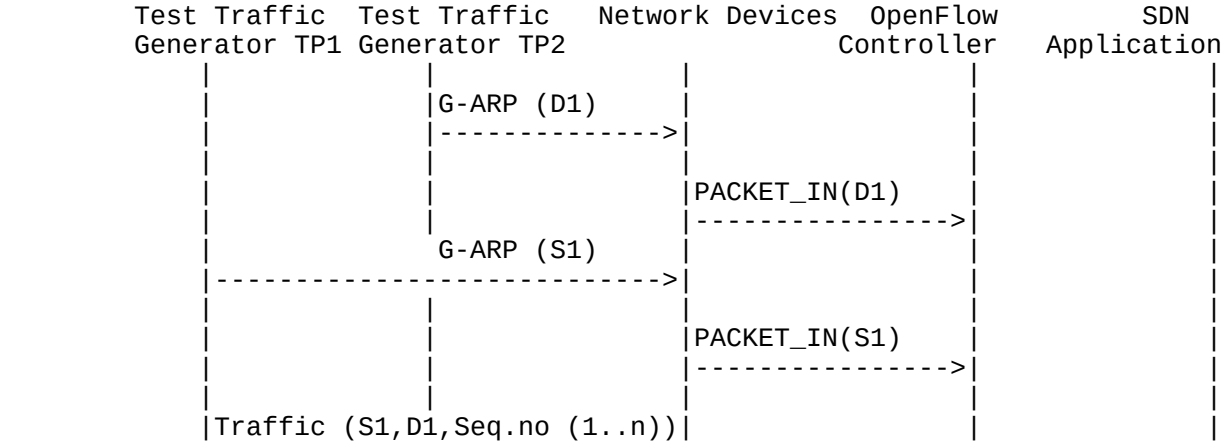
Discussion:

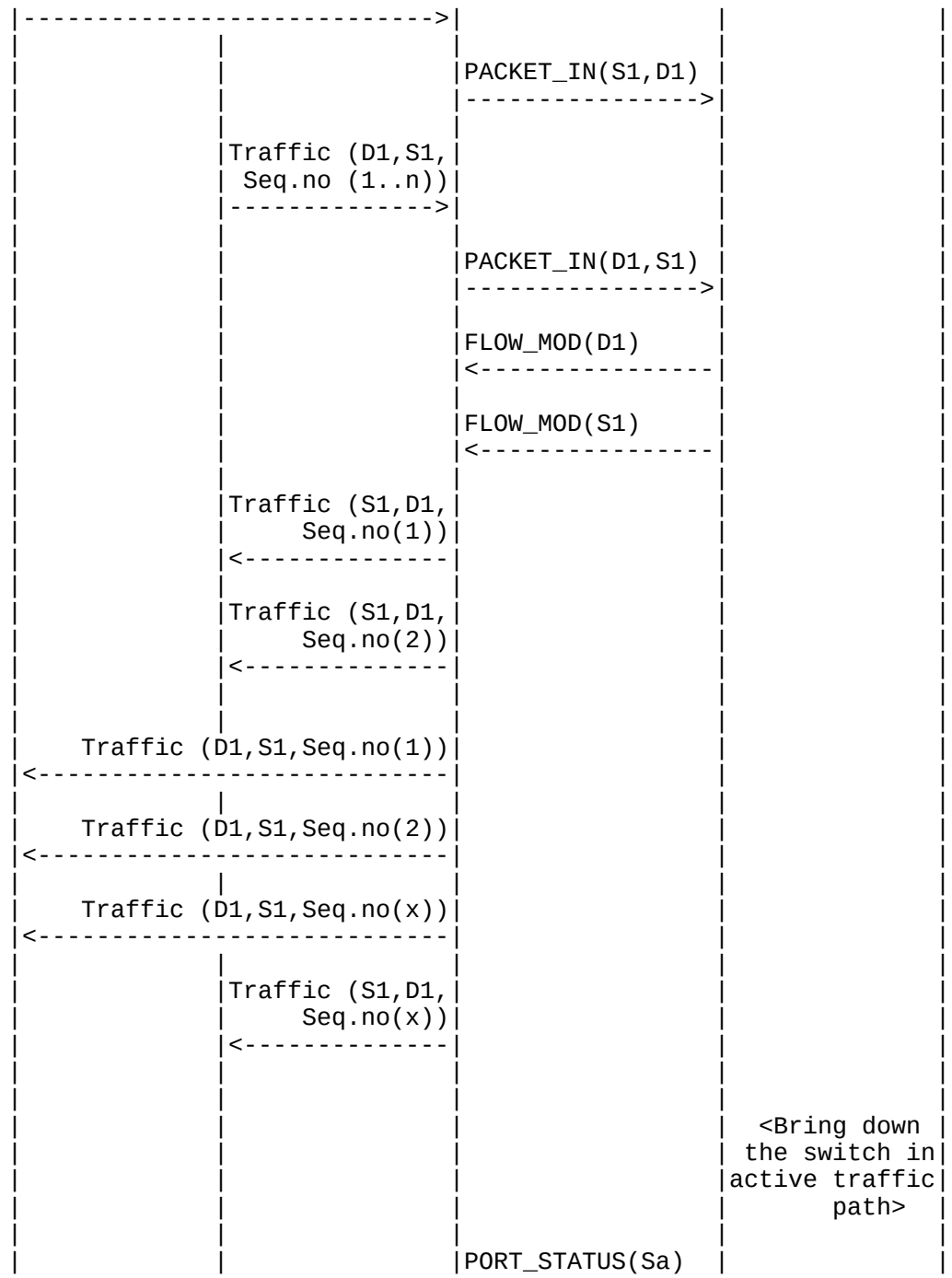
The time difference between the last valid frame received before the traffic loss and the first frame received after the traffic loss will provide the controller failover time.

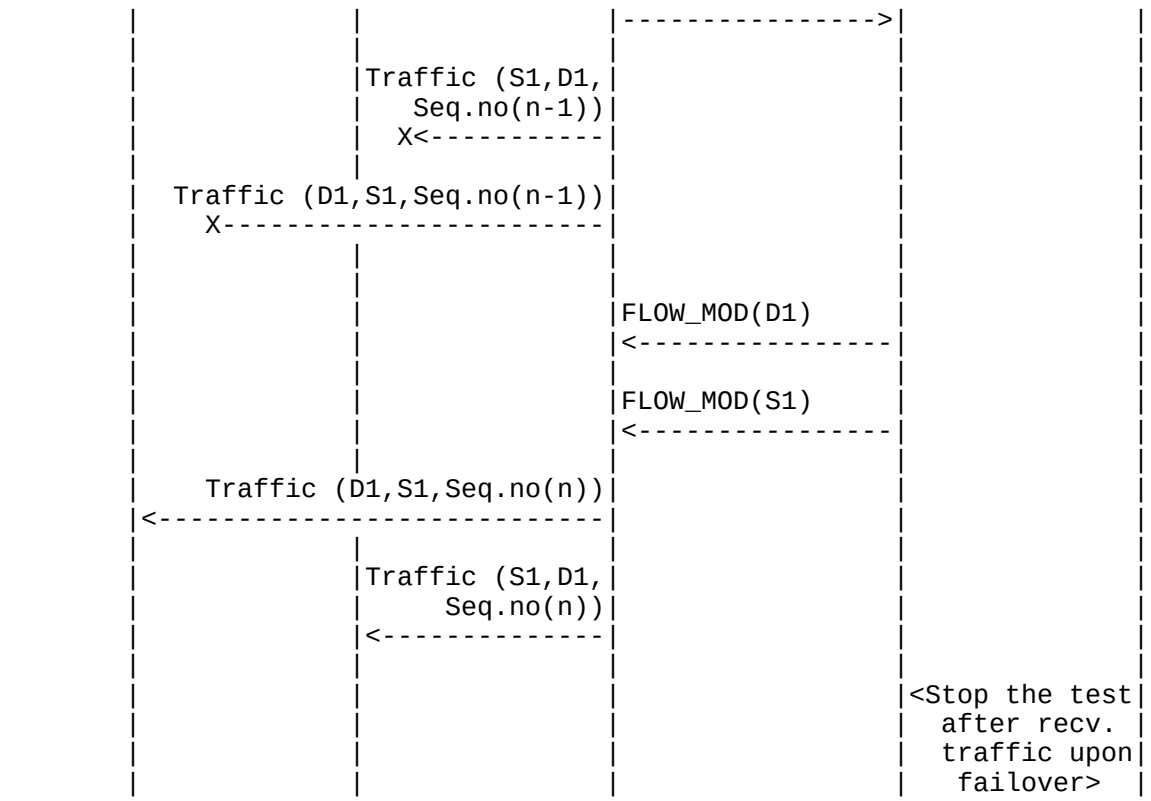
If there is no frame loss during controller failover time, the controller failover time can be deemed negligible.

A.7.2. Network Re-Provisioning Time

Procedure:




PACKET_IN(S1,D1)
Traffic (D1,S1,
Seq.no (1..n))
PACKET_IN(D1,S1)
FLOW_MOD(D1)
FLOW_MOD(S1)
Traffic (S1,D1,
Seq.no(1))
Traffic (S1,D1,
Seq.no(2))
Traffic (D1,S1,Seq.no(1))
Traffic (D1,S1,Seq.no(2))
Traffic (D1,S1,Seq.no(x))
Traffic (S1,D1,
Seq.no(x))
<Bring down
the switch in
active traffic
path>
PORT_STATUS(Sa)



Legend:

G-ARP: Gratuitous ARP message.
Seq.no: Sequence number.
Sa: Neighbor switch of the switch that was brought down.

Discussion:

The time difference between the last valid frame received before the traffic loss (Packet number with sequence number x) and the first frame received after the traffic loss (packet with sequence number n) will provide the network path re-provisioning time.

Note that the trial is valid only when the controller provisions the alternate path upon network failure.

Authors' Addresses

Bhuvaneswaran Vengainathan
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: bhuvaneswaran.vengainathan@veryxtech.com

Anton Basil
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: anton.basil@veryxtech.com

Mark Tassinari
Hewlett-Packard,
8000 Foothills Blvd,
Roseville, CA 95747

Email: mark.tassinari@hpe.com

Vishwas Manral
Nano Sec,
CA

Email: vishwas.manral@gmail.com

Sarah Banks
VSS Monitoring
930 De Guigne Drive,
Sunnyvale, CA

Email: sbanks@encrypted.net

Internet-Draft
Network Working Group
Intended Status: Informational
Expires: April 01, 2018

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Mark Tassinari
Hewlett-Packard
Vishwas Manral
Nano Sec
Sarah Banks
VSS Monitoring
October 01, 2017

Terminology for Benchmarking SDN Controller Performance
draft-ietf-bmwg-sdn-controller-benchmark-term-05

Abstract

This document defines terminology for benchmarking an SDN controller's control plane performance. It extends the terminology already defined in RFC 7426 for the purpose of benchmarking SDN controllers. The terms provided in this document help to benchmark SDN controller's performance independent of the controller's supported protocols and/or network services. A mechanism for benchmarking the performance of SDN controllers is defined in the companion methodology document. These two documents provide a standard mechanism to measure and evaluate the performance of various controller implementations.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress.

This Internet-Draft will expire on April 01, 2018.

Table of Contents

1. Introduction

Software Defined Networking (SDN) is a networking architecture in which network control is decoupled from the underlying forwarding function and is placed in a centralized location called the SDN controller. The SDN controller abstracts the underlying network and offers a global view of the overall network to applications and business logic. Thus, an SDN controller provides the flexibility to program, control, and manage network behaviour dynamically through standard interfaces. Since the network controls are logically centralized, the need to benchmark the SDN controller performance becomes significant. This document defines terms to benchmark various controller designs for performance, scalability, reliability and security, independent of northbound and southbound protocols.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2. Term Definitions

2.1. SDN Terms

The terms defined in this section are extensions to the terms defined in [RFC7426] "Software-Defined Networking (SDN): Layers and Architecture Terminology". This RFC should be referred before attempting to make use of this document.

2.1.1. Flow

Definition:
The definition of Flow is same as microflows defined in [RFC4689] Section 3.1.5.

Discussion:
A flow can be set of packets having same source address, destination address, source port and destination port, or any of these combinations.

Measurement Units:
N/A

See Also:
None

2.1.2. Northbound Interface

Definition:
The definition of northbound interface is same Service Interface defined in [RFC7426] .

Discussion:
The northbound interface allows SDN applications and orchestration systems to program and retrieve the network information through the SDN controller.

Measurement Units:
N/A

See Also:
None

2.1.3. Controller Forwarding Table

Definition:
A controller forwarding table contains flow entries learned in one of two ways: first, entries could be learned from traffic received through the data plane, or, second, these entries could be statically provisioned on the controller, and distributed to devices via the southbound interface.

Discussion:
The controller forwarding table has an aging mechanism which will be applied only for dynamically learnt entries.

Measurement Units:
N/A

See Also:
None

2.1.4. Proactive Flow Provisioning Mode

Definition:
Controller programming flows in Network Devices based on the flow entries provisioned through controller’s northbound interface.

Discussion:
Orchestration systems and SDN applications can define the network forwarding behaviour by programming the controller using proactive flow provisioning. The controller can then program the Network Devices with the pre-provisioned entries.

Measurement Units:
N/A

See Also:
None

2.1.5. Reactive Flow Provisioning Mode

Definition:
Controller programming flows in Network Devices based on the traffic received from Network Devices through controller’s southbound interface

Discussion:
The SDN controller dynamically decides the forwarding behaviour based on the incoming traffic from the Network Devices. The controller then programs the Network Devices using Reactive Flow Provisioning.

Measurement Units:
N/A

See Also:
None

2.1.6. Path

Definition:
Refer to Section 5 in [RFC2330]

Discussion:
None

Measurement Units:
N/A

See Also:
None

2.1.7. Standalone Mode

Definition:

Single controller handling all control plane functionalities without redundancy, or the ability to provide high availability and/or automatic failover.

Discussion:
In standalone mode, one controller manages one or more network domains.

Measurement Units:
N/A

See Also:
None

2.1.8. Cluster/Redundancy Mode

Definition:
A group of 2 or more controllers handling all control plane functionalities.

Discussion:
In cluster mode, multiple controllers are teamed together for the purpose of load sharing and/or high availability. The controllers in the group may work in active/standby (master/slave) or active/active (equal) mode depending on the intended purpose.

Measurement Units:
N/A

See Also:
None

2.1.9. Asynchronous Message

Definition:
Any message from the Network Device that is generated for network events.

Discussion:
Control messages like flow setup request and response message is classified as asynchronous message. The controller has to return a response message. Note that the Network Device will not be in blocking mode and continues to send/receive other control messages

Measurement Units:

N/A

See Also:
None

2.1.10. Test Traffic Generator

Definition:
Test Traffic Generator is an entity that generates/receives network traffic.

Discussion:
Test Traffic Generator can be an entity that interfaces with Network Devices to send/receive real-time network traffic.

Measurement Units:
N/A

See Also:
None

2.2. Test Configuration/Setup Terms

2.2.1. Number of Network Devices

Definition:
The number of Network Devices present in the defined test topology.

Discussion:
The Network Devices defined in the test topology can be deployed using real hardware or emulated in hardware platforms.

Measurement Units:
N/A

See Also:
None

2.2.2. Trails

Definition:
The number of times the test needs to be repeated.

Discussion:

The test needs to be repeated for multiple iterations to obtain a reliable metric. It is recommended that this test SHOULD be performed for at least 10 iterations to increase the confidence in measured result.

Measurement Units:
N/A

See Also:
None

2.2.3. Trail Duration

Definition:
Defines the duration of test trails for each iteration.

Discussion:
Trail duration forms the basis for stop criteria for benchmarking tests. Trail not completed within this time interval is considered as incomplete.

Measurement Units:
seconds

See Also:
None

2.2.4. Number of Cluster nodes

Definition:
Defines the number of controllers present in the controller cluster.

Discussion:
This parameter is relevant when testing the controller performance in clustering/teaming mode. The number of nodes in the cluster MUST be greater than 1.

Measurement Units:
N/A

See Also:
None

2.3. Benchmarking Terms

This section defines metrics for benchmarking the SDN controller. The procedure to perform the defined metrics is defined in the accompanying methodology document [I-D.sdn-controller-benchmark-meth]

2.3.1. Performance

2.3.1.1. Network Topology Discovery Time

Definition:
The time taken by controller(s) to determine the complete network topology, defined as the interval starting with the first discovery message from the controller(s) at its Southbound interface, ending with all features of the static topology determined.

Discussion:
Network topology discovery is key for the SDN controller to provision and manage the network. So it is important to measure how quickly the controller discovers the topology to learn the current network state. This benchmark is obtained by presenting a network topology (Tree, Mesh or Linear) with the given number of nodes to the controller and wait for the discovery process to complete .It is expected that the controller supports network discovery mechanism and uses protocol messages for its discovery process.

Measurement Units:
milliseconds

See Also:
None

2.3.1.2. Asynchronous Message Processing Time

Definition:
The time taken by controller(s) to process an asynchronous message, defined as the interval starting with an asynchronous message from a network device after the discovery of all the devices by the controller(s), ending with a response message from the controller(s) at its Southbound interface.

Discussion:
For SDN to support dynamic network provisioning, it is important to measure how quickly the controller responds to an event triggered

from the network. The event could be any notification messages generated by an Network Device upon arrival of a new flow, link down etc. This benchmark is obtained by sending asynchronous messages from every connected Network Devices one at a time for the defined trail duration. This test assumes that the controller will respond to the received asynchronous message.

Measurement Units:
milliseconds

See Also:
None

2.3.1.3. Asynchronous Message Processing Rate

Definition:
The number responses to asynchronous messages (such as new flow arrival notification message, etc.) for which the controller(s) performed processing and replied with a valid and productive (non-trivial) response message.

Discussion:
As SDN assures flexible network and agile provisioning, it is important to measure how many network events that the controller can handle at a time. This benchmark is obtained by sending asynchronous messages from every connected Network Devices at the rate that the controller processes without dropping. This test assumes that the controller responds to all the received asynchronous messages (the messages can be designed to elicit individual responses).

When sending asynchronous messages to the controller(s) at high rates, some messages or responses may be discarded or corrupted and require retransmission to controller(s). Therefore, a useful qualification on Asynchronous Message Processing Rate is whether the in-coming message count equals the response count in each trial. This is called the Loss-free Asynchronous Message Processing Rate.

Note that several of the early controller benchmarking tools did not consider lost messages, and instead report the maximum response rate. This is called the Maximum Asynchronous Message Processing Rate.

To characterize both the Loss-free and Maximum Rates, a test could begin the first trial by sending asynchronous messages to the controller (s) at the maximum possible rate and record the message

reply rate and the message loss rate. The message sending rate is then decreased by the step-size. The message reply rate and the message loss rate are recorded. The test ends with a trial where the controller(s) processes the all asynchronous messages sent without loss. This is the Loss-free Asynchronous Message Processing Rate.

The trial where the controller(s) produced the maximum response rate is the Maximum Asynchronous Message Processing Rate. Of course, the first trial could begin at a low sending rate with zero lost responses, and increase until the Loss-free and Maximum Rates are discovered.

Measurement Units:
Messages processed per second.

See Also:
None

2.3.1.4. Reactive Path Provisioning Time

Definition:
The time taken by the controller to setup a path reactively between source and destination node, defined as the interval starting with the first flow provisioning request message received by the controller(s), ending with the last flow provisioning response message sent from the controller(s) at it Southbound interface.

Discussion:
As SDN supports agile provisioning, it is important to measure how fast that the controller provisions an end-to-end flow in the dataplane. The benchmark is obtained by sending traffic from a source endpoint to the destination endpoint, finding the time difference between the first and the last flow provisioning message exchanged between the controller and the Network Devices for the traffic path.

Measurement Units:
milliseconds.

See Also:
None

2.3.1.5. Proactive Path Provisioning Time

Definition:
The time taken by the controller to setup a path proactively between source and destination node, defined as the interval starting with

the first proactive flow provisioned in the controller(s) at its Northbound interface, ending with the last flow provisioning response message sent from the controller(s) at it Southbound interface.

Discussion:
For SDN to support pre-provisioning of traffic path from application, it is important to measure how fast that the controller provisions an end-to-end flow in the dataplane. The benchmark is obtained by provisioning a flow on controller's northbound interface for the traffic to reach from a source to a destination endpoint, finding the time difference between the first and the last flow provisioning message exchanged between the controller and the Network Devices for the traffic path.

Measurement Units:
milliseconds.

See Also:
None

2.3.1.6. Reactive Path Provisioning Rate

Definition:
The maximum number of independent paths a controller can concurrently establish between source and destination nodes reactively, defined as the number of paths provisioned by the controller(s) at its Southbound interface for the flow provisioning requests received for path provisioning at its Southbound interface between the start of the trail and the expiry of given trail duration

Discussion:
For SDN to support agile traffic forwarding, it is important to measure how many end-to-end flows that the controller could setup in the dataplane. This benchmark is obtained by sending traffic each with unique source and destination pairs from the source Network Device and determine the number of frames received at the destination Network Device.

Measurement Units:
Paths provisioned per second.

See Also:
None

2.3.1.7. Proactive Path Provisioning Rate

Definition:
Measure the maximum number of independent paths a controller can concurrently establish between source and destination nodes proactively, defined as the number of paths provisioned by the controller(s) at its Southbound interface for the paths provisioned in its Northbound interface between the start of the trail and the expiry of given trail duration

Discussion:
For SDN to support pre-provisioning of traffic path for a larger network from the application, it is important to measure how many end-to-end flows that the controller could setup in the dataplane. This benchmark is obtained by sending traffic each with unique source and destination pairs from the source Network Device. Program the flows on controller’s northbound interface for traffic to reach from each of the unique source and destination pairs and determine the number of frames received at the destination Network Device.

Measurement Units:
Paths provisioned per second.

See Also:
None

2.3.1.8. Network Topology Change Detection Time

Definition:
The amount of time required for the controller to detect any changes in the network topology, defined as the interval starting with the notification message received by the controller(s) at its Southbound interface, ending with the first topology rediscovery messages sent from the controller(s) at its Southbound interface.

Discussion:
In order to for the controller to support fast network failure recovery, it is critical to measure how fast the controller is able to detect any network-state change events. This benchmark is obtained by triggering a topology change event and measuring the time controller takes to detect and initiate a topology re-discovery process.

Measurement Units:
milliseconds

See Also:

None

2.3.2. Scalability

2.3.2.1. Control Sessions Capacity

Definition:
Measure the maximum number of control sessions the controller can maintain, defined as the number of sessions that the controller can accept from network devices, starting with the first control session, ending with the last control session that the controller(s) accepts at its Southbound interface.

Discussion:
Measuring the controller's control sessions capacity is important to determine the controller's system and bandwidth resource requirements. This benchmark is obtained by establishing control session with the controller from each of the Network Device until it fails. The number of sessions that were successfully established will provide the Control Sessions Capacity.

Measurement Units:
N/A

See Also:
None

2.3.2.2. Network Discovery Size

Definition:
Measure the network size (number of nodes, links and hosts) that a controller can discover, defined as the size of a network that the controller(s) can discover, starting from a network topology given by the user for discovery, ending with the topology that the controller(s) could successfully discover.

Discussion:
For optimal network planning, it is key to measure the maximum network size that the controller can discover. This benchmark is obtained by presenting an initial set of Network Devices for discovery to the controller. Based on the initial discovery, the number of Network Devices is increased or decreased to determine the maximum nodes that the controller can discover.

Measurement Units:

N/A

See Also:
None

2.3.2.3. Forwarding Table Capacity

Definition:
The maximum number of flow entries that a controller can manage in its Forwarding table.

Discussion:
It is significant to measure the capacity of controller's Forwarding Table to determine the number of flows that controller could forward without flooding/dropping. This benchmark is obtained by continuously presenting the controller with new flow entries through reactive or proactive flow provisioning mode until the forwarding table becomes full. The maximum number of nodes that the controller can hold in its Forwarding Table will provide Forwarding Table Capacity.

Measurement Units:
Maximum number of flow entries managed.

See Also:
None

2.3.3. Security

2.3.3.1. Exception Handling

Definition:
To determine the effect of handling error packets and notifications on performance tests.

Discussion:
This benchmark test is to be performed after obtaining the baseline performance of the performance tests defined in Section 2.3.1. This benchmark determines the deviation from the baseline performance due to the handling of error or failure messages from the connected Network Devices.

Measurement Units:
N/A

See Also:

None

2.3.3.2. Denial of Service Handling

Definition:
To determine the effect of handling denial of service (DoS) attacks on performance and scalability tests.

Discussion:
This benchmark test is to be performed after obtaining the baseline performance of the performance and scalability tests defined in section 2.3.1 and section 2.3.1.. This benchmark determines the deviation from the baseline performance due to the handling of denial of service attacks on controller.

Measurement Units:
Deviation of baseline metrics while handling Denial of Service Attacks.

See Also:
None

2.3.4. Reliability

2.3.4.1. Controller Failover Time

Definition:
The time taken to switch from an active controller to the backup controller, when the controllers work in redundancy mode and the active controller fails, defined as the interval starting with the active controller bringing down, ending with the first re-discovery message received from the new controller at its Southbound interface.

Discussion:
This benchmark determine the impact of provisioning new flows when controllers are teamed and the active controller fails.

Measurement Units:
milliseconds.

See Also:
None

2.3.4.2. Network Re-Provisioning Time

Definition:
The time taken to re-route the traffic by the Controller, when there is a failure in existing traffic paths, defined as the interval starting from the first failure notification message received by the controller, ending with the last flow re-provisioning message sent by the controller at its Southbound interface .

Discussion:
This benchmark determines the controller's re-provisioning ability upon network failures. This benchmark test assumes the following:

i. Network topology supports redundant path between source and destination endpoints.
ii. Controller does not pre-provision the redundant path.

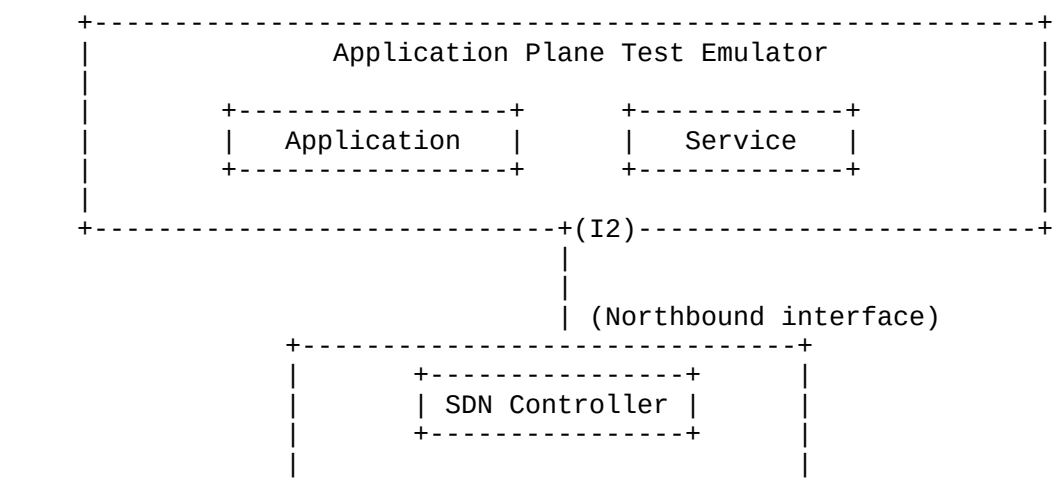
Measurement Units:
milliseconds.

See Also:
None

3. Test Setup

This section provides common reference topologies that are later referred to in individual tests defined in the companion methodology document.

3.1. Test setup - Controller working in Standalone Mode



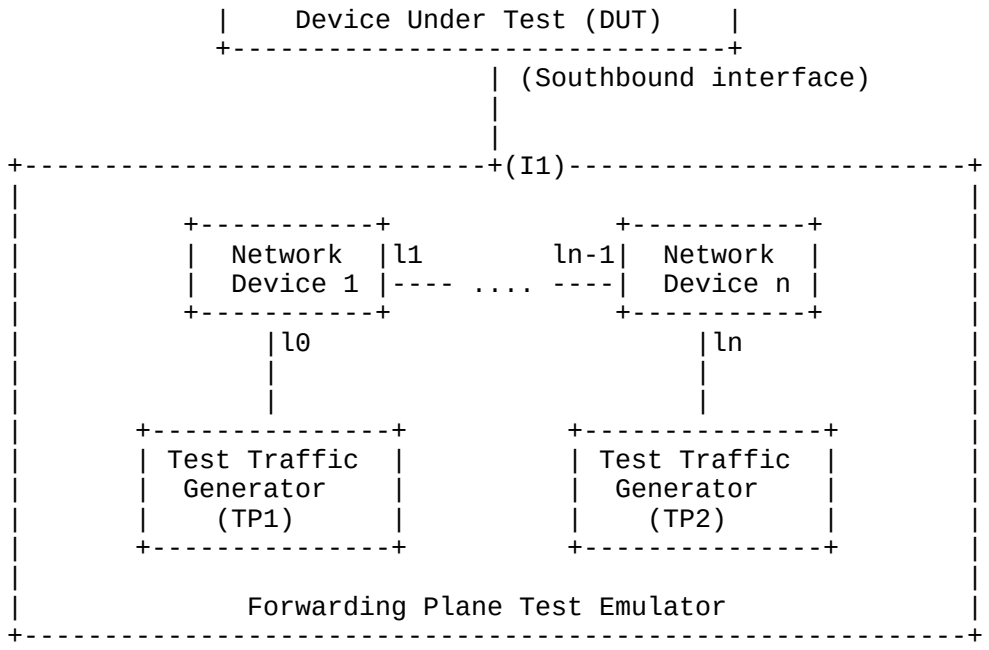
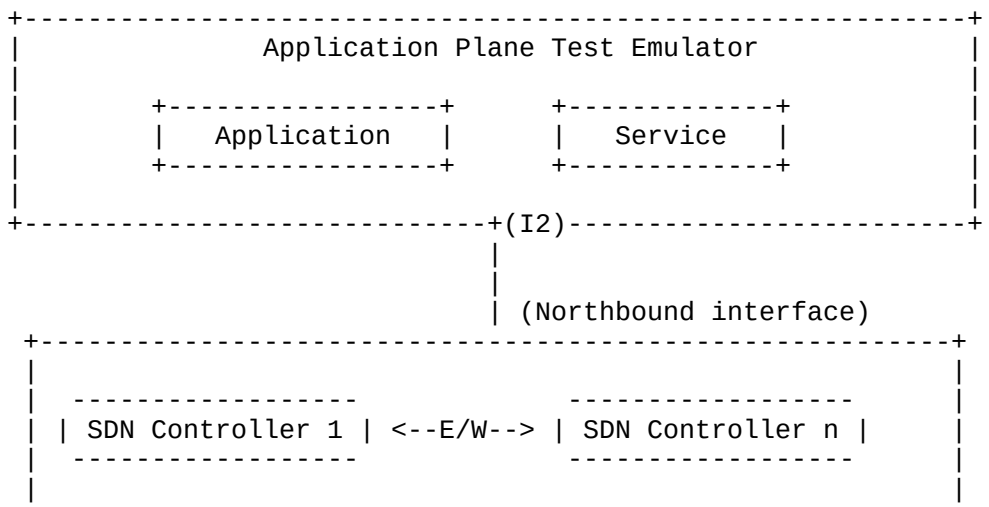


Figure 1

3.2. Test setup - Controller working in Cluster Mode



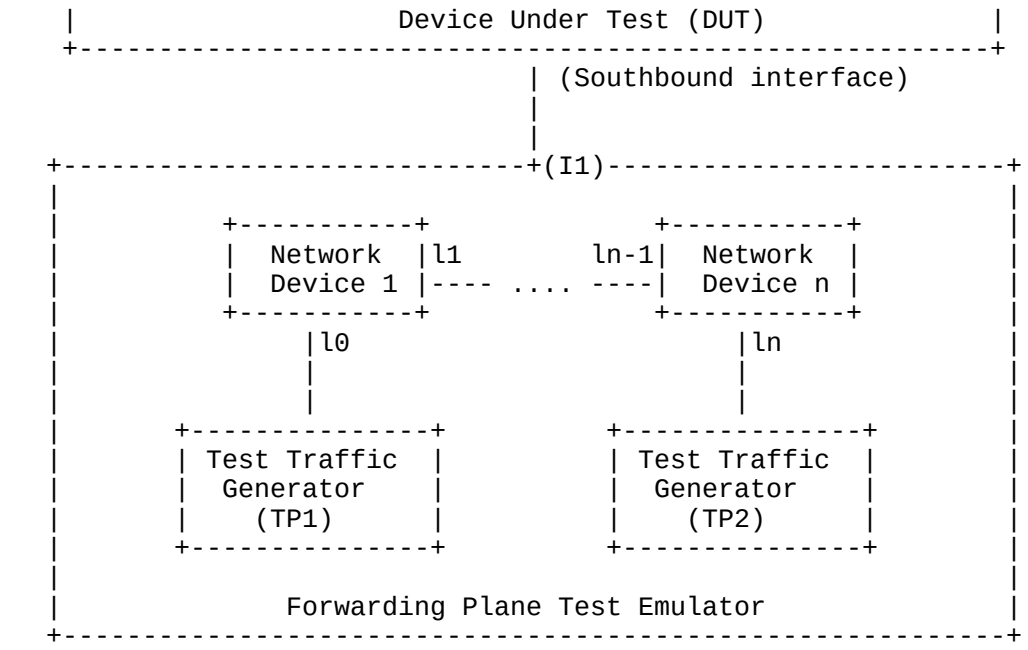


Figure 2

4. Test Coverage

	Speed	Scalability	Reliability
Setup	1. Network Topology Discovery 2. Reactive Path Provisioning Time 3. Proactive Path Provisioning Time 4. Reactive Path Provisioning Rate 5. Proactive Path	1. Network Discovery Size	

	Provisioning Rate			
Operational	1. Asynchronous Message Proces-sing Rate 2. Asynchronous Message Proces-sing Time	1. Control Sessions Capacity 2. Forwarding Table Capacity	1. Network Topology Change Detection Time 2. Exception Handling 3. Denial of Service Handling 4. Network Re-Provisioning Time	
Tear Down			1. Controller Failover Time	

5. References

5.1. Normative References

[RFC7426] E. Haleplidis, K. Pentikousis, S. Denazis, J. Hadi Salim, D. Meyer, O. Koufopavlou "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, January 2015.

[RFC4689] S. Poretsky, J. Perser, S. Erramilli, S. Khurana "Terminology for Benchmarking Network-layer Traffic Control Mechanisms", RFC 4689, October 2006.

[RFC2330] V. Paxson, G. Almes, J. Mahdavi, M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.

[I-D.sdn-controller-benchmark-meth] Bhuvaneswaran.V, Anton Basil, Mark.T, Vishwas Manral, Sarah Banks "Benchmarking Methodology for SDN Controller Performance", draft-ietf-bmwg-sdn-controller-benchmark-meth-05 (Work in progress), October 1, 2017

5.2. Informative References

[OpenFlow Switch Specification] ONF,"OpenFlow Switch Specification" Version 1.4.0 (Wire Protocol 0x05), October 14, 2013.

6. IANA Considerations

This document does not have any IANA requests.

7. Security Considerations

Security issues are not discussed in this memo.

8. Acknowledgements

The authors would like to acknowledge Al Morton (AT&T) for the significant contributions to the earlier versions of this document. The authors would like to thank the following individuals for providing their valuable comments to the earlier versions of this document: Sandeep Gangadharan (HP), M. Georgescu (NAIST), Andrew McGregor (Google), Scott Bradner (Harvard University), Jay Karthik (Cisco), Ramakrishnan (Dell), Khasanov Boris (Huawei).

9. Authors' Addresses

Bhuvaneswaran Vengainathan
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: bhuvaneswaran.vengainathan@veryxtech.com

Anton Basil
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: anton.basil@veryxtech.com

Mark Tassinari
Hewlett-Packard,
8000 Foothills Blvd,
Roseville, CA 95747

Email: mark.tassinari@hpe.com

Vishwas Manral
Nano Sec,CA

Email: vishwas.manral@gmail.com

Sarah Banks
VSS Monitoring

Email: sbanks@encrypted.net

Internet-Draft
Network Working Group
Intended Status: Informational
Expires: November 25, 2018

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Mark Tassinari
Hewlett-Packard
Vishwas Manral
Nano Sec
Sarah Banks
VSS Monitoring
May 25, 2018

Terminology for Benchmarking SDN Controller Performance
draft-ietf-bmwg-sdn-controller-benchmark-term-10

Abstract

This document defines terminology for benchmarking an SDN controller's control plane performance. It extends the terminology already defined in RFC 7426 for the purpose of benchmarking SDN controllers. The terms provided in this document help to benchmark SDN controller's performance independent of the controller's supported protocols and/or network services.

Table of Contents

1. Introduction

Software Defined Networking (SDN) is a networking architecture in which network control is decoupled from the underlying forwarding function and is placed in a centralized location called the SDN controller. The SDN controller provides an abstraction of the underlying network and offers a global view of the overall network to applications and business logic. Thus, an SDN controller provides the flexibility to program, control, and manage network behaviour dynamically through northbound and southbound interfaces. Since the network controls are logically centralized, the need to benchmark the SDN controller performance becomes significant. This document defines terms to benchmark various controller designs for performance, scalability, reliability and security, independent of northbound and southbound protocols. A mechanism for benchmarking the performance of SDN controllers is defined in the companion methodology document [I-D.sdn-controller-benchmark-meth]. These two documents provide a method to measure and evaluate the performance of various controller implementations.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Term Definitions

2.1. SDN Terms

The terms defined in this section are extensions to the terms defined in [RFC7426] "Software-Defined Networking (SDN): Layers and Architecture Terminology". That RFC should be referred before attempting to make use of this document.

2.1.1. Flow

Definition:
The definition of Flow is same as microflows defined in [RFC4689] Section 3.1.5.

Discussion:
A flow can be set of packets having same source address, destination address, source port and destination port, or any of these combinations.

Measurement Units:
N/A

See Also:
None

2.1.2. Northbound Interface

Definition:
The definition of northbound interface is same the Service Interface defined in [RFC7426].

Discussion:
The northbound interface allows SDN applications and orchestration systems to program and retrieve the network information through the SDN controller.

Measurement Units:
N/A

See Also:
None

2.1.3. Southbound Interface

Definition:
The southbound interface is the application programming interface provided by the SDN controller to interact with the SDN nodes.

Discussion:
Southbound interface enables controller to interact with the SDN nodes in the network for dynamically defining the traffic forwarding behaviour.

Measurement Units:
N/A

See Also:
None

2.1.4. Controller Forwarding Table

Definition:
A controller forwarding table contains flow entries learned in one of two ways: first, entries could be learned from traffic received

through the data plane, or second, these entries could be statically provisioned on the controller and distributed to devices via the southbound interface.

Discussion:
The controller forwarding table has an aging mechanism which will be applied only for dynamically learned entries.

Measurement Units:
N/A

See Also:
None

2.1.5. Proactive Flow Provisioning Mode

Definition:
Controller programming flows in Network Devices based on the flow entries provisioned through controller’s northbound interface.

Discussion:
Network orchestration systems and SDN applications can define the network forwarding behaviour by programming the controller using proactive flow provisioning. The controller can then program the Network Devices with the pre-provisioned entries.

Measurement Units:
N/A

See Also:
None

2.1.6. Reactive Flow Provisioning Mode

Definition:
Controller programming flows in Network Devices based on the traffic received from Network Devices through controller’s southbound interface

Discussion:
The SDN controller dynamically decides the forwarding behaviour based on the incoming traffic from the Network Devices. The controller then programs the Network Devices using Reactive Flow Provisioning.

Measurement Units:
N/A

See Also:
None

2.1.7. Path

Definition:
Refer to Section 5 in [RFC2330]

Discussion:
None

Measurement Units:
N/A

See Also:
None

2.1.8. Standalone Mode

Definition:
Single controller handling all control plane functionalities without redundancy, or the ability to provide high availability and/or automatic failover.

Discussion:
In standalone mode, one controller manages one or more network domains.

Measurement Units:
N/A

See Also:
None

2.1.9. Cluster/Redundancy Mode

Definition:
A group of 2 or more controllers handling all control plane functionalities.

Discussion:
In cluster mode, multiple controllers are teamed together for the purpose of load sharing and/or high availability. The controllers in

the group may work in active/standby (master/slave) or active/active (equal) mode depending on the intended purpose.

Measurement Units:
N/A

See Also:
None

2.1.10. Asynchronous Message

Definition:
Any message from the Network Device that is generated for network events.

Discussion:
Control messages like flow setup request and response message is classified as asynchronous message. The controller has to return a response message. Note that the Network Device will not be in blocking mode and continues to send/receive other control messages.

Measurement Units:
N/A

See Also:
None

2.1.11. Test Traffic Generator

Definition:
Test Traffic Generator is an entity that generates/receives network traffic.

Discussion:
Test Traffic Generator typically connects with Network Devices to send/receive real-time network traffic.

Measurement Units:
N/A

See Also:
None

2.1.12. Leaf-Spine Topology

Definition:
Leaf-Spine is a two layered network topology, where a series of leaf switches, form the access layer, are fully meshed to a series of spine switches that form the backbone layer.

Discussion:
In Leaf-Spine Topology, every leaf switch is connected to each of the spine switches in the topology.

Measurement Units:
N/A

See Also:
None

2.2. Test Configuration/Setup Terms

2.2.1. Number of Network Devices

Definition:
The number of Network Devices present in the defined test topology.

Discussion:
The Network Devices defined in the test topology can be deployed using real hardware or emulated in hardware platforms.

Measurement Units:
Number of network devices

See Also:
None

2.2.2. Trial Repetition

Definition:
The number of times the test needs to be repeated.

Discussion:
The test needs to be repeated for multiple iterations to obtain a reliable metric. It is recommended that this test SHOULD be performed for at least 10 iterations to increase the confidence in measured result.

Measurement Units:
Number of trials

See Also:
None

2.2.3. Trial Duration

Definition:
Defines the duration of test trials for each iteration.

Discussion:
Trial duration forms the basis for stop criteria for benchmarking tests. Trials not completed within this time interval is considered as incomplete.

Measurement Units:
Seconds

See Also:
None

2.2.4. Number of Cluster nodes

Definition:
Defines the number of controllers present in the controller cluster.

Discussion:
This parameter is relevant when testing the controller performance in clustering/teaming mode. The number of nodes in the cluster MUST be greater than 1.

Measurement Units:
Number of controller nodes

See Also:
None

2.3. Benchmarking Terms

This section defines metrics for benchmarking the SDN controller. The procedure to perform the defined metrics is defined in the accompanying methodology document[I-D.sdn-controller-benchmark-meth]

2.3.1. Performance

2.3.1.1. Network Topology Discovery Time

Definition:
The time taken by controller(s) to determine the complete network topology, defined as the interval starting with the first discovery message from the controller(s) at its Southbound interface, ending with all features of the static topology determined.

Discussion:
Network topology discovery is key for the SDN controller to provision and manage the network. So it is important to measure how quickly the controller discovers the topology to learn the current network state. This benchmark is obtained by presenting a network topology (Tree, Mesh or Linear) with the given number of nodes to the controller and wait for the discovery process to complete. It is expected that the controller supports network discovery mechanism and uses protocol messages for its discovery process.

Measurement Units:
Milliseconds

See Also:
None

2.3.1.2. Asynchronous Message Processing Time

Definition:
The time taken by controller(s) to process an asynchronous message, defined as the interval starting with an asynchronous message from a network device after the discovery of all the devices by the controller(s), ending with a response message from the controller(s) at its Southbound interface.

Discussion:
For SDN to support dynamic network provisioning, it is important to measure how quickly the controller responds to an event triggered from the network. The event could be any notification messages generated by a Network Device upon arrival of a new flow, link down etc. This benchmark is obtained by sending asynchronous messages from every connected Network Devices one at a time for the defined trial duration. This test assumes that the controller will respond to the received asynchronous message.

Measurement Units:
Milliseconds

See Also:
None

2.3.1.3. Asynchronous Message Processing Rate

Definition:
The number responses to asynchronous messages per second (such as new flow arrival notification message, link down, etc.) for which the controller(s) performed processing and replied with a valid and productive (non-trivial) response message.

Discussion:
As SDN assures flexible network and agile provisioning, it is important to measure how many network events (such as new flow arrival notification message, link down, etc.) the controller can handle at a time. This benchmark is measured by sending asynchronous messages from every connected Network Device at the rate that the controller processes (without dropping them). This test assumes that the controller responds to all the received asynchronous messages (the messages can be designed to elicit individual responses).

When sending asynchronous messages to the controller(s) at high rates, some messages or responses may be discarded or corrupted and require retransmission to controller(s). Therefore, a useful qualification on Asynchronous Message Processing Rate is whether the in-coming message count equals the response count in each trial. This is called the Loss-free Asynchronous Message Processing Rate.

Note that several of the early controller benchmarking tools did not consider lost messages, and instead report the maximum response rate. This is called the Maximum Asynchronous Message Processing Rate.

To characterize both the Loss-free and Maximum Rates, a test could begin the first trial by sending asynchronous messages to the controller(s) at the maximum possible rate and record the message reply rate and the message loss rate. The message sending rate is then decreased by the step-size. The message reply rate and the message loss rate are recorded. The test ends with a trial where the controller(s) processes the all asynchronous messages sent without loss. This is the Loss-free Asynchronous Message Processing Rate.

The trial where the controller(s) produced the maximum response rate is the Maximum Asynchronous Message Processing Rate. Of course, the first trial could begin at a low sending rate with zero lost responses, and increase until the Loss-free and Maximum Rates are discovered.

Measurement Units:
Messages processed per second.

See Also:
None

2.3.1.4. Reactive Path Provisioning Time

Definition:
The time taken by the controller to setup a path reactively between source and destination node, defined as the interval starting with the first flow provisioning request message received by the controller(s), ending with the last flow provisioning response message sent from the controller(s) at its Southbound interface.

Discussion:
As SDN supports agile provisioning, it is important to measure how fast that the controller provisions an end-to-end flow in the dataplane. The benchmark is obtained by sending traffic from a source endpoint to the destination endpoint, finding the time difference between the first and the last flow provisioning message exchanged between the controller and the Network Devices for the traffic path.

Measurement Units:
Milliseconds.

See Also:
None

2.3.1.5. Proactive Path Provisioning Time

Definition:
The time taken by the controller to proactively setup a path between source and destination node, defined as the interval starting with the first proactive flow provisioned in the controller(s) at its Northbound interface, ending with the last flow provisioning command message sent from the controller(s) at its Southbound interface.

Discussion:

For SDN to support pre-provisioning of traffic path from application, it is important to measure how fast that the controller provisions an end-to-end flow in the dataplane. The benchmark is obtained by provisioning a flow on controller's northbound interface for the traffic to reach from a source to a destination endpoint, finding the time difference between the first and the last flow provisioning message exchanged between the controller and the Network Devices for the traffic path.

Measurement Units:
Milliseconds.

See Also:
None

2.3.1.6. Reactive Path Provisioning Rate

Definition:
The maximum number of independent paths a controller can concurrently establish per second between source and destination nodes reactively, defined as the number of paths provisioned per second by the controller(s) at its Southbound interface for the flow provisioning requests received for path provisioning at its Southbound interface between the start of the trial and the expiry of given trial duration.

Discussion:
For SDN to support agile traffic forwarding, it is important to measure how many end-to-end flows that the controller could setup in the dataplane. This benchmark is obtained by sending traffic each with unique source and destination pairs from the source Network Device and determine the number of frames received at the destination Network Device.

Measurement Units:
Paths provisioned per second.

See Also:
None

2.3.1.7. Proactive Path Provisioning Rate

Definition:
Measure the maximum number of independent paths a controller can concurrently establish per second between source and destination nodes proactively, defined as the number of paths provisioned per

second by the controller(s) at its Southbound interface for the paths provisioned in its Northbound interface between the start of the trial and the expiry of given trial duration.

Discussion:
For SDN to support pre-provisioning of traffic path for a larger network from the application, it is important to measure how many end-to-end flows that the controller could setup in the dataplane. This benchmark is obtained by sending traffic each with unique source and destination pairs from the source Network Device. Program the flows on controller’s northbound interface for traffic to reach from each of the unique source and destination pairs and determine the number of frames received at the destination Network Device.

Measurement Units:
Paths provisioned per second.

See Also:
None

2.3.1.8. Network Topology Change Detection Time

Definition:
The amount of time required for the controller to detect any changes in the network topology, defined as the interval starting with the notification message received by the controller(s) at its Southbound interface, ending with the first topology rediscovery messages sent from the controller(s) at its Southbound interface.

Discussion:
In order for the controller to support fast network failure recovery, it is critical to measure how fast the controller is able to detect any network-state change events. This benchmark is obtained by triggering a topology change event and measuring the time controller takes to detect and initiate a topology re-discovery process.

Measurement Units:
Milliseconds

See Also:
None

2.3.2. Scalability

2.3.2.1. Control Sessions Capacity

Definition:
Measure the maximum number of control sessions the controller can maintain, defined as the number of sessions that the controller can accept from network devices, starting with the first control session, ending with the last control session that the controller(s) accepts at its Southbound interface.

Discussion:
Measuring the controller's control sessions capacity is important to determine the controller's system and bandwidth resource requirements. This benchmark is obtained by establishing control session with the controller from each of the Network Device until it fails. The number of sessions that were successfully established will provide the Control Sessions Capacity.

Measurement Units:
Maximum number of control sessions

See Also:
None

2.3.2.2. Network Discovery Size

Definition:
Measure the network size (number of nodes and links) that a controller can discover, defined as the size of a network that the controller(s) can discover, starting from a network topology given by the user for discovery, ending with the topology that the controller(s) could successfully discover.

Discussion:
For optimal network planning, it is key to measure the maximum network size that the controller can discover. This benchmark is obtained by presenting an initial set of Network Devices for discovery to the controller. Based on the initial discovery, the number of Network Devices is increased or decreased to determine the maximum nodes that the controller can discover.

Measurement Units:
Maximum number of network nodes and links

See Also:
None

2.3.2.3. Forwarding Table Capacity

Definition:
The maximum number of flow entries that a controller can manage in its Forwarding table.

Discussion:
It is significant to measure the capacity of controller's Forwarding Table to determine the number of flows that controller could forward without flooding/dropping. This benchmark is obtained by continuously presenting the controller with new flow entries through reactive or proactive flow provisioning mode until the forwarding table becomes full. The maximum number of nodes that the controller can hold in its Forwarding Table will provide Forwarding Table Capacity.

Measurement Units:
Maximum number of flow entries managed.

See Also:
None

2.3.3. Security

2.3.3.1. Exception Handling

Definition:
To determine the effect of handling error packets and notifications on performance tests.

Discussion:
This benchmark test is to be performed after obtaining the baseline performance of the performance tests defined in Section 2.3.1. This benchmark determines the deviation from the baseline performance due to the handling of error or failure messages from the connected Network Devices.

Measurement Units:
Deviation of baseline metrics while handling Exceptions.

See Also:
None

2.3.3.2. Denial of Service Handling

Definition:
To determine the effect of handling denial of service (DoS) attacks on performance and scalability tests.

Discussion:
This benchmark test is to be performed after obtaining the baseline performance of the performance and scalability tests defined in section 2.3.1 and section 2.3.2. This benchmark determines the deviation from the baseline performance due to the handling of denial of service attacks on controller.

Measurement Units:
Deviation of baseline metrics while handling Denial of Service Attacks.

See Also:
None

2.3.4. Reliability

2.3.4.1. Controller Failover Time

Definition:
The time taken to switch from an active controller to the backup controller, when the controllers work in redundancy mode and the active controller fails, defined as the interval starting with the active controller bringing down, ending with the first re-discovery message received from the new controller at its Southbound interface.

Discussion:
This benchmark determines the impact of provisioning new flows when controllers are teamed and the active controller fails.

Measurement Units:
Milliseconds.

See Also:
None

2.3.4.2. Network Re-Provisioning Time

Definition:
The time taken to re-route the traffic by the Controller, when there is a failure in existing traffic paths, defined as the interval starting from the first failure notification message received by the controller, ending with the last flow re-provisioning message sent by the controller at its Southbound interface.

Discussion:
This benchmark determines the controller's re-provisioning ability upon network failures. This benchmark test assumes the following:
1. Network topology supports redundant path between source and destination endpoints.
2. Controller does not pre-provision the redundant path.

Measurement Units:
Milliseconds.

See Also:
None

3. Test Setup

This section provides common reference topologies that are later referred to in individual tests defined in the companion methodology document.

3.1. Test setup - Controller working in Standalone Mode

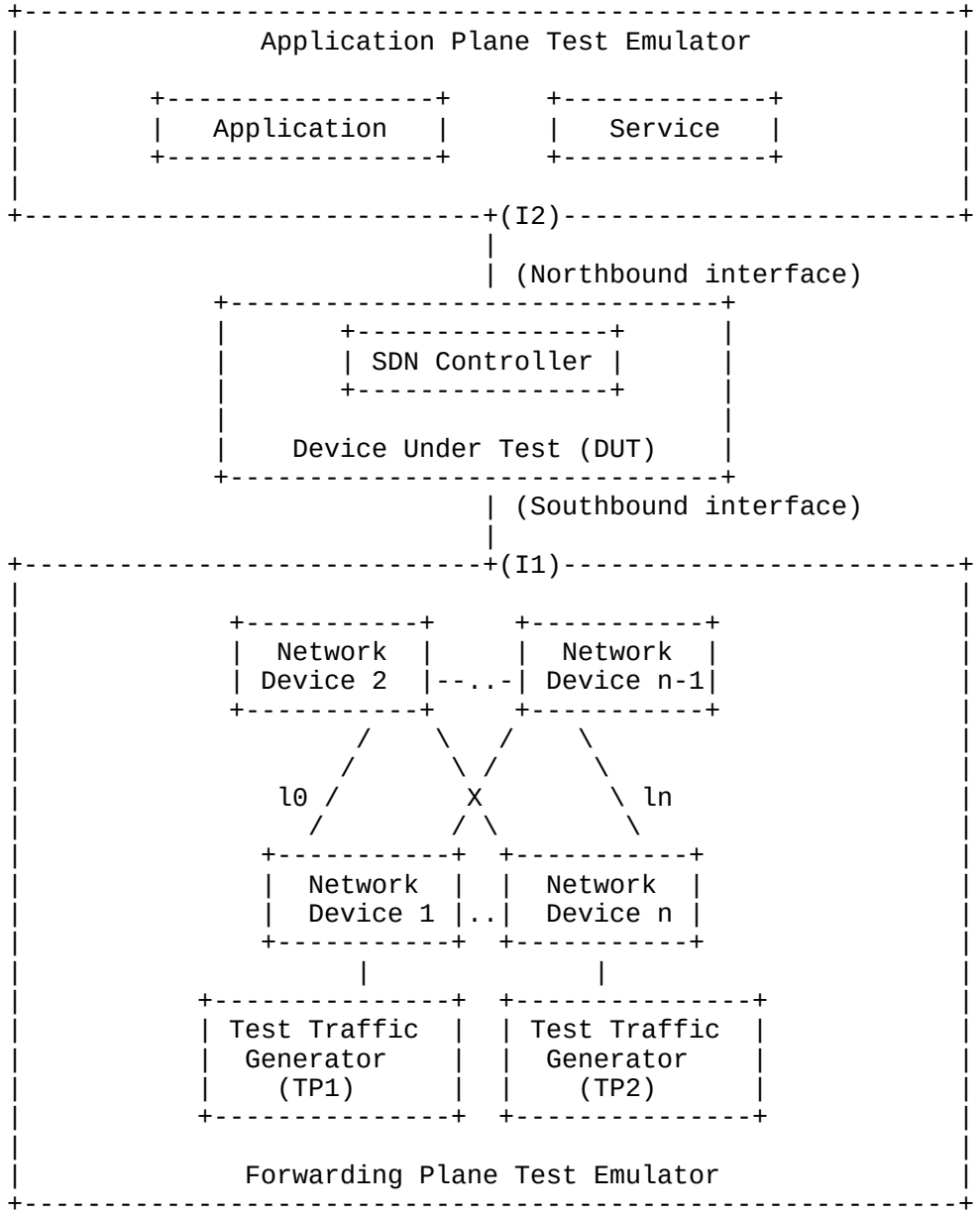


Figure 1

3.2. Test setup - Controller working in Cluster Mode

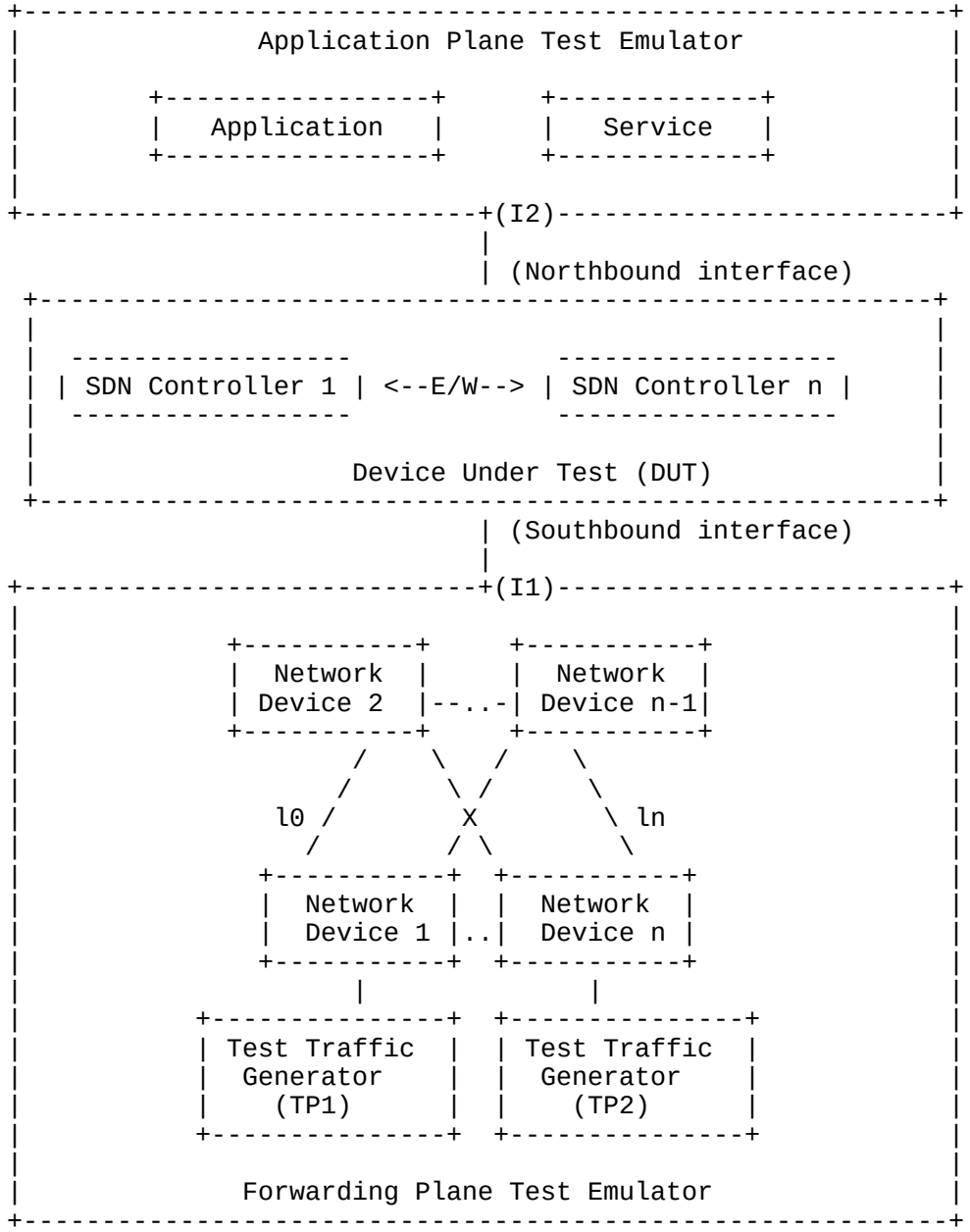


Figure 2

4. Test Coverage

Lifecycle	Speed	Scalability	Reliability
Setup	1. Network Topology Discovery Time 2. Reactive Path Provisioning Time 3. Proactive Path Provisioning Time 4. Reactive Path Provisioning Rate 5. Proactive Path Provisioning Rate	1. Network Discovery Size	
Operational	1. Maximum Asynchronous Message Processing Rate 2. Loss-Free Asynchronous Message Processing Rate 3. Asynchronous Message Processing Time	1. Control Sessions Capacity 2. Forwarding Table Capacity	1. Network Topology Change Detection Time 2. Exception Handling 3. Denial of Service Handling 4. Network Re-Provisioning Time
Tear Down			1. Controller Failover Time

5. References

5.1. Normative References

[RFC7426] E. Haleplidis, K. Pentikousis, S. Denazis, J. Hadi Salim, D. Meyer, O. Koufopavlou "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, January 2015.

[RFC4689] S. Poretsky, J. Perser, S. Erramilli, S. Khurana "Terminology for Benchmarking Network-layer Traffic Control Mechanisms", RFC 4689, October 2006.

[RFC2330] V. Paxson, G. Almes, J. Mahdavi, M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.

[RFC2119] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[RFC8174] B. Leiba, "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017.

[I-D.sdn-controller-benchmark-meth] Bhuvaneswaran.V, Anton Basil, Mark.T, Vishwas Manral, Sarah Banks "Benchmarking Methodology for SDN Controller Performance", draft-ietf-bmwg-sdn-controller-benchmark-meth-09 (Work in progress), May 25, 2018

5.2. Informative References

[OpenFlow Switch Specification] ONF,"OpenFlow Switch Specification" Version 1.4.0 (Wire Protocol 0x05), October 14, 2013.

6. IANA Considerations

This document does not have any IANA requests.

7. Security Considerations

Security issues are not discussed in this memo.

8. Acknowledgements

The authors would like to acknowledge Al Morton (AT&T) for the significant contributions to the earlier versions of this document. The authors would like to thank the following individuals for providing their valuable comments to the earlier versions of this document: Sandeep Gangadharan (HP), M. Georgescu (NAIST), Andrew McGregor (Google), Scott Bradner , Jay Karthik (Cisco), Ramakrishnan (Dell), Khasanov Boris (Huawei).

9. Authors' Addresses

Bhuvaneswaran Vengainathan
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: bhuvaneswaran.vengainathan@veryxtech.com

Anton Basil
Veryx Technologies Inc.
1 International Plaza, Suite 550
Philadelphia
PA 19113

Email: anton.basil@veryxtech.com

Mark Tassinari
Hewlett-Packard,
8000 Foothills Blvd,
Roseville, CA 95747

Email: mark.tassinari@hpe.com

Vishwas Manral
Nano Sec,CA

Email: vishwas.manral@gmail.com

Sarah Banks
VSS Monitoring
930 De Guigne Drive,
Sunnyvale, CA

Email: sbanks@encrypted.net

Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 4, 2018

T. Kim
B. Koo
J. Park
E. Paik
KT

July 03, 2017

Considerations for Benchmarking Service Function Chain
draft-kim-bmwg-sfc-benchmark-00

Abstract

Service Function Chain(SFC) is a ordered set of service functions. Packets flow restrictively at the service functions according to the order. To enable a network service, operator composes the service function chain logically. Though SFC is efficient where network/ service requirements are dynamically changing, the reliability of SFC should be guaranteed. This memo describes the considerations for benchmarking SFC reliability.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 4, 2018.

Table of Contents

1. Introduction

As Service Function Chain(SFC) is the ordered set of service functions. It is logically defined on demand of a service. To enable the service, SDN controller set flow rules at each physical/ virtual switch which belongs to the SFC. SFC is efficient where the network/service requirements are keep changing dynamically. The number of physical/virtual switches which will accept the flow rules is differ from the size of the domain or service.

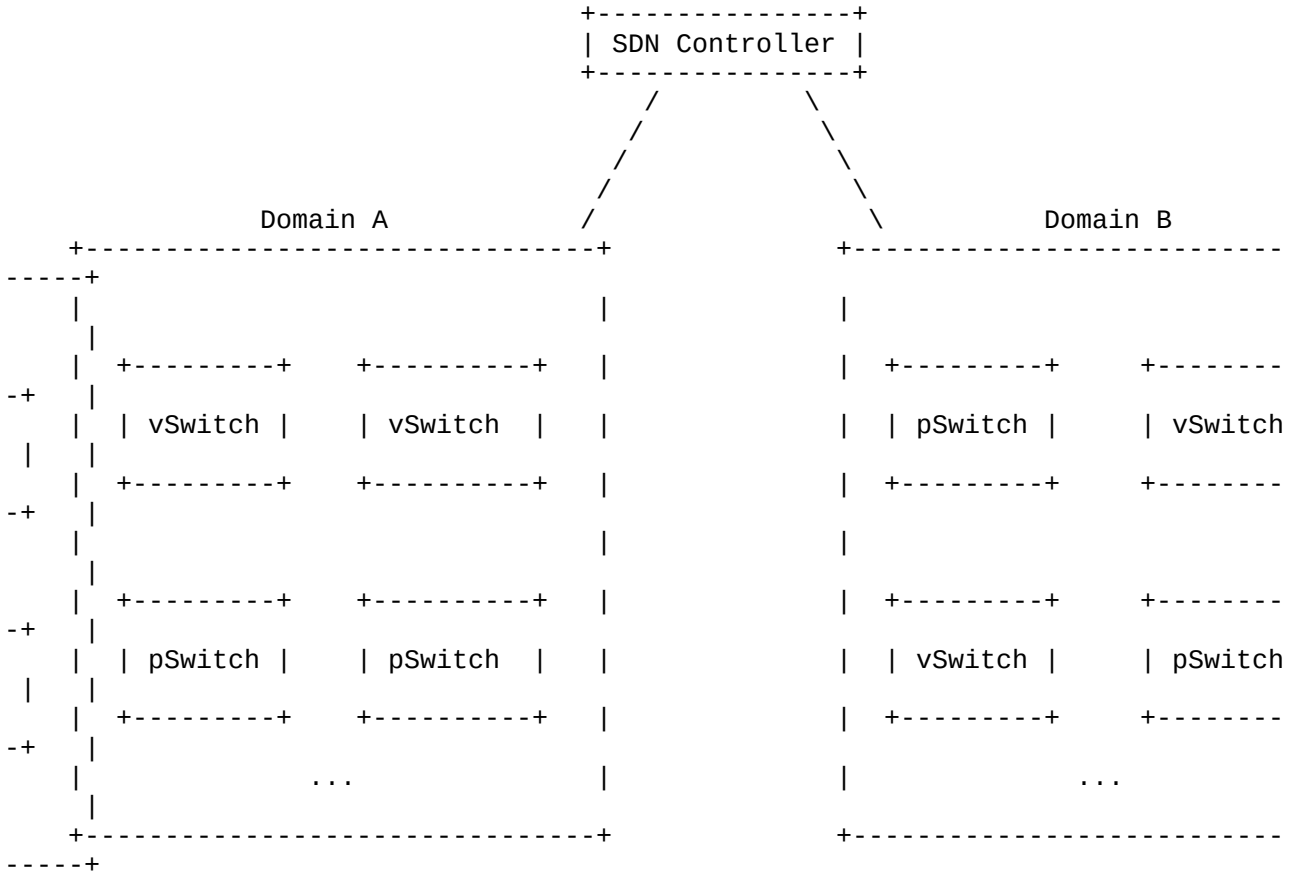
As an operator perspective, at the stage of SFC creation, modification, and deletion, the reliability of SFC should always be guaranteed. To apply the change of the SFC, SDN controller will set flow rules at some switches and delete flow rules at other switches. For certain reasons such as the heavy traffic on the target switches which should accept new rules or the link failure between the target switches and the SDN controller, the new SFC may not be applied properly.

This draft memo describes considerations for benchmarking Service Function Chain reliability.

2. Scope

At the time of writing this memo, SFC standardization is now in progress. But operators and vendors are implementing SFC their own way. This memo does not target NSH enabled architecture and target general operation circumstances. The scope of SFC reliability benchmark is when the initial SFC is already provisioned and the traffic also flows over the certain SFCs, and SFC needs to be updated. Also, SFC is made over multi-domain network, which covers the whole country.

This figure is an example of the network.



3. Considerations for Benchmarking SFC Reliability

This section defines and lists considerations which must be addressed to benchmark the reliability of SFC

3.1. Configuration Parameters for Benchmarking Test

This section lists the parameters affecting the SFC reliability. To apply new SFC, SDN controller set rules to the target switches. Depending on the status of the swithes and the network, the new SFC can be applied right as intended, or not. The right operation of SFC as intended includes the right time of the operation activates.

- Types of Switches : Virtual switch or Physical switch
- The number of switches in target SFC domain
  - Depending on the composition of the target SFC, the number of switches which need to update their flow tables is different.
- The Usage of Flow table of the target switch
  - When the new SFC rule needs to setup, if the flow table entries are not enought and have to stored elsewhere,not TCAM, the usage of flow table can affect the reliability of SFC.
    - TCAM Usage
    - Flow table Entries
- The physical distances between the Controller and Switch
  - As the network grows broad, the delay is same as propagation delay. And this make SFC Activation time different.
- The traffic loads on the target switch
  - The limitaion of the CPU, when the target switch needs to process large amount of the traffic, the new SFC rules setup cannot be done in intended time.

3.2. Testing Parameter Benchmarking Test

This section describes the testing parameter for Benchmark SFC Reliability. In terms of operation, the reliability of SFC is "operate the SFC in right time and at right path."

Rule Activation Time

- The time interval from the new flow rule setup requests to the time when packets start to flow following the new matched rule.

TBD

4. Security Considerations

TBD.

5. IANA Considerations

No IANA Action is requested at this time.

6. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <http://www.rfc-editor.org/info/rfc2544>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <http://www.rfc-editor.org/info/rfc7665>.

Authors' Addresses

Taekhee Kim
KT
Infra R&D Lab. KT
17 Woomyeon-dong, Seocho-gu
Seoul 137-792
Korea

Phone: +82-2-526-6688
Fax: +82-2-526-5200
Email: taekhee.kim@kt.com

Bummo Koo
KT
Infra R&D Lab. KT
17 Woomyeon-dong, Seocho-gu
Seoul 137-792
Korea

Phone: +82-2-526-6688
Fax: +82-2-526-5200
Email: bm.koo@kt.com

Jisu Park
KT
Infra R&D Lab. KT
17 Woomyeon-dong, Seocho-gu
Seoul 137-792
Korea

Phone: +82-2-526-6688
Fax: +82-2-526-5200
Email: jisu.park@kt.com

EunKyoung Paik
KT
Infra R&D Lab. KT
17 Woomyeon-dong, Seocho-gu
Seoul 137-792
Korea

Phone: +82-2-526-5233
Fax: +82-2-526-5200
Email: eun.paik@kt.com
URI: http://mmlab.snu.ac.kr/~eun/

INTERNET-DRAFT
Intended Status:Standard
Expires: March 7,2018

Kishore Tiruveedhula
Sudhin Jacob
Juniper Networks
October 6,2017

Benchmarking Methodology for EVPN and PBB-EVPN
draft-kishjac-bmwg-evpntest-08

Abstract

This document defines the methodologies for benchmarking performance of EVPN and PBB-EVPN.EVPN is defined in RFC 7432.It is being deployed in provider network.This document provides the benchmarking methodologies for EVPN/PBB-EVPN convergence,data plane,control plane learning of mac.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 7,2018.

Table of Contents

1 Introduction

EVPN which is defined in RFC7432 which describes procedures for BGP MPLS-based Ethernet VPNs(EVPN).This document defines the methodologies for benchmarking performance of EVPN.EVPN has been implemented with many varying designs in order to achieve their intended network functionality.The scope of this document is to provide methodologies for benchmarking evpn data,control plane mac learning,mac flush,mac ageing,convergence,high availability,scale. The methodologies defined for evpn can be used for benchmarking the performance of PBB-EVPN.PBB-EVPN is defined in RFC 7623.It is being deployed in provider network.The difference between PBB-EVPN and EVPN is the former learns the customer mac in data plane the later learns in control plane.

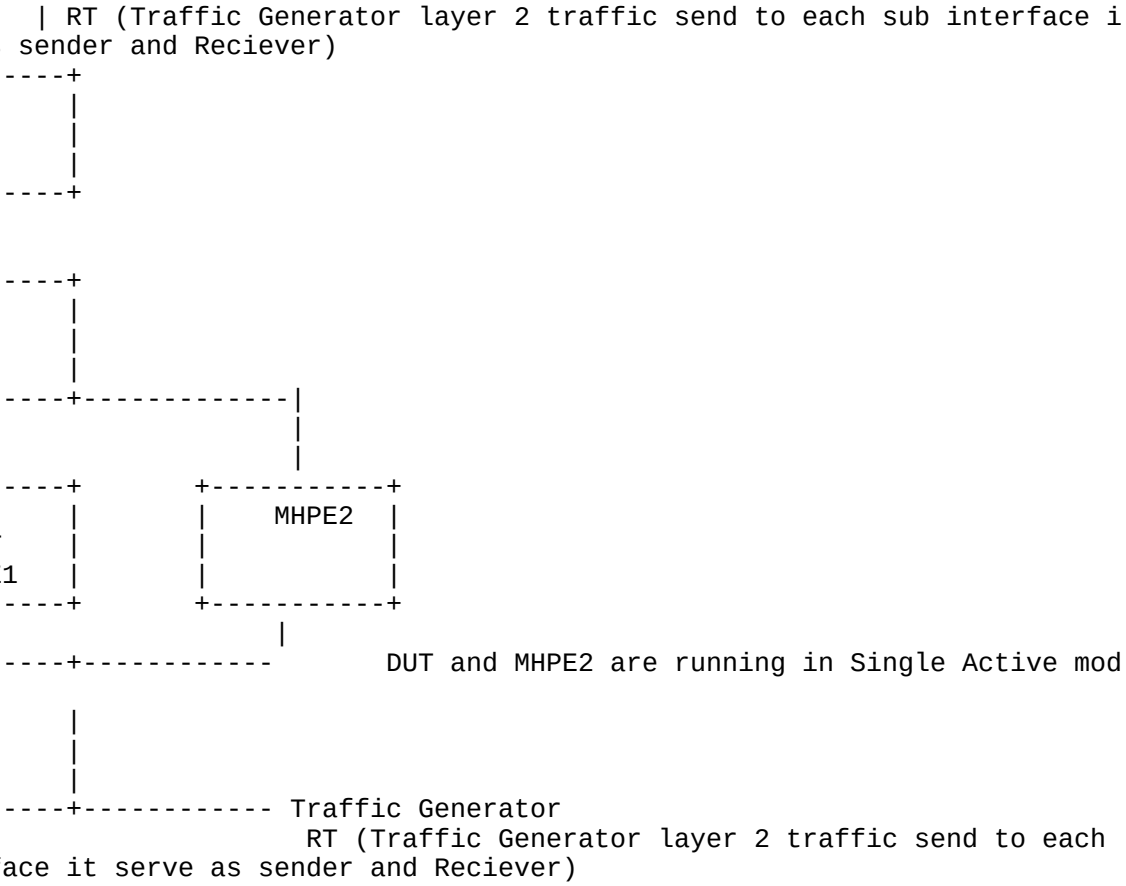
1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.Test Topology

Topology Diagram

Topology Diagram



Topology Diagram

Figure 1

2.1 Terminologies

MHPE Multi homed Provide Edge router.

RR Route Reflector

CE Customer Router/Devices

PE2 Provider Edge router 2

AA EVPN Terminologies AA All-Active

SA EVPN Terminologies SA Single-Active

RT Router Tester

3. Network

The network consists of 5 routers and 2 traffic generator ports.DUT is acting as one of the MHPE to CE.The RR is acting as route reflector and core router.R1 is a Single home router running evpn. All four routers except CE are running mpls,bgp emulating a provider scenario.CE is a dual home connected to DUT and MHPE2.The testing will be done on DUT in order to bench mark the service. DUT and the MHPE2 is running EVPN with SA/AA, CE will be configured with layer 2 bridge.In AA EVPN there will be LAG running from CE to both MHPE’s.The DUT and other PE’s will be running X EVI’s (EVPN instances) on X sub interfaces.The traffic generator will be connected to R1 and the CE, which is capable of sending layer 2 frames.The traffic will be send either uni directional or bi directional based on the benchmark parameters to be measured.

3.1 PBB-EVPN Network Setup

The network consists of 5 routers and 2 traffic generator ports.DUT is acting as one of the MHPE to CE.The RR is acting as route reflector and core router.R1 is a Single home router running pbbevpn. All four routers except CE are running mpls,bgp emulating a provider scenario.CE is a dual home connected to DUT and MHPE2.DUT and MHPE2 is running PBB-EVPN with SA/AA with CE.CE will be configured with layer 2 bridge.In AA PBB-EVPN there will be LAG running from CE to both MHPE’s.The DUT and other PE’s will be running X EVI’s (PBB-EVPN instances) on X sub interfaces.The traffic generator will be connected to R1 and the CE, which is capable of sending layer 2 frames.The traffic will be send either uni directional or bi directional based on the benchmark parameters to be measured.

4. Test Procedure

The test defined to bench mark the performance of EVPN mac learning in control pland and data plane. Mac flush,High Availability, convergence time in link failures,scale scenarios.

4.1 MAC Learning in Control plane and Data Plane

The MAC will be learned in data plane and control plane, test is to measure the time taken to learn the "X" number of mac's in time "T" sec. The data plane learning will from the locally connected interface. The control plane learning is through BGP advertisements from the remote PE. Let the local learning time be "T" and control plane learning time be "T'". The data plane mac learning can be measured using the parameters defined in RFC 2889 section 5.8.

4.2 MAC flush for locally learned and remote learned MAC

The time taken to flush the "X" locally learned mac, let it be "T1" sec, once the traffic is stopped. The time taken to flush the remote mac which is learned by control plane let it be "X" macs.The time taken to flush the "X" remote macs, which is measured as "T2" sec.

4.3 High Availability

The traffic is flowing bi direction. The bgp is converged,consider there are "X" numbers of macs learned locally and remotely. Then traffic is flowing at "P" packets per sec. The traffic generator is measuring the Tx and Rx packets, while the routing engine fail over there should not any packet loss the router tester must show both "P" packet per seconds.

4.4 Convergence Time

During any events like link failure, hard reset measure the time taken to learn "X" mac's locally and remotely.

4.5 Scale

This is to measure the performance of DUT in scaling to "X" EVPN instances. The measured parameters are CPU usage, memory leak,crashes.

4.6 SOAK

This test is used to measure the performance of DUT over a period of time,with scaled configuration and traffic over a period of time "T'". In each interval "t1" the parameters measured are CPU usage, memory usage and crashes.

4.7 Measurement Statistics

The test is repeated for "N" times and the value is taken by averaging the values.

5 Test Cases

The following tests are conducted to measure mac learning of local as well as remote.

5.1.1 To check the time taken to learn the mac address in DUT

Objective:

To check the time taken to learn the mac address locally and time taken to send the locally learned routes to peers.

a. Send X unicast frames from CE to MHPE1(DUT) working in SA mode with different source and destination address, where DUT is the DF so that it can forward the traffic.Measure the time taken to learn these mac in forwarding table and in control plane. The data plane learning is measured using RFC 2889 section 5.8. Sending frames to the limit of bridge domain of particular EVI. Measure the time taken to learn all X mac in data plane/hardware. The Range of MAC is known from RT and this is verified in DUT.

b. Measure the time taken to send these X type 2 routes from DUT to its peers.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must

be configured per IFD/Interface.Using RT (traffic generator) to send the traffic to the CE. The traffic is unidirectional.Since CE is working in bridge mode,frames will be send to ingress sub interface of DUT.

Measurement

The DUT mac table must learn the X macs in data plane in T time frame.The DUT must send type 2 routes to remote router in T' time frame.Repeat the test and plot the data.The data plane measurement is taken by considering DUT as black box the range of X mac is known from RT and the same is learned in DUT, the time to learn that is measured.

PBB-EVPN To check the time taken to learn the mac address in DUT

Objective:

To check the time taken to learn the mac address locally.

a. Send X unicast frames from CE to MHPE1(DUT) working in SA mode with different source and destination address, where DUT is the DF so that it can forward the traffic. Measure the time taken to learn these mac in forwarding table.The data plane learning is measured using RFC 2889 section 5.8.Sending frames to the limit of bridge domain of particular EVI.Measure the time taken to learn all X mac in data plane/hardware.The Range of MAC is known from RT and this is verified in DUT.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface.Using RT (traffic generator) to send the traffic to the routers.

Measurement

The DUT mac table must learn the X macs in data plane in T time frame. Repeat the test and plot the data.The data plane measurement is taken by considering DUT as black box the range of X mac is known from RT and the same is learned in DUT, the time to learn that is measured.

5.1.2. To check the time taken to learn X macs from remote peer by DUT

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator. Measure the time taken to learn these X macs from remote peer and program the mac address table.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table. For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) send the traffic to R1.The traffic is uni directional.There wont be any traffic flow from CE to DUT during this test.

Measurement:

The DUT mac table must learn the X mac address in T time frame. Repeat these test and plot the data.

PBB-EVPN To check the time taken to learn X mac’s from remote peer by DUT.

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator.Measure the time taken to learn these X macs from remote peer and program the mac address table.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and

DUT.Once the bgp comes up check the DUT pbb-evpn table. For MHPE ESI must be configured per IFD/Interface.Using RT(traffic generator) send the traffic to R1. The traffic is uni directional. There wont be any traffic flow from CE to DUT during this test.

Measurement:

The DUT mac table must learn the X mac address in T time frame. Repeat these test and plot the data.

The following test cases are executed to measure the time taken to flush the mac table in case of an event

5.1.3. To check the time taken to flush the local entry due to CE link failure and measure the relearning rate of X macs

Objective:

Send X frames with different SA and DA to DUT from CE using traffic generator.Wait for a while to learn all X mac address. Then fail the DUT CE link and measure the time taken to flush these X macs from the mac table and from control plane.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT. Once the bgp comes up check the DUT evpn table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.In this traffic will be only send from CE.

Measurement:

Measure the time taken for flushing these X mac address. Measure the time taken to relearn the X macs in other PE.Repeat the test and plot the data.

PBB-EVPN To check the time taken to flush the local entry due to CE link failure

Objective:

Send X frames with different SA and DA to DUT from CE using traffic generator.Wait for a while to learn all X mac address.Then

fail the DUT CE link and measure the time taken to flush these X macs from the mac table.Measure the time taken to relearn.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT. Once the bgp comes up check the DUT pbb-evpn table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.In this traffic will be only send from CE.

Measurement:

The DUT mac table must learn the X mac address and measure the time taken for flushing these X mac address.Measure the time taken to relearn these X macs.

5.1.4. To check the time taken by DUT to flush X routes learned from remote PE after R1 traffic generator link failure

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator.Bring down the link between R1 and traffic generator.Then measure the time taken to flush the DUT mac address table.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the R1.There wont be any traffic flowing to CE from RT.

Measurement:

Measure the time taken to flush X remote routes from mac table of DUT.Repeat the test and plot the data.

PBB-EVPN To check the time taken by DUT to flush X macs learned from remote PE after R1 traffic generator link failure

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator.Bring down the link between R1 and traffic generator.Then measure the time taken to flush the DUT mac address table.The remote macs will be learned by Data plane,but the B-MAC will be learned by control plane.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT. Once the bgp comes up check the DUT pbb-evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.In this traffic will be only send from CE.

Measurement:

Measure the time taken to flush X remote macs from mac table of DUT.

The following test cases are executed to measure the mac ageing in locally learned and remote mac learned by control plane.

5.1.5. To measure the mac ageing time.

Objective:

Send X frames with different SA and DA to DUT from CE using traffic generator.Wait to learn all X mac address.Then stop the traffic.Wait to see how long it takes to flush these mac entries due to ageing.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are

running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table. For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.The traffic will be flowing from CE to DUT.There wont be any traffic from R1.

Measurement:

Measure the time taken to flush X mac address due to ageing.Repeat the test and plot the data.

PBB-EVPN To measure the mac ageing time.

Objective:

Send X frames with different SA and DA to DUT from CE using traffic generator.Wait to learn all X mac address.Then stop the traffic.Wait to see how long it takes to flush these mac entries due to ageing.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT mac table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

The DUT mac table must learn the X mac address, measure the time taken for flushing these X mac address due to ageing.

5.1.6. To check the time taken by DUT to age X routes learned from remote PE after stopping the traffic at remote PE.

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator.After stopping the traffic at remote PE R1 traffic generator due to mac ageing it will withdraw its routes from remote PE's.Measure the time taken to remove these macs from DUT mac table.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to R1.There wont be any traffic from CE.

Measurement:

Measure the time taken to flush X remote macs from mac table of DUT due to ageing.Repeat the test and plot the data.

PBB-EVPN To check the time taken by DUT to age X mac from remote PE after stopping the traffic at remote PE.

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator.After stopping the traffic at remote PE(R1) traffic generator.Measure the time taken to remove these macs from DUT mac table.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT mac table.For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.

Measurement:

Measure the time taken to flush the remote macs from mac table of DUT due to ageing.

The following tests are executed to measure the convergence time in case of an event or by lerning the mac without any external trigger.

5.1.7. To check the time taken by DUT to learn X routes from local and X from remote and measure the time of flood from DUT.

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator.Send X frames with different SA and DA from traffic generator connected to CE.The SA and DA of flows must be complimentary to have unicast flows.Measure the time taken by the DUT to learn 2X in mac table and in control plane.Measure the flood time period of DUT.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table. For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.The traffic is bi directional.

Measurement:

Measure the time taken to learn 2X routes in control and mac address table in DUT and measure the flood time of DUT.Repeat the test and plot the data.

PBB-EVPN To check the time taken by DUT to learn X macs from local and X from remote and measure the time of flood from DUT.

Objective:

Send X frames with different SA and DA to DUT from R1 using traffic generator. Send X frames with different SA and DA from traffic generator connected to CE.The SA and DA of flows must be complimentary to have unicast flows.Measure the time taken by the DUT to learn 2X in mac table.Measure the flood time period of DUT.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.

Measurement:

Measure the time taken to learn 2X mac address table in DUT. and measure the flood time of DUT.Repeat the test and plot the data.

5.1.8. To measure the time taken to elect a new DF by adding a a MHPE.

Objective:

Send X frames from CE to DUT from traffic generator with different SA and DA.Wait to learn all X mac address.Then add a new router configured with evpn to the same ethernet segment.Configure same ESI value which is configured in DUT and in MHPE1 on IFD.Then the new DF election take place during that time there should not be any loop and measure the time taken to come up the new DF.Measure the time taken to elect the new DF.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MHPE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.

Measurement:

Measure the time taken for new DF election in DUT and there should

not be any loop or forwarding the BUM traffic back to the same segment.

PBB-EVPN To measure the time taken to elect a new DF by adding a a MHPE.

Objective:

Send X frames from CE to DUT from traffic generator with different SA and DA.Wait to learn all X mac address.Then add a new router to the same Ethernet segment by configuring the same ESI value configured in DUT,MHPE2 in IFD.Then the new DF election take place during that time there should not be any loop and measure the time taken to come up the new DF.Measure the time taken to elect the new DF.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

Measure the time taken for new DF election in DUT and there should not be any loop or forwarding the BUM traffic back to the same segment.

5.2 High Availability

These tests are conducted to check after an event there wont be any change in functionality.

5.2.1 To check the whether there is traffic loss due to routing engine failover for redundancy test.

Objective:

Send X frames from CE to DUT from traffic generator with different SA and DA. Send X frames from traffic generator to R1 with different SA and DA so that 2X mac address will be learned in DUT. There is a bi directional traffic flow with X pps in each direction. Then do a routing engine failover.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.

Measurement:

There should be 0 traffic loss which is the ideal case,No change in the DF role.No withdraw of any routes.

PBB-EVPN To check the whether there is traffic loss due to routing engine failover for redundancy test.

Objective:

Send X frames from CE to DUT from traffic generator with different SA and DA.Send X frames from traffic generator to R1 with different SA and DA so that 2X mac address will be learned in DUT. There is a bi directional traffic flow with X pps in each direction.Then do a routing engine failover.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are

running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

There should be 0 packet loss loss which is a ideal case,No change in the DF role. No withdraw of any routes.

5.3 ARP/ND Scaling

These tests are conducted to check the scaling parameter of arp/ND of the DUT.

EVPN: To check the ARP/ND scale of the DUT when the gateway is configured.

Objective:

Send X arp/icmpv6 request from RT to DUT,that with different sender ip/ipv6 address to the same target gateway ip address.

Procedure:

Configure EPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the arp request to the DUT which has gateway configured.

Measurement:

The DUT must learn X mac+ip/mac+ipv6 and it must advertise the X mac+ip/ mac+ipv6 to the remote router.

6. Scale

6.1. To Scale the DUT to N EVI and clear bgp in DUT with out traffic.

Objective:

The main purpose of the test the DUT performance to scale N EVI's. Then clear bgp neighbor. There should not be any loss of routes or any crashes.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MHPE,DUT ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement

There should not be any loss of route types 1,2,3 and 4 in DUT.

PBB-EVPN To Scale the DUT to N PBB-EVPN instances and clear bgp in DUT without traffic.

Objective:

The main purpose of the test the DUT performance to scale N PBB-EVPN instances.Then clear bgp neighbor.There should not be any loss of routes or any crashes.

Procedure:

Configure PBB-EVPN instances in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface.Using RT(traffic generator) to send the traffic to the routers.

Measurement

There should not be any loss of route types 2,3 and 4 in DUT.

6.2. To Scale the DUT to N EVI and clear bgp in DUT with traffic.Measure the convergence time

Objective:

The main purpose of the test the DUT performance to scale N EVI's with traffic. Then clear bgp neighbor.There should not be any loss of routes or any crashes. Send F frames from CE to DUT from traffic generator with different SA and DA for N EVI's. Send F frames from traffic generator to R1 with different SA and DA for N EVI's.so that 2F number of mac address will be learned in DUT. There is a bi directional traffic flow with F pps in each direction. Then clear the bgp nei in DUT after the bgp comes up and started learning the routes, measure the time taken to learn all 2F mac routes.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

The DUT must learn all 2F mac address.Measure the time taken to learn 2F mac in DUT, measure the flood traffic time "t" of DUT

PBB-EVPN To Scale the DUT to "N" PBB-EVPN instances and clear bgp in DUT with traffic.Measure the convergence time

Objective:

The main purpose of the test the DUT performance to scale N pbb-evpn instances with traffic.Then clear bgp neighbor.There should not be any loss of routes or any crashes.Send F frames from CE to DUT from traffic generator with different SA and DA for N pbb-evpn instances.Send F frames from traffic generator to R1 with different SA and DA for N pbb-evpn instances.so that F mac address will be learned in DUT.There is a bi directional traffic flow with F pps in each direction.Then clear the bgp nei in DUT after the bgp comes up and started learning the routes, measure the time taken to learn all 2F macs in DUT.

Procedure:

Configure PBB-EVPN instances in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

The DUT must learn all 2F mac address.Measure the time taken to learn 2F mac in DUT, measure the flood traffic time "t" of DUT

7. Soak Test

7.1. To Scale the DUT to N EVI in DUT with traffic and run the set up for 24hrs

Objective:

The main purpose of the test the DUT performance to scale N EVI's with traffic. Then clear bgp neighbor. There should not be any loss of routes or any crashes. Send F frames from CE to DUT from traffic generator with different SA and DA for N EVI's. Send F frames from traffic generator to R1 with different SA and DA for N EVI's.so that 2F mac address will be learned in DUT. There is a bi directional traffic flow with F pps in each direction.Keep the setup up and running for 24 hrs,take hourly CPU utilization,memory usage.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

Take the hourly reading of CPU,process memory.There should not be any leak,crashes,CPU spikes.

PBB-EVPN To Scale the DUT to N PBB-EVPN instances in DUT with traffic and run the set up for 24hrs

Objective:

The main purpose of the test the DUT performance to scale N EVI's with traffic. Then clear bgp neighbor. There should not be any loss of routes or any crashes. Send F frames from CE to DUT from traffic generator with different SA and DA for N EVI's. Send F frames from traffic generator to R1 with different SA and DA for N EVI's.so that 2F mac address will be learned in DUT. There is a bi directional traffic flow with F pps in each direction.Keep the setup up and running for 24 hrs,take hourly CPU utilization,memory usage.

Procedure:

Configure PBB-EVPN instances in R1,MHPE2,DUT.All 4 routers except CE are running mpls,bgp,RR is acting as route reflector to R1,MHPE2 and DUT.Once the bgp comes up check the DUT evpn table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) to send the traffic to the routers.

Measurement:

Take the hourly reading of CPU,process memory.There should not be any leak,crashes,CPU spikes.

8. Acknowledgements

We would like to thank Fioccola Giuseppe of Telecom Italia reviewing our draft and commenting it.

9. IANA Considerations

This memo includes no request to IANA.

10. Security Considerations

There is no additional consideration from RFC 6192.

11 References

11.1 Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, June 1997,<http://www.rfc-editor.org/info/rfc2119>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544,DOI 10.17487/RFC2544, June 1999,<http://www.rfc-editor.org/info/rfc2544>.

[RFC2889] R.Mandeville and J. Perser "Benchmarking Methodology for LAN Switching Devices"

11.2 Informative References

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A.,Uttaro, J., Drake, J., and W.Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <http://www.rfc-editor.org/info/rfc7432>.

[RFC7623] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A.,J., Drake, J., and W. Henderickx, " Provider Backbone Bridging Combined with Ethernet VPN(PBB-EVPN)",RFC 7623,10.17487/RFC7623,September 2015 <http://www.rfc-editor.org/info/rfc7623>.

Authors' Addresses

Appendix A. Appendix

Authors' Addresses

Sudhin Jacob (editor)
Juniper Networks
Bangalore
India

Phone: +91 8061212543
Email: sjacob@juniper.net
sudhinjacob@rediffmail.com

Kishore Tiruveedhula
Juniper Networks
10 Technology Park Dr
Westford, MA 01886
USA

Phone: +1 9785898861
Email: kishoret@juniper.net

Internet Engineering Task Force S. Jacob, Ed.
Internet-Draft K. Tiruveedhula
Intended status: Standards Track Juniper Networks
Expires: December 26, 2018 June 24, 2018

Benchmarking Methodology for EVPN and PBB-EVPN
draft-kishjac-bmwg-evpntest-10

Abstract

This document defines methodologies for benchmarking EVPN and PBB-EVPN performance. EVPN is defined in RFC 7432, and is being deployed in Service Provider networks. Specifically this document defines the methodologies for benchmarking EVPN/PBB-EVPN convergence, data plane performance, and control plane performance.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 26, 2018.

Table of Contents

1. Introduction

EVPN is defined in RFC 7432, and describes BGP MPLS- based Ethernet VPNs (EVPN). PBB-EVPN is defined in RFC 7623, discusses how Ethernet Provider backbone Bridging can be combined with EVPNs to provide a new/combined solution. This draft defines methodologies that can be used to benchmark both RFC 7432 and RFC 7623 solutions. Further, this draft provides methodologies for benchmarking the performance of EVPN data and control planes, MAC learning, MAC flushing, MAC ageing, convergence, high availability, and scale.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Terminologies

MHPE Multi homed Provide Edge router.

RR Route Reflector.

P Provider Router.

CE Customer Router/Devices/Switch.

MHPE2 Multi homed Provider Edge router 2.

MHPE1 Multi homed Provider Edge router 1.

SHPE3 Single homed Provider Edge Router 3.

AA EVPN Terminologies AA All-Active.

SA EVPN Terminologies SA Single-Active.

RT Router Tester.

Sub Interface Each physical Interfaces is subdivided in to Logical units.

EVI EVPN Instances which will be running on sub interface or physical port of the provider Edge routers.

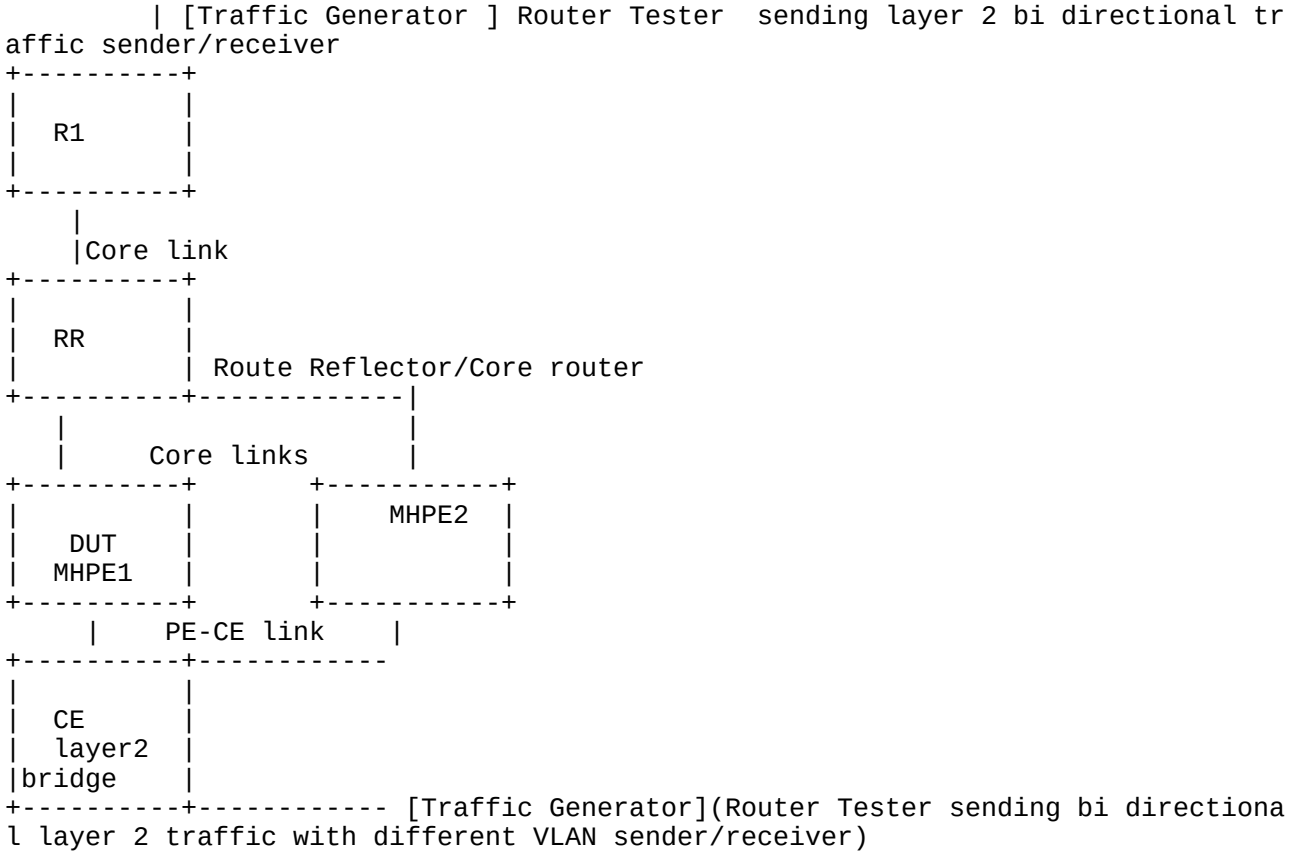
DF Designated Forwarder.

ESI Ethernet Segment Identifier.

2. Test Topology

EVPN/PBB-EVPN Services running on R1, MHPE1 and MHPE2 in Single Active Mode:

Topology Diagram



Topology Diagram

Figure 1

There are five routers in the topology. R1, RR/P, MHPE1 and MHPE2 emulating a service provider network. CE is a customer device connected to MHPE1 and MHPE2, it is configured with bridge domains in different vlans. The Router tester is connected to R1 and CE which send layer 2 traffic for configured vlans. MHPE1,MHPE2,RR/P,R1 run MPLS. The EVPN/PBB-EVPN services are running on MHPE1,MHPE2 and R1. The MHPE1 acts DUT. The RT will act as sender and receiver. The measurement will be taken in DUT.

3. Test Cases

The following tests are conducted to measure the time taken to learn the "X" number of MAC's locally in EVI . The data plane learning of MAC will happen locally from connected interface. The control plane learning of MAC is through BGP advertisements from the remote PE(SHPE3). The control plane learning of "X" MAC. The data plane MAC learning can be measured using the parameters defined in RFC 2889 section 5.8.

3.1. How long it takes to learn local mac address in EVPN

Objective:

To Record the time taken to learn the MAC address locally in DUT.

Procedure:

Configure EVPN EVI in R1, MHPE2, DUT. All 4 routers except CE are running MPLS, BGP, RR is acting as route reflector to R1,MHPE2 and DUT. For MH PE ESI must be configured per IFD/Interface. Using RT (traffic generator)to send the traffic to the CE. The traffic is unidirectional. Since CE is working in bridge mode, frames will be send to ingress sub interface of DUT. The BGP must be established in R1, MHPE1(DUT), RR, MHPE2. Send "X" unicast frames from CE to MHPE1(DUT) working in SA mode with "X" different source and destination address from RT. The DUT must learn these "X" macs in data plane.

Measurement :

Measure the time taken to learn "X" MACs in DUT evpn mac table. The data plane measurement is taken by considering DUT as black box the range of X MAC is known from RT and the same must be learned in DUT, the time taken to learn "X" macs is measured.

Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The mac learning time is calculated by averaging the values obtained from "N" samples.

Mac learning in sec = (T1+T2+..Tn/N)

3.2. How long it takes to learn local mac address in PBB EVPN

Objective:

To Record the time taken to learn the MAC address locally.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running MPLS, BGP, RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP comes up. Record the DUT PBB-EVPN table. For MH PE ESI must be configured per IFD/Interface. From RT (traffic generator) send the traffic to the DUT. The BGP must be established in R1,MHPE1 (DUT), RR, MHPE2. The traffic is unidirectional. Since CE is working in bridge mode, frames will be send to ingress sub interface of DUT.

Send "X" unicast frames from CE to MHPE1(DUT) working in SA mode with "X" different source and destination address from RT. The DUT must learn "X" macs in data plane.

Measurement :

Measure the time taken by the DUT to learn the "X" MACs in the data plane. The data plane measurement is taken by considering DUT as black box the range of "X" MAC is known from RT and the same must be learned in DUT, the time taken to learn "X" MAC is measured. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The mac learning time is calculated by averaging the values obtained from "N" samples.

Mac learning in sec = (T1+T2+..Tn/N)

3.3. How long it takes to learn the remote macs

Objective:

To Record the time taken to learn the remote macs.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/ Interface. Using RT(traffic generator) send the traffic to R1.The traffic is uni directional. There wont be any traffic flow from CE to DUT during this test. The BGP must be in established state. The MACS learned in R1 will be advertised to DUT by BGP.

Send X frames with X different SA and DA to R1 from RT. R1 will advertise these locally learned macs to MHPE1 and MHPE2 via control plane.Measure the time taken to learn these X MACs from remote peer in DUT EVPN MAC address table.The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken by the DUT to learn the "X" MACs in the data plane.Repeat these test and plot the data.The test is repeated for "N" times and the values are collected.The mac learning time is calculated by averaging the values obtained from "N" samples.

Mac learning in sec = (T1+T2+..Tn/N)

3.4. PBB-EVPN How long it takes to learn the mac from remote peer

Objective:

To Record the time taken to learn the remote macs.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Record the DUT PBB-EVPN table. For MHPE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to R1. The traffic is uni directional.There wont be any traffic flow from CE to DUT during this test.The BGP must be in established state.

Send X frames with X different SA and DA to R1 from RT.These macs will be flooded to MHPE1 and MHPE2 by R1.The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to learn X mac address in DUT mac table. Repeat these test and plot the data.The test is repeated for "N" times and the values are collected.The mac learning time is calculated by averaging the values obtained by "N" samples.

Mac learning in sec = (T1+T2+..Tn/N)

3.5. How long it takes to flush the local macs due to CE link flap and measure the relearning rate of MACs

Objective:

To record the time taken to flush the mac learned locally and the time taken to relearn the same amount of macs.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic. In this scenario traffic will be only send from CE side.

Send X frames with X different SA and DA to DUT from CE using traffic generator. Wait till the MHPE1 learns all X MAC address. Then fail the MHPE1 CE link and measure the time taken to flush these X MACs from the EVPN MAC table. Bring up the link which was made Down(the link between MHPE1 and CE).Measure time taken to relearn it. The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken for flushing these X MAC address. Measure the time taken to relearn the X MACs in DUT. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The flush and the relearning time is calculated by averaging the values obtained by "N" samples.

Flush time for X Macs in sec = (T1+T2+..Tn/N) Relearning time for X macs in sec = (T1+T2+..Tn/N)

3.6. PBB-EVPN how long it takes to flush the local macs and measure the relearning rate of macs during PE-CE link flap

Objective:

To record the time taken to flush the mac learned locally and the time taken to relearn the same amount of macs.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT PBB-EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the CE. The traffic is uni directional.

Send X frames with X different SA and DA to DUT from CE using traffic generator. Wait till the MHPE1 learn all X MAC address. Then fail the MHPE1 CE link and measure the time taken to flush these X MACs from the PBB-EVPN MAC table. Then bring up the link. Measure the time taken to relearn X MACS. The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken for flushing these X MAC address. Measure the time taken to relearn the X MACs in DUT. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The flush and the relearning time is calculated by averaging the values obtained by "N" samples.

Flush time for X Macs in sec = (T1+T2+..Tn/N) Relearning time for X macs in sec = (T1+T2+..Tn/N)

3.7. How long it takes to flush the remote macs, due to remote link failure.

Objective:

To record the time taken to flush the remote mac learned in DUT during remote link failure.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established Record the DUT EVPN table. For MHPE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to R1.There wont be any traffic flowing to CE from RT.

Send X frames with X different SA and DA to DUT from R1 using traffic generator. Bring down the link between R1 and traffic generator. Then measure the time taken to flush the DUT EVPN MAC table. The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to flush X remote MACs from EVPN MAC table of DUT. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The flush rate is calculated averaging the values obtained by "N" samples.

Flush time for X Macs in sec = (T1+T2+..Tn/N)

3.8. PBB-EVPN How long it takes to flush the remote macs due to remote link failure

Objective:

To record the time taken to flush the remote mac learned in DUT during remote link failure.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established Record the DUT PBB-EVPN MAC table. For MHPE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to R1.In this scenario traffic will be flowing only from R1.

Send X frames with X different SA and DA to DUT from R1 using traffic generator. Bring down the link between R1 and traffic generator. Then measure the time taken to flush the DUT PBB-EVPN MAC address table. The remote MACs will be learned by Data plane, but the B-MAC will be learned by control plane. The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to flush X remote MACs from PBB-EVPN MAC table of DUT. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The flush rate is calculated by averaging the values obtained by "N" samples.

Flush time for X Macs in sec = (T1+T2+..Tn/N)

3.9. To measure the MAC aging time.

Objective:

To measure the mac aging time.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT .Once the BGP is established. Record the DUT EVPN table. For MHPE ESI must be configured per IFD/Interface. Using RT(traffic generator), send the traffic to the DUT. The traffic will be flowing from CE to DUT. There wont be any traffic from R1.

Send X frames with X different SA and DA to DUT from CE using traffic generator. Wait till X MAC address are learned. Then stop the traffic. Record the time taken to flush X MACS from DUT EVPN MAC table due to aging. The DUT and MHPE2 are running SA mode

Measurement :

Measure the time taken to flush X MAC address due to aging. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The aging is calculated averaging the values obtained by "N" samples.

Aging time for X Macs in sec = (T1+T2+..Tn/N)

3.10. PBB-EVPN To measure the MAC aging time.

Objective:

To measure the mac aging time.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT.Once the BGP is established. Record the DUT PBB-EVPN MAC table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the DUT. The traffic is unidirectional flowing from CE to DUT.

Send X frames with X different SA and DA to DUT from CE using traffic generator. Wait till X MAC address are learned in DUT PBB- EVPN MAC table. Then stop the traffic. Record the time taken to flush X MAC entries due to aging. The DUT and MHPE2 running in SA mode

Measurement :

Measure the time taken to flush X MAC address due to aging. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The aging is calculated by averaging the values obtained by "N" samples.

Aging time for X Macs in sec = (T1+T2+..Tn/N)

3.11. How long it takes to age out the remote macs

Objective:

To measure the remote mac aging time.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT.

Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to R1.There wont be any traffic from CE.

Send X frames with X different SA and DA to DUT from R1 using traffic generator. Stop the traffic at remote PE R1.Due to MAC aging R1 will withdraw its routes from DUT and MHPE2. Measure the time taken to remove these MACs from DUT EVPN MAC table. DUT and MHPE2 are running in SA mode

Measurement :

Measure the time taken to flush X remote MACs learned in DUT EVPN MAC table due to aging. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The aging is calculated by averaging the values obtained by "N" samples.

Aging time for X Macs in sec = (T1+T2+..Tn/N)

3.12. PBB-EVPN How long it takes to age out the remote macs.

Objective:

To measure the remote mac aging time.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT MAC table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic Generator) send the traffic to R1. There is no traffic from CE side.

Send X frames with X different SA and DA to DUT from R1 using traffic generator. Stop the traffic at remote PE(R1).Measure the time taken to remove these remote MACs from DUT PBB-EVPN MAC table. The DUT and MHPE2 are running in SA mode.

Measurement :

Measure the time taken to flush the X remote MACs from DUT PBB-EVPN MAC table due to aging Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The aging is calculated by averaging the values obtained by "N" samples.

Aging time for X Macs in sec = (T1+T2+..Tn/N)

3.13. How long it takes to learn both local and remote macs.

Objective:

To record the time taken to learn both local and remote macs.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the routers. The traffic is bi directional.

Send X frames with X different SA and DA to DUT from R1 using traffic generator. Send X frames with different SA and DA from traffic generator connected to CE. The SA and DA of flows must be complimentary to have unicast flows. Measure the time taken by the DUT to learn 2X in EVPN MAC. DUT and MHPE2 are running in SA mode.

Measurement :

Measure the time taken to learn 2X MAC address in DUT EVPN MAC table. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The mac learning time is calculated by averaging the values obtained by "N" samples.

Time to learn 2X Macs in sec = (T1+T2+..Tn/N)

3.14. PBB-EVPN How long it takes to learn both local and remote macs

Objective:

To record the time taken to learn both local and remote macs.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table.For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the routers.

Send X frames with X different SA and DA to DUT from R1 using traffic generator. Send X frames with different SA and DA from traffic generator connected to CE. The SA and DA of flows must be

complimentary to have unicast flows. Measure the time taken by the DUT to learn 2X in MAC table. DUT and MHPE2 are running in SA mode.

Measurement :

Measure the time taken to learn 2X MAC address table in DUT PBB-EVPN MAC table. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The mac learning time is calculated by averaging the values obtained by "N" samples.

Time to learn 2X Macs in sec = (T1+T2+..Tn/N)

4. High Availability

4.1. To Record the whether there is traffic loss due to routing engine failover for redundancy test.

Objective:

To record traffic loss during routing engine failover.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) Send bi directional to the routers.

Send X frames from CE to DUT from traffic generator withX different SA and DA. Send X frames from traffic generator to R1 with X different SA and DA so that 2X MAC address will be learned in DUT. There is a bi directional traffic flow with X pps in each direction. Then do a routing engine fail-over.

Measurement :

There should be 0 traffic loss which is the ideal case, No change in the DF role. DUT should not withdraw any routes.Repeat the test "N" times and plot the data.The packet loss is calculated by averaging the values obtained from "N" samples.

Packet loss in sec = (T1+T2+..Tn/N)

4.2. PBB-EVPN To Record the whether there is traffic loss due to routing engine failover for redundancy test

Objective:

To record traffic loss during routing engine failover.

Procedure:

Configure PBB-EVPN EVI in R1,MHPE2,DUT. All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT PBB-EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the routers.

Send X frames to DUT with X different SA and DA from CE using the traffic generator. Send X frames from traffic generator to R1 with X different SA and DA so that 2X MAC address will be Learned in DUT. There is a bi directional traffic flow with X pps in each direction. Then do a routing engine fail-over.

Measurement :

There should be 0 traffic loss which is the ideal case, No change in the DF role. DUT should not withdraw any routes.Repeat the test "N" times and plot the data.The packet loss is calculated by averaging the values obtained from "N" samples.

Packet loss in sec = (T1+T2+..Tn/N)

5. ARP/ND Scale

These tests are conducted to Record the scaling parameter of ARP/ND of the DUT.

5.1. To find ARP/ND scale

Objective:

To Record the ARP/ND scale of the DUT.

Procedure:

Configure EPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send arp/ICMPv6 request to the DUT which has gateway configured.

Send X arp/icmpv6 request from RT to DUT with different sender ip/ ipv6 address to the same target gateway ip address. Measure whether X MAC+IPv4 address/MAC+IPv6 address of the hosts are learned in DUT.

Measurement :

The DUT must learn X MAC+IPV4/MAC+IPv6 and it must advertise the X MAC+IPV4/MAC+IPV6 to the remote router.

6. Scale

This is to measure the performance of DUT in scaling to "X" EVPN instances. The measured parameters are CPU usage, memory leak,crashes.

6.1. To Measure the scale limit of DUT with trigger (Scale without traffic)

Objective:

To measure the scale limit of DUT for EVPN.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MHPE,DUT ESI must be configured per IFD/Interface.

The DUT,MHPE2 and R1 are scaled to "N" EVI.Clear BGP neighbors of the DUT. Once adjacency is established in the DUT. Measure the routes received from MHPE2 and R1 for "N" EVI in the DUT.

Measurement :

There should not be any loss of route types 1,2,3 and 4 in DUT. DUT must relearn all type 1,2,3 and 4 from remote routers. The DUT must be subjected to various values of N to find the optimal scale limit

6.2. PBB-EVPN To measure the scale limit with trigger.

Objective:

To measure the scale limit of DUT for PBB-EVPN.

Procedure:

Configure "N" PBB-EVPN instances in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting route reflector to R1,MHPE2 and DUT. Once BGP is established. Record the DUT PBB-EVPN table.For MHPE ESI must be configured on IFD/Interface.

The DUT,MHPE2 and R1 are scaled to "N" PBB-EVPN instances. Clear BGP neighbors in the DUT Once adjacency is established in DUT, check routes received from R1 and MHPE2.

Measurement :

There should not be any loss of route types 2,3 and 4 in DUT. The DUT must relearn all type 2,3 and 4 routes from remote routers. The DUT must be subjected to various values of N to find the optimal scale limit.

6.3. To measure the convergence time of DUT with scale and traffic.

Objective:

To measure the convergence time of DUT when the DUT is scaled with EVPN instance along with traffic.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator)send the traffic to the routers.

Scale N EVIs in DUT,R1 and MHPE2.Send F frames to DUT from CE using traffic generator with X different SA and DA for N EVI’s. Send F frames from traffic generator to R1 with X different SA and DA. There will be 2X number of MAC address will be learned in DUT EVPN MAC table. There is a bi directional traffic flow with F pps in each direction. Then clear the BGP neighbors in the DUT. Once the adjacency is restored in DUT. Measure the time taken to learn 2X MAC address in DUT MAC table.

Measurement :

The DUT must learn 2X MAC address. Measure the time taken to learn 2X MAC in DUT. Repeat these test and plot the data.The test is repeated for "N" times and the values are collected.The convergence time is calculated by averaging the values obtained by "N" samples.

Convergence time in sec = (T1+T2+..Tn/N)

6.4. .PBB-EVPN To measure the convergence time of DUT with scale and traffic.

Objective:

To measure the convergence time of DUT when the DUT is scaled with PBB-EVPN instance along with traffic.

Procedure:

Configure PBB-EVPN instances in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the routers.

Scale N PBB-EVI's in DUT,R1 and MHPE2.Send F frames to DUT from CE using traffic generator with X different SA and DA for N EVI's. Send F frames from traffic generator to R1 with X different SA and DA. There will be 2X number of MAC address will be learned in DUT PBB-EVPN MAC table. There is a bi directional traffic flow with F pps in each direction. Then clear the BGP neighbors in the DUT. Once the adjacency is restored in DUT. Measure the time taken to learn 2X MAC address in DUT PBB-MAC table.

Measurement :

The DUT must learn 2X MAC address. Measure the time taken to learn 2X MAC in DUT. Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The convergence time is calculated by averaging the values obtained by "N" samples.

Convergence time in sec = (T1+T2+..Tn/N)

7. SOAK Test

This is measuring the performance of DUT running with scaled configuration with traffic over a peroid of time "T'". In each interval "t1" the parameters measured are CPU usage, memory usage, crashes.

7.1. To Measure the stability of the DUT with scale and traffic.

Objective:

To measure the stability of the DUT in a scaled environment with traffic.

Procedure:

Configure EVPN EVI in R1,MHPE2,DUT.All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP is established. Record the DUT EVPN table. For MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator) send the traffic to the routers.

Scale N EVI's in DUT,R1 and MHPE2.Send F frames to DUT from CE using traffic generator with different X SA and DA for N EVI's. Send F frames from traffic generator to R1 with X different SA and DA. There will be 2X number of MAC address will be learned in DUT EVPN MAC table. There is a bi directional traffic flow with F pps in each direction. The DUT must run with traffic for 24 hours, every hour check for memory leak, crash.

Measurement :

Take the hourly reading of CPU, process memory. There should not be any leak, crashes, CPU spikes.

7.2. PBB-EVPN to measure the stability of DUT with scale and traffic.

Objective:

To measure the stability of the DUT in a scaled environment with traffic.

Procedure:

Configure N PBB-EVPN instances in R1, MHPE2, DUT. All 4 routers except CE are running MPLS,BGP,RR is acting as route reflector to R1,MHPE2 and DUT. Once the BGP comes up Record the DUT EVPN table. for MH PE ESI must be configured per IFD/Interface. Using RT(traffic generator)send the traffic to the routers.

Scale N PBB-EVI's in DUT,R1 and MHPE2.Send F frames to DUT from CE using traffic generator with X different SA and DA for N EVI's. Send F frames from traffic generator to R1 with X different SA and DA. There will be 2X number of MAC address will be learned in DUT PBB-EVPN MAC table. There is a bi directional traffic flow with F pps in Each direction. The DUT must run with traffic for 24 hours, every hour check the memory leak, crashes.

Measurement :

Take the hourly reading of CPU process, memory usages. There should not be any memory leak, crashes,CPU spikes.

8. Acknowledgements

We would like to thank Fioccola Giuseppe of Telecom Italia reviewing our draft and commenting it. We would like to thank Sarah Banks for guiding and mentoring us.

9. IANA Considerations

This memo includes no request to IANA.

10. Security Considerations

There is no additional consideration from RFC 6192.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <https://www.rfc-editor.org/info/rfc2544>.

[RFC2899] Ginoza, S., "Request for Comments Summary RFC Numbers 2800-2899", RFC 2899, DOI 10.17487/RFC2899, May 2001, <https://www.rfc-editor.org/info/rfc2899>.

11.2. Informative References

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RFC7623] Sajassi, A., Ed., Salam, S., Bitar, N., Isaac, A., and W. Henderickx, "Provider Backbone Bridging Combined with Ethernet VPN (PBB-EVPN)", RFC 7623, DOI 10.17487/RFC7623, September 2015, <https://www.rfc-editor.org/info/rfc7623>.

Appendix A. Appendix

Authors' Addresses

Sudhin Jacob (editor)
Juniper Networks
Bangalore
India

Phone: +91 8061212543
Email: sjacob@juniper.net

Kishore Tiruveedhula
Juniper Networks
10 Technology Park Dr
Westford, MA 01886
USA

Phone: +1 9785898861
Email: kishoret@juniper.net

Network Working Group
Internet-Draft
Updates: 2544 (if approved)
Intended status: Informational
Expires: May 3, 2018

A. Morton
AT&T Labs

October 30, 2017

Updates for the Back-to-back Frame Benchmark in RFC 2544
draft-morton-bmwg-b2b-frame-00

Abstract

Fundamental Benchmarking Methodologies for Network Interconnect Devices of interest to the IETF are defined in RFC 2544. This memo updates the provisions of the test to measure the Back-to-back frames Benchmark of RFC 2544, based on further experience.

This memo updates Section 26.4 of RFC 2544.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 3, 2018.

Copyright Notice

Table of Contents

1. Introduction

The IETF's fundamental Benchmarking Methodologies are defined in[RFC2544], supported by the terms and definitions in [RFC1242], and [RFC2544] actually obsoletes an earlier specification, [RFC1944]. Over time, the benchmarking community has updated [RFC2544] several times, including the Device Reset Benchmark [RFC6201], and the important Applicability Statement [RFC6815] concerning use outside the Isolated Test Environment (ITE) required for accurate benchmarking. Other specifications implicitly update [RFC2544], such as the IPv6 Benchmarking Methodologies in [RFC5180].

Recent testing experience with the Back-to-back Frame test and Benchmark in Section 26.4 of [RFC2544] indicates that an update is warranted [OPNFV-2017] [VSPERF-b2b]. This memo describes the rationale and provides the updated method.

[RFC2544] provides its own Requirements Language consistent with [RFC2119], since [RFC1944] predates [RFC2119]. Thus, the requirements presented in this memo are expressed in [RFC2119] terms, and intended for those performing/reporting laboratory tests to improve clarity and repeatability, and for those designing devices that facilitate these tests.

2. Scope and Goals

The scope of this memo is to define an updated method to unambiguously perform tests, measure the benchmark(s), and report the results for Back-to-back Frames (presently described Section 26.4 of [RFC2544]).

The goal is to provide more efficient test procedures where possible, and to expand reporting with additional interpretation of the results.

[RFC2544] Benchmarks rely on test conditions with constant frame sizes, with the goal of understanding what network device capability has been tested. Tests with the smallest size stress the header processing capacity, and tests with the largest size stress the overall bit processing capacity. Tests with sizes in-between may determine the transition between these two capacities. However, conditions simultaneously sending multiple frame sizes, such as those described in [RFC6985], MUST NOT be used in Back-to-back Frame testing.

3. Motivation

Section 3.1 of [RFC1242] describes the rationale for the Back-to-back Frames Benchmark. To summarize, there are several reasons that devices on a network produce bursts of frames at the minimum allowed spacing, and it is therefore worthwhile to understand the Device Under Test (DUT) limit on the length of such bursts in practice. Also, [RFC1242] states:

> "Tests of this parameter are intended to determine the extent of data buffering in the device."

After this test was defined, there have been occasional discussions of the stability and repeatability of the results, both over time and across labs. Fortunately, the Open Platform for Network Function Virtualization (OPNFV) VSPERF project’s Continuous Integration (CI) testing routinely repeats Back-to-back Frame tests to verify that test functionality has been maintained through development of the test control programs. These tests were used as a basis to evaluate

stability and repeatability, even across lab set-ups when the test platform was migrated to new DUT hardware at the end of 2016.

When the VSPERF CI results were examined [VSPERF-b2b], several aspects of the results were considered notable:

1. Back-to-back Frame Benchmark was very consistent for some fixed frame sizes, and somewhat variable for others.

2. The Back-to-back Frame length reported for large frame sizes was unexpectedly long, and no explanation or measurement limit condition was indicated.

3. Calculation of the extent of buffer time in the DUT helped explain the results with all frame sizes (some frame sizes cannot exceed the frame header processing rate of the DUT, and therefore no buffering occurs).

4. It was observed that the actual buffer time in the DUT could be estimated using results from the Throughput tests conducted according to Section 26.1 of [RFC2544].

Further, if the Throughput tests of Section 26.1 of [RFC2544] are conducted as a pre-requiste test, the number of frame sizes required for Back-to-back Frame Benchmarking can be reduced to one or more of the small frame sizes, or results for large frame sizes can be noted as invalid in the results.

[VSPERF-b2b] provides the details of the calculation to estimate the actual buffer time available in the DUT, using results from the Throughput tests for each frame size, and the maximum theoretical frame rate for the DUT links (which constrain the minimum frame spacing).

4. Pre-Requisites

The Test Setup MUST be consistent with Figure 1 of [RFC2544], or Figure 2 when the tester’s sender and reciver are different devices. Other mandatory testing aspects described in [RFC2544] MUST be included, unless explicitly modified in the next section.

The ingress and egress link speeds and link layer protocols MUST be specified and used to compute the maximum theoretical frame rate when respecting the minimum inter-frame gap.

The test results for the Throughput Benchmark conducted according to Section 26.1 of [RFC2544] for all [RFC2544]-RECOMMENDED frame sizes MUST be available to reduce the tested frame size list, or to note

invalid results for individual frame sizes (because the burst length may be infinite for large frame sizes).

Note that:

- the Throughput and the Back-to-back Frame measurement configuration traffic characteristics (unidirectional or bi-directional) MUST match.

- the Throughput measurement MUST be under zero-loss conditions, according to Section 26.1 of [RFC2544].

The Back-to-back Benchmark described in Section 3.1 of [RFC1242] MUST be measured directly by the tester. Additional measurement reuirements are described below in Section 5.

5. Back-to-back Frames

Objective: To characterize the ability of a DUT to process back-to-back frames as defined in [RFC1242].

The Procedure follows.

5.1. Preparing the list of Frame sizes

From the list of RECOMMENDED Frame sizes (Section 9 of [RFC2544]), select the subset of Frame sizes whose measured Throughput was less than the maximum theoretical Frame Rate. Only these Frame sizes make it possible to produce a burst of frames that cause the DUT buffers to fill and eventually overflow, producing one or more discarded frames.

5.2. Test for a Single Frame Size

Each trial in the test requires the tester to send a burst of frames (after idle time) with the minimum inter-frame gap, and to count the frames forwarded by the DUT.

The duration of the trial MUST be at least 2 seconds, to allow DUT buffers to deplete.

If all frames have been received, the tester increases the length of the burst and performs another trial.

If the received frame count is less than the number of frames in the burst, then the limit of DUT processing and buffering may have been exceeded, and the burst length is reduced for the next trial.

@@@@ Should a particular search algorithm be included?

The Back-to-back Frame value is the longest burst of frames that the DUT can successfully process and buffer without frame loss, as determined from the series of trials. The tester may impose a (configurable) minimum step size for burst length, and the step size MUST be reported with the results (as this influences the accuracy and variation of test results).

5.3. Test Repetition

The test MUST be repeated N times for each frame size in the subset list, and each Back-to-back Frame value made available for further processing (below).

5.4. Benchmark Calculations

For each Frame size, calculate the following summary statistics for Back-to-back Frame values over the N tests:

- Average (Benchmark)
- Minimum
- Maximum
- Standard Deviation

Further, calculate the Implied DUT Buffer Time and the Corrected DUT Buffer Time in seconds, as follows:

Implied DUT Buffer Time =

Average Back-to-back Frames / Max Theoretical Frame Rate

Corrected DUT Buffer Time =

$$\text{Implied DUT Buffer Time} * \frac{\text{Measured Throughput}}{\text{Max Theoretical Frame Rate}}$$

6. Reporting

The back-to-back results SHOULD be reported in the format of a table with a row for each of the tested frame sizes. There SHOULD be columns for the frame size and for the resultant average frame count for each type of data stream tested.

The number of tests Averaged for the Benchmark, N, MUST be reported.

The Minimum, Maximum, and Standard Deviation across all complete tests SHOULD also be reported.

The Corrected DUT Buffer Time SHOULD also be reported.

If the tester operates using a maximum burst length in frames, then this maximum length SHOULD be reported.

Frame Size, octets	Ave B2B Length, frames	Min,Max,StdDev	Corrected Buff Time, Sec
64	26000	25500,27000,20	0.00004

Back-to-Back Frame Results

Static and configuration parameters:

Number of test repetitions, N

Minimum Step Size (during searches), in frames.

7. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization using controlled stimuli in a laboratory environment, with dedicated address space and the other constraints [RFC2544].

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the DUT/SUT.

Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

8. IANA Considerations

This memo makes no requests of IANA.

9. Acknowledgements

Thanks to Trevor Cooper, Sridhar Rao, and Martin Klozik of the VSPERF project for many contributions to the testing [VSPERF-b2b].

10. References

10.1. Normative References

[RFC1242] Bradner, S., "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, DOI 10.17487/RFC1242, July 1991, <https://www.rfc-editor.org/info/rfc1242>.

[RFC1944] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 1944, DOI 10.17487/RFC1944, May 1996, <https://www.rfc-editor.org/info/rfc1944>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <https://www.rfc-editor.org/info/rfc2544>.

[RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, DOI 10.17487/RFC5180, May 2008, <https://www.rfc-editor.org/info/rfc5180>.

[RFC6201] Asati, R., Pignataro, C., Calabria, F., and C. Olvera, "Device Reset Characterization", RFC 6201, DOI 10.17487/RFC6201, March 2011, <https://www.rfc-editor.org/info/rfc6201>.

[RFC6815] Bradner, S., Dubray, K., McQuaid, J., and A. Morton, "Applicability Statement for RFC 2544: Use on Production Networks Considered Harmful", RFC 6815, DOI 10.17487/RFC6815, November 2012, <https://www.rfc-editor.org/info/rfc6815>.

[RFC6985] Morton, A., "IMIX Genome: Specification of Variable Packet Sizes for Additional Testing", RFC 6985, DOI 10.17487/RFC6985, July 2013, <https://www.rfc-editor.org/info/rfc6985>.

10.2. Informative References

[OPNFV-2017] Cooper, T., "Dataplane Performance, Capacity, and Benchmarking in OPNFV", June 2017.

[VSPERF-b2b] Morton, A., "Back2Back Testing Time Series (from CI)", June 2017.

Author's Address

Al Morton
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com

Network Working Group
Internet-Draft
Updates: 2544 (if approved)
Intended status: Informational
Expires: September 3, 2019

A. Morton
AT&T Labs

March 2, 2019

Updates for the Back-to-back Frame Benchmark in RFC 2544
draft-morton-bmwg-b2b-frame-05

Abstract

Fundamental Benchmarking Methodologies for Network Interconnect Devices of interest to the IETF are defined in RFC 2544. This memo updates the procedures of the test to measure the Back-to-back frames Benchmark of RFC 2544, based on further experience.

This memo updates Section 26.4 of RFC 2544.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14[RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 3, 2019.

Table of Contents

1. Introduction

The IETF's fundamental Benchmarking Methodologies are defined in[RFC2544], supported by the terms and definitions in [RFC1242], and [RFC2544] actually obsoletes an earlier specification, [RFC1944]. Over time, the benchmarking community has updated [RFC2544] several times, including the Device Reset Benchmark [RFC6201], and the important Applicability Statement [RFC6815] concerning use outside the Isolated Test Environment (ITE) required for accurate benchmarking. Other specifications implicitly update [RFC2544], such as the IPv6 Benchmarking Methodologies in [RFC5180].

Recent testing experience with the Back-to-back Frame test and Benchmark in Section 26.4 of [RFC2544] indicates that an update is warranted [OPNFV-2017] [VSPERF-b2b]. This memo describes the rationale and provides the updated method.

[RFC2544] provides its own Requirements Language consistent with [RFC2119], since [RFC1944] predates [RFC2119]. Thus, the requirements presented in this memo are expressed in [RFC2119] terms, and intended for those performing/reporting laboratory tests to improve clarity and repeatability, and for those designing devices that facilitate these tests.

2. Scope and Goals

The scope of this memo is to define an updated method to unambiguously perform tests, measure the benchmark(s), and report the results for Back-to-back Frames (presently described Section 26.4 of [RFC2544]).

The goal is to provide more efficient test procedures where possible, and to expand reporting with additional interpretation of the results. The tests described in this memo address the cases where the maximum frame rate of a single ingress port cannot be transferred to an egress port loss-free (for some frame sizes of interest).

[RFC2544] Benchmarks rely on test conditions with constant frame sizes, with the goal of understanding what network device capability has been tested. Tests with the smallest size stress the header processing capacity, and tests with the largest size stress the overall bit processing capacity. Tests with sizes in-between may determine the transition between these two capacities. However, conditions simultaneously sending multiple frame sizes, such as those described in [RFC6985], MUST NOT be used in Back-to-back Frame testing.

Section 3 of [RFC8239] describes buffer size testing for physical networking devices in a Data Center. The [RFC8239] methods measure buffer latency directly with traffic on multiple ingress ports that overload an egress port on the Device Under Test (DUT), and are not subject to the revised calculations presented in this memo.

3. Motivation

Section 3.1 of [RFC1242] describes the rationale for the Back-to-back Frames Benchmark. To summarize, there are several reasons that devices on a network produce bursts of frames at the minimum allowed spacing, and it is therefore worthwhile to understand the Device

Under Test (DUT) limit on the length of such bursts in practice. Also, [RFC1242] states:

> "Tests of this parameter are intended to determine the extent of data buffering in the device."

After this test was defined, there have been occasional discussions of the stability and repeatability of the results, both over time and across labs. Fortunately, the Open Platform for Network Function Virtualization (OPNFV) VSPERF project's Continuous Integration (CI) testing routinely repeats Back-to-back Frame tests to verify that test functionality has been maintained through development of the test control programs. These tests were used as a basis to evaluate stability and repeatability, even across lab set-ups when the test platform was migrated to new DUT hardware at the end of 2016.

When the VSPERF CI results were examined [VSPERF-b2b], several aspects of the results were considered notable:

1. Back-to-back Frame Benchmark was very consistent for some fixed frame sizes, and somewhat variable for others.

2. The Back-to-back Frame length reported for large frame sizes was unexpectedly long, and no explanation or measurement limit condition was indicated.

3. Calculation of the extent of buffer time in the DUT helped to explain the results observed with all frame sizes (for example, some frame sizes cannot exceed the frame header processing rate of the DUT and therefore no buffering occurs, therefore the results depended on the test equipment and not the DUT).

4. It was found that the actual buffer time in the DUT could be estimated using results from the Throughput tests conducted according to Section 26.1 of [RFC2544], because it appears that the DUT's frame processing rate may tend to increase the estimate.

Further, if the Throughput tests of Section 26.1 of [RFC2544] are conducted as a prerequisite test, the number of frame sizes required for Back-to-back Frame Benchmarking can be reduced to one or more of the small frame sizes, or the results for large frame sizes can be noted as invalid in the results if tested anyway (these are the frame sizes for which the back-to-back frame rate cannot exceed the exceed the frame header processing rate of the DUT and no buffering occurs).

[VSPERF-b2b] provides the details of the calculation to estimate the actual buffer storage available in the DUT, using results from the

Throughput tests for each frame size, and the maximum theoretical frame rate for the DUT links (which constrain the minimum frame spacing). We present some of these details here.

The simplified model used in these calculations for the DUT includes a packet header processing function with limited rate of operation, as shown below:

```
                        |------------ DUT --------|
Generator -> Ingress -> Buffer -> HeaderProc -> Egress -> Receiver
```

So, in the back2back frame testing:

1. The Ingress burst arrives at Max Theoretical Frame Rate, and initially the frames are buffered

2. The packet header processing function (HeaderProc) operates at approximately the "Measured Throughput", removing frames from the buffer

3. Frames that have been processed are clearly not in the buffer, so the Corrected DUT buffer time equation (Section 5.4) estimates and removes the frames that the DUT forwarded on Egress during the burst.

Knowledge of approximate buffer storage size (in time or bytes) may be useful to estimate whether frame losses will occur if DUT forwarding is temporarily suspended in a production deployment, due to an unexpected interruption of frame processing (an interruption of duration greater than the estimated buffer would certainly cause lost frames).

The presentation of OPNFV VSPERF evaluation and development of enhanced search alogorithms [VSPERF-BSLV] was discussed at IETF-102. The enhancements are intended to compensate for transient inerrrupts that may cause loss at near-Throughput levels of offered load. Subsequent analysis of the results indicates that buffers within the DUT can compensate for some interrupts, and this finding increases the importance of the Back-to-back frame characterization described here.

4. Prerequisites

The Test Setup MUST be consistent with Figure 1 of [RFC2544], or Figure 2 when the tester’s sender and recover are different devices. Other mandatory testing aspects described in [RFC2544] MUST be included, unless explicitly modified in the next section.

The ingress and egress link speeds and link layer protocols MUST be specified and used to compute the maximum theoretical frame rate when respecting the minimum inter-frame gap.

The test results for the Throughput Benchmark conducted according to Section 26.1 of [RFC2544] for all [RFC2544]-RECOMMENDED frame sizes MUST be available to reduce the tested frame size list, or to note invalid results for individual frame sizes (because the burst length may be essentially infinite for large frame sizes).

Note that:

- the Throughput and the Back-to-back Frame measurement configuration traffic characteristics (unidirectional or bi-directional) MUST match.

- the Throughput measurement MUST be under zero-loss conditions, according to Section 26.1 of [RFC2544].

The Back-to-back Benchmark described in Section 3.1 of [RFC1242] MUST be measured directly by the tester. Additional measurement requirements are described below in Section 5.

5. Back-to-back Frames

Objective: To characterize the ability of a DUT to process back-to-back frames as defined in [RFC1242].

The Procedure follows.

5.1. Preparing the list of Frame sizes

From the list of RECOMMENDED Frame sizes (Section 9 of [RFC2544]), select the subset of Frame sizes whose measured Throughput was less than the maximum theoretical Frame Rate. These are the only Frame sizes where it is possible to produce a burst of frames that cause the DUT buffers to fill and eventually overflow, producing one or more discarded frames.

5.2. Test for a Single Frame Size

Each trial in the test requires the tester to send a burst of frames (after idle time) with the minimum inter-frame gap, and to count the corresponding frames forwarded by the DUT.

The duration of the trial MUST be at least 2 seconds, to allow DUT buffers to deplete.

If all frames have been received, the tester increases the length of the burst according to the search algorithm and performs another trial.

If the received frame count is less than the number of frames in the burst, then the limit of DUT processing and buffering may have been exceeded, and the burst length is determined by the search algorithm for the next trial.

Classic search algorithms have been adapted for use in benchmarking, where the search requires discovery of a pair of outcomes, one with no loss and another with loss, at load conditions within the acceptable tolerance. Also for conditions encountered when benchmarking the Infrastructure for Network Function Virtualization require algorithm enhancement. Fortunately, the adaptation of Binary Search, and an enhanced Binary Search with Loss Verification have been specified in [TST009]. These alogorithms (see clause 12.3) can easily be used for Back-to-back Frame benchmarking by replacing the Offered Load level with burst length in frames. [TST009] Annex B describes the theory behind the enhanced Binary Search algorithm.

Either the [TST009] Binary Search or Binary Search with Loss Verification algorithms MUST be used, and input parameters to the algorithm(s) MUST be reported.

The Back-to-back Frame value is the longest burst of frames that the DUT can successfully process and buffer without frame loss, as determined from the series of trials. The tester may impose a (configurable) minimum step size for burst length, and the step size MUST be reported with the results (as this influences the accuracy and variation of test results).

5.3. Test Repetition

The test MUST be repeated N times for each frame size in the subset list, and each Back-to-back Frame value made available for further processing (below).

5.4. Benchmark Calculations

For each Frame size, calculate the following summary statistics for Back-to-back Frame values over the N tests:

- Average (Benchmark)
- Minimum
- Maximum

o Standard Deviation

Further, calculate the Implied DUT Buffer Time and the Corrected DUT Buffer Time in seconds, as follows:

Implied DUT Buffer Time =

Average num of Back-to-back Frames / Max Theoretical Frame Rate

The formula above is simply expressing the Burst of Frames in units of time.

The next step is to apply a correction factor that accounts for the DUT's frame forwarding operation during the test (assuming a simple model of the DUT composed of a buffer and a forwarding function).

Corrected DUT Buffer Time =

$$= \text{Implied DUT Buffer Time} * \frac{\text{Measured Throughput}}{\text{Max Theoretical Frame Rate}}$$

where:

1. The "Measured Throughput" is the RFC2544 Throughput Benchmark for the frame size tested, and MUST be expressed in Frames per second in this equation.

2. The "Max Theoretical Frame Rate" is a calculated value for the interface speed and link layer technology used, and MUST be expressed in Frames per second in this equation.

The term on the far right in the formula for Corrected DUT Buffer Time accounts for all the frames in the Burst that were transmitted by the DUT *while the Burst of frames were sent in*. So, these frames are not in the Buffer and the Buffer size is more accurately estimated by excluding them.

6. Reporting

The back-to-back results SHOULD be reported in the format of a table with a row for each of the tested frame sizes. There SHOULD be columns for the frame size and for the resultant average frame count for each type of data stream tested.

The number of tests Averaged for the Benchmark, N, MUST be reported.

The Minimum, Maximum, and Standard Deviation across all complete tests SHOULD also be reported.

The Corrected DUT Buffer Time SHOULD also be reported.

If the tester operates using a maximum burst length in frames, then this maximum length SHOULD be reported.

Frame Size, octets	Ave B2B Length, frames	Min,Max,StdDev	Corrected Buff Time, Sec
64	26000	25500,27000,20	0.00004

Back-to-Back Frame Results

Static and configuration parameters:

Number of test repetitions, N

Minimum Step Size (during searches), in frames.

7. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization using controlled stimuli in a laboratory environment, with dedicated address space and the other constraints [RFC2544].

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network. See [RFC6815].

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the DUT/SUT.

Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

8. IANA Considerations

This memo makes no requests of IANA.

9. Acknowledgements

Thanks to Trevor Cooper, Sridhar Rao, and Martin Klozik of the VSPERF project for many contributions to the testing [VSPERF-b2b]. Yoshiaki Itou has also investigated the topic, and made useful suggestions.

10. References

10.1. Normative References

[RFC1242] Bradner, S., "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, DOI 10.17487/RFC1242, July 1991, <https://www.rfc-editor.org/info/rfc1242>.

[RFC1944] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 1944, DOI 10.17487/RFC1944, May 1996, <https://www.rfc-editor.org/info/rfc1944>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <https://www.rfc-editor.org/info/rfc2544>.

[RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, DOI 10.17487/RFC5180, May 2008, <https://www.rfc-editor.org/info/rfc5180>.

[RFC6201] Asati, R., Pignataro, C., Calabria, F., and C. Olvera, "Device Reset Characterization", RFC 6201, DOI 10.17487/RFC6201, March 2011, <https://www.rfc-editor.org/info/rfc6201>.

[RFC6815] Bradner, S., Dubray, K., McQuaid, J., and A. Morton, "Applicability Statement for RFC 2544: Use on Production Networks Considered Harmful", RFC 6815, DOI 10.17487/RFC6815, November 2012, <https://www.rfc-editor.org/info/rfc6815>.

[RFC6985] Morton, A., "IMIX Genome: Specification of Variable Packet Sizes for Additional Testing", RFC 6985, DOI 10.17487/RFC6985, July 2013, <https://www.rfc-editor.org/info/rfc6985>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[OPNFV-2017]
Cooper, T., Morton, A., and S. Rao, "Dataplane Performance, Capacity, and Benchmarking in OPNFV", June 2017, <https://wiki.opnfv.org/download/attachments/10293193/VSPERF-Dataplane-Perf-Cap-Bench.pptx?api=v2>.

[RFC8239] Avramov, L. and J. Rapp, "Data Center Benchmarking Methodology", RFC 8239, DOI 10.17487/RFC8239, August 2017, <https://www.rfc-editor.org/info/rfc8239>.

[TST009] ETSI Network Function Virtualization ISG, "ETSI GS NFV-TST 009 V3.1.1 (2018-10), "Network Functions Virtualisation (NFV) Release 3; Testing; Specification of Networking Benchmarks and Measurement Methods for NFVI"", October 2018, <https://www.etsi.org/deliver/etsi_gs/NFV-TST/001_099/009/03.01.01_60/gs_NFV-TST009v030101p.pdf>.

[VSPERF-b2b]
Morton, A., "Back2Back Testing Time Series (from CI)", June 2017, <https://wiki.opnfv.org/display/vsperf/Traffic+Generator+Testing#TrafficGeneratorTesting-AppendixB:Back2BackTestingTimeSeries(fromCI)>.

[VSPERF-BSLV]
Morton, A. and S. Rao, "Evolution of Repeatability in Benchmarking: Fraser Plugfest (Summary for IETF BMWG)", July 2018, <https://datatracker.ietf.org/meeting/102/materials/slides-102-bmwg-evolution-of-repeatability-in-benchmarking-fraser-plugfest-summary-for-ietf-bmwg-00>.

Author's Address

Al Morton
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com

BMWG
Internet-Draft
Intended status: Informational
Expires: September 22, 2016

R. Rosa, Ed.
Unicamp
R. Szabo
Ericsson

March 21, 2016

VNF Benchmarking Methodology
draft-rosa-bmwg-vnfbench-00

Abstract

This document describes VNF benchmarking methodologies.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 22, 2016.

Copyright Notice

Table of Contents

1. Introduction

New paradigms of network services envisioned by NFV bring VNFs as software based entities, which can be deployed in virtualized environments [ETS14a]. In order to be managed/orchestrated or compared with physical network functions, VNF Descriptors can specify performance profiles containing metrics (e.g., throughput) associated with allocated resources (e.g., vCPU). This document describes benchmarking methodologies to obtain VNF profiles (resource - performance figures).

2. Terminology

The reader is assumed to be familiar with the terminology as defined in the European Telecommunications Standards Institute (ETSI) NFV document [ETS14b]. Some of these terms, and others commonly used in this document, are defined below.

NFV: Network Function Virtualization - The principle of separating network functions from the hardware they run on by using virtual hardware abstraction.

NFVI PoP: NFV Infrastructure Point of Presence - Any combination of virtualized compute, storage and network resources.

NFVI: NFV Infrastructure - Collection of NFVI PoPs under one orchestrator.

VIM: Virtualized Infrastructure Manager - functional block that is responsible for controlling and managing the NFVI compute, storage and network resources, usually within one operator's Infrastructure Domain (e.g. NFVI-PoP).

NFVO: NFV Orchestrator - functional block that manages the Network Service (NS) life-cycle and coordinates the management of NS life-cycle, VNF life-cycle (supported by the VNFM) and NFVI resources (supported by the VIM) to ensure an optimized allocation of the necessary resources and connectivity

VNF: Virtualized Network Function - a software-based network function.

VNFD: Virtualised Network Function Descriptor - configuration template that describes a VNF in terms of its deployment and operational behaviour, and is used in the process of VNF on-boarding and managing the life cycle of a VNF instance.

VNF-FG: Virtualized Network Function Forwarding Graph - an ordered list of VNFs creating a service chain.

MANO: Management and Orchestration - In the ETSI NFV framework [ETS14a], this is the global entity responsible for management and orchestration of NFV life-cycle.

Network Service: composition of Network Functions and defined by its functional and behavioural specification.

Additional terminology not defined by ETSI NFV ISG.

VNF-BP: VNF Benchmarking Profile - the specification how to measure a VNF Profile. VNF-BP may be specific to a VNF or applicable to several VNF types. The specification includes structural and functional instructions, and variable parameters (metrics) at different abstractions (e.g., vCPU, memory, throughput, latency; session, transaction, tenants, etc.).

VNF Profile: is a mapping between virtualized resources (e.g., vCPU, memory) and VNF performance (e.g., throughput, latency between in/out ports) at a given NFVI PoP. An orchestration function can use the VNF Profile to select a host (NFVI PoP) for a VNF and to allocate necessary resources to deliver the required performance characteristics.

Customer: A user/subscriber/consumer of ETSI's Network Service.

Agents: Network Functions performing benchmarking tasks (e.g., synthetic traffic sources and sinks; measurement and observation functions, etc.).

SUT: System Under Test comprises the VNF under test.

3. Scope

This document assumes VNFs as black boxes when defining VNF performance benchmarking methodologies. White box benchmarking of VNFs are left for further studies and may be added later.

4. Assumptions

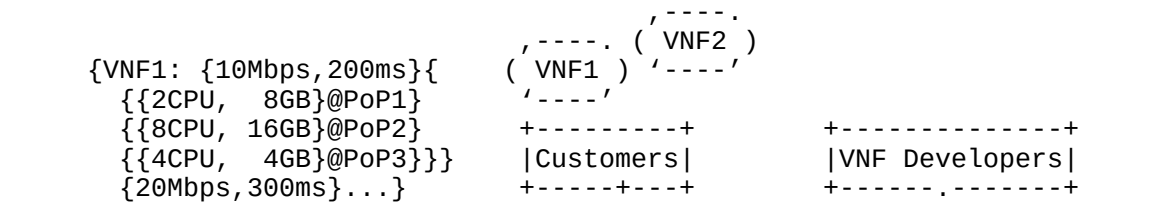
We assume a VNF benchmarking set-up as shown in Figure 1. Customers can request Network Services (NS) from an NFVO with associated service level specifications (e.g., throughput and delay). The NFVO, in turn, must select hosts and software resource allocations for the VNFs and build the necessary network overlay to meet the requirements. Therefore, the NFVO must know VNF Profiles per target hosts to perform location and resource assignments.

In a highly dynamic environment, where both the VNF instances (e.g., revised VM image) and the NFVI resources (e.g., hw upgrades) are changing, the NFVO should be able to create VNF Profiles on-demand.

We assume, that based on VNF Benchmarking Profile definitions NFVOs can run benchmarking evaluations to learn VNF Profiles per target hosts.

In a virtualization environment, however, not only the SUT but all the other benchmarking agents may be software defined (physical or virtualized network functions).

Figure 1 shows an example, where the NFVO can use PoPa and PoPb to set-up benchmarking functions to test VNFs hosted in PoP 1, 2, 3 domains corresponding to VIM 1, 2 and 3. The NFVO uses the VNF Benchmarking Profiles to deploy agents according to the SUT VNF. The VNF Benchmarking Profile is defined by the VNF Developer. The results of the VNF benchmarking is stored in a VNF Profile.



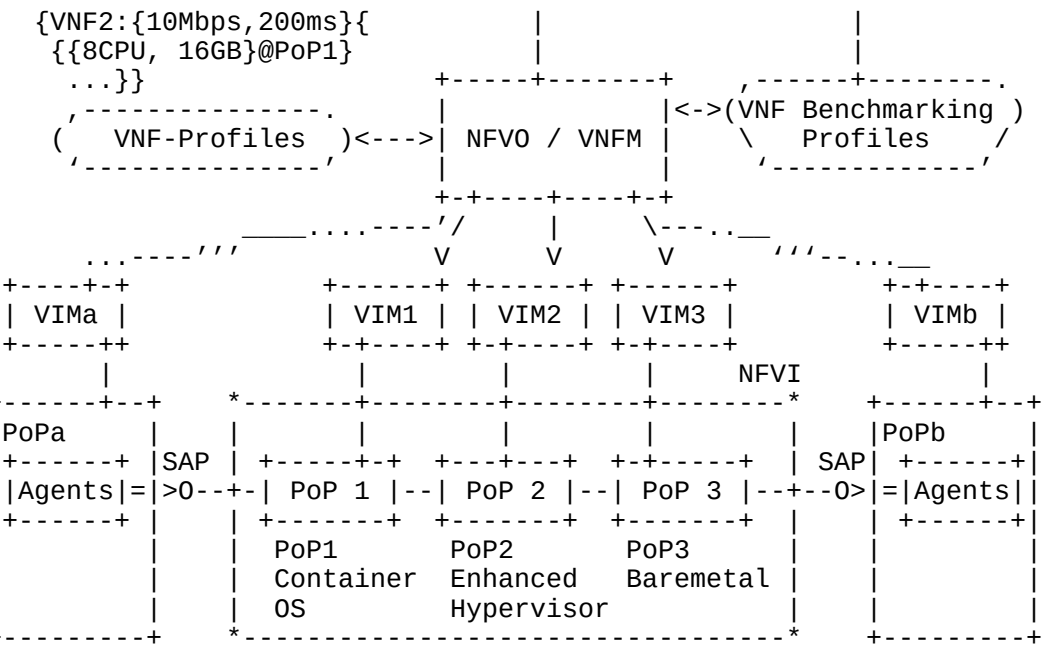


Figure 1: VNF Testing Scenario

5. VNF Benchmarking Considerations

VNF benchmarking considerations are defined in [Mor15]. Additionally, VNF pre-deployment testing considerations are well explored in [ETS14c].

This document list further considerations:

Black-Box SUT with Black-Box Benchmarking Agents: In virtualization environments neither the VNF instance nor the underlying virtualization environment nor the agents specifics may be known by the entity managing abstract resources. This implies black box testing with black box functional components, which are configured by opaque configuration parameters defined by the VNF developers or alike for the benchmarking entity (e.g., NFVO).

6. Methodology

Following the ETSI's model ([ETS14c]), we distinguish three methods for VNF evaluation:

Benchmarking: Where resource {cpu, memory, storage} parameters are provided and the corresponding {latency, throughput} performance

parameters are obtained. Note, such request might create multiple reports, for example, with minimal latency or maximum throughput results.

Verification: Both resources {cpu, memory, storage} and performance {latency, throughput} parameters are provided and agents verifies if the given association is correct or not.

Dimensioning: Where performance parameters {latency, throughput} are provided and the corresponding {cpu, memory, storage} resource parameters obtained. Note, multiple deployment interactions may be required, or if possible, underlying allocated resources need to be dynamically altered.

Note: Verification and Dimensioning can be reduced to Benchmarking. Therefore, we detail Benchmarking in what follows.

6.1. Benchmarking

All benchmarking methodologies described in this section consider the definition of VNF-BPs for each testing procedure. Information about Benchmarking Methodology for Network Interconnect Devices, defined in [rfc2544], is considered in all subsections below. Besides, the tests are defined based on notions introduced and discussed in the IP Performance Metrics (IPPM) Framework document [rfc2330].

6.1.1. Throughput

Objective: Provide, for a particular set of resources allocated, the throughput among two or more VNF ports, expressed in VNF-BP.

Prerequisite: VNF (SUT) must be deployed and stable and its allocated resources collected. VNF must be reachable by agents. The frame size to be used for agents must be defined in the VNF-BP.

Procedure:

1. Establish connectivity between agents and VNF ports.

2. Agents initiate source of traffic, specifically designed for VNF test, increasing rate periodically.

3. Throughput is measured when traffic rate is achieved without frame losses.

Reporting Format: report must contain VNF allocated resources and throughput measured (aka throughput in [rfc2544]).

6.1.2. Latency

Objective: Provide, for a particular set of resources allocated, the latency among two or more VNF ports, expressed in VNF-BP.

Prerequisite: VNF (SUT) must be deployed and stable and its allocated resources collected. VNF must be reachable by agents. The frame size and respective throughput to be used for agents must be defined in the VNF-BP.

Procedure:

1. Establish connectivity between agents and VNF ports.

2. Agents initiate source of traffic, throughput and frame size specifically designed for VNF test.

3. Latency is measured when throughput is achieved for the period of time specified in VNF-BP.

Reporting Format: report must contain VNF allocated resources, throughput used for stimulus and latency measurement (aka latency in [rfc2544]).

6.1.3. Frame Loss Rate

Objective: Provide, for a particular set of resources allocated, the frame loss rate among two or more VNF ports, expressed in VNF-BP.

Prerequisite: VNF (SUT) must be deployed and stable, its allocated resources collected specifying any particular feature of the underlying VNF virtualized environment, provided by NFVO/VIM or independently extracted. VNF must be reachable by agents. Rate of source traffic and frame type used for agents stimulus must be defined in VNF-BP.

Procedure:

1. Establish connectivity between agents and VNF ports.

2. Agents initiate source of traffic, specifically designed for VNF test, achieving rate of source traffic defined in VNF-BP.

3. Frame loss rate is measured when pre-defined traffic rate is achieved for period of time established in VNF-BP.

Reporting Format: report must contain VNF allocated resources, rate of source traffic used as stimulus and frame loss rate measurement (aka frame loss rate in [rfc2544]).

7. Summary

This document describes black-box benchmarking methodologies for black-box VNFs in virtualization environments (e.g., ETSI NFV framework) to create VNF Profiles containing the association of resources and performance metrics of a given VNF at a given host (e.g., NFVI PoP).

The authors see the following next steps:

VNF Scaling: Two scaling options: single instance with more resources or multiple instances. Questions: What is the maximum performance of a single instance VNF at a given host with increasing resources? How many independent VNF instances (or components) can be run with maximum performance at a given host? On the other hand, what is the performance of the smallest resource footprint VNF allocation?

VNF instantiation time: this metric concerns at least three components: VNF bootstraping (SUT), execution environment and the orchestration process.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

TBD

10. Acknowledgement

The authors would like to thank the support of Ericsson Research, Brazil.

This work is partially supported by FP7 UNIFY, a research project partially funded by the European Community under the Seventh Framework Program (grant agreement no. 619609). The views expressed here are those of the authors only. The European Commission is not liable for any use that may be made of the information in this document.

11. Informative References

[ETS14a] ETSI, "Architectural Framework - ETSI GS NFV 002 V1.2.1", Dec 2014, <http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01-_60/gs_NFV002v010201p.pdf>.

[ETS14b] ETSI, "Terminology for Main Concepts in NFV - ETSI GS NFV 003 V1.2.1", Dec 2014, <http://www.etsi.org/deliver/etsi_gs/NFV/001_099-/003/01.02.01_60/gs_NFV003v010201p.pdf>.

[ETS14c] ETSI, "NFV Pre-deployment Testing - ETSI GS NFV TST001 V0.0.15", February 2016, <http://docbox.etsi.org/ISG/NFV/Open/DRAFTS/TST001_-_Pre-deployment_Validation/NFV-TST001v0015.zip>.

[Mor15] A. Morton, "Considerations for Benchmarking Virtual Network Functions and Their Infrastructure", February 2015, <http://tools.ietf.org/html/draft-morton-bmwg-virtual-net-03>.

[rfc2330] V. Paxson, G. Almes, J. Mahdavi, M. Mathis, "Framework for IP Performance Metrics", May 1998, <https://tools.ietf.org/html/rfc2330>.

[rfc2544] S. Bradner and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", March 1999, <https://tools.ietf.org/html/rfc2544>.

Authors' Addresses

Raphael Vicente Rosa (editor)
University of Campinas
Av. Albert Einstein 300
Campinas, Sao Paulo 13083-852
Brazil

Email: raphaelvrosa@gmail.com
URI: http://www.intrig.dca.fee.unicamp.br/

Robert Szabo
Ericsson Research, Hungary
Irinyi Jozsef u. 4-20
Budapest 1117
Hungary

Email: robert.szabo@ericsson.com
URI: http://www.ericsson.com/

BMWG
Internet Draft
Intended status: Informational
Expires: September 2017

S. Kommu
VMware
J. Rapp
VMware

March 13, 2017

Considerations for Benchmarking Network Virtualization Platforms
draft-skommu-bmwg-nvp-00.txt

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on September 13, 2017.

Copyright Notice

Abstract

Current network benchmarking methodologies are focused on physical networking components and do not consider the actual application layer traffic patterns and hence do not reflect the traffic that virtual networking components work with. The purpose of this document is to distinguish and highlight benchmarking considerations when testing and evaluating virtual networking components in the data center.

Table of Contents

1. Introduction

Datacenter virtualization that includes both compute and network virtualization is growing rapidly as the industry continues to look for ways to improve productivity, flexibility and at the same time

cut costs. Network virtualization, is comparatively new and expected to grow tremendously similar to compute virtualization. There are multiple vendors and solutions out in the market, each with their own benchmarks to showcase why a particular solution is better than another. Hence, the need for a vendor and product agnostic way to benchmark multivendor solutions to help with comparison and make informed decisions when it comes to selecting the right network virtualization solution.

Applications traditionally have been segmented using VLANs and ACLs between the VLANs. This model does not scale because of the 4K scale limitations of VLANs. Overlays such as VXLAN were designed to address the limitations of VLANs

With VXLAN, applications are segmented based on VXLAN encapsulation (specifically the VNI field in the VXLAN header), which is similar to VLAN ID in the 802.1Q VLAN tag, however without the 4K scale limitations of VLANs. For a more detailed discussion on this subject please refer RFC 7364 "Problem Statement: Overlays for Network Virtualization".

VXLAN is just one of several Network Virtualization Overlays(NVO). Some of the others include STT, Geneve and NVGRE. . STT and Geneve have expanded on the capabilities of VXLAN. Please refer IETF’s nvo3 working group < https://datatracker.ietf.org/wg/nvo3/documents/> for more information.

Modern application architectures, such as Micro-services, are going beyond the three tier app models such as web, app and db. Benchmarks MUST consider whether the proposed solution is able to scale up to the demands of such applications and not just a three-tier architecture.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

3. Definitions

3.1. System Under Test (SUT)

Traditional hardware based networking devices generally use the device under test (DUT) model of testing. In this model, apart from any allowed configuration, the DUT is a black box from a testing perspective. This method works for hardware based networking devices since the device itself is not influenced by any other components outside the DUT.

Virtual networking components cannot leverage DUT model of testing as the DUT is not just the virtual device but includes the hardware components that were used to host the virtual device

Hence SUT model MUST be used instead of the traditional device under test

With SUT model, the virtual networking component along with all software and hardware components that host the virtual networking component MUST be considered as part of the SUT.

Virtual networking components may also work with higher level TCP segments such as TSO. In contrast, all physical switches and routers, including the ones that act as initiators for NVOs, work with L2/L3 packets.

Please refer to section 5 Figure 1 for a visual representation of System Under Test in the case of Intra-Host testing and section 5 Figure 2 for System Under Test in the case of Inter-Host testing

3.2. Network Virtualization Platform

This document does not focus on Network Function Virtualization.

Network Function Virtualization (NFV) focuses on being independent of networking hardware while providing the same functionality. In the case of NFV, traditional benchmarking methodologies recommended by IETF may be used. Considerations for Benchmarking Virtual Network Functions and Their Infrastructure IETF document addresses benchmarking NFVs.

Typical NFV implementations emulate in software, the characteristics and features of physical switches. They are similar to any physical L2/L3 switch from the perspective of the packet size, which is typically enforced based on the maximum transmission unit used.

Network Virtualization platforms on the other hand, are closer to the application layer and are able to work with not only L2/L3

packets but also segments that leverage TCP optimizations such as Large Segment Offload (LSO).

NVPs leverage TCP stack optimizations such as TCP Segmentation Offload (TSO) and Large Receive Offload (LRO) that enables NVPs to work with much larger payloads of up to 64K unlike their counterparts such as NFVs.

Because of the difference in the payload, which translates into one operation per 64K of payload in NVP verses ~40 operations for the same amount of payload in NFV because of having to divide it to MTU sized packets, results in considerable difference in performance between NFV and NVP.

Please refer to figure 1 for a pictorial representation of this primary difference between NPV and NFV for a 64K payload segment/packet running on network set to 1500 bytes MTU.

Note: Payload sizes in figure 1 are approximates.

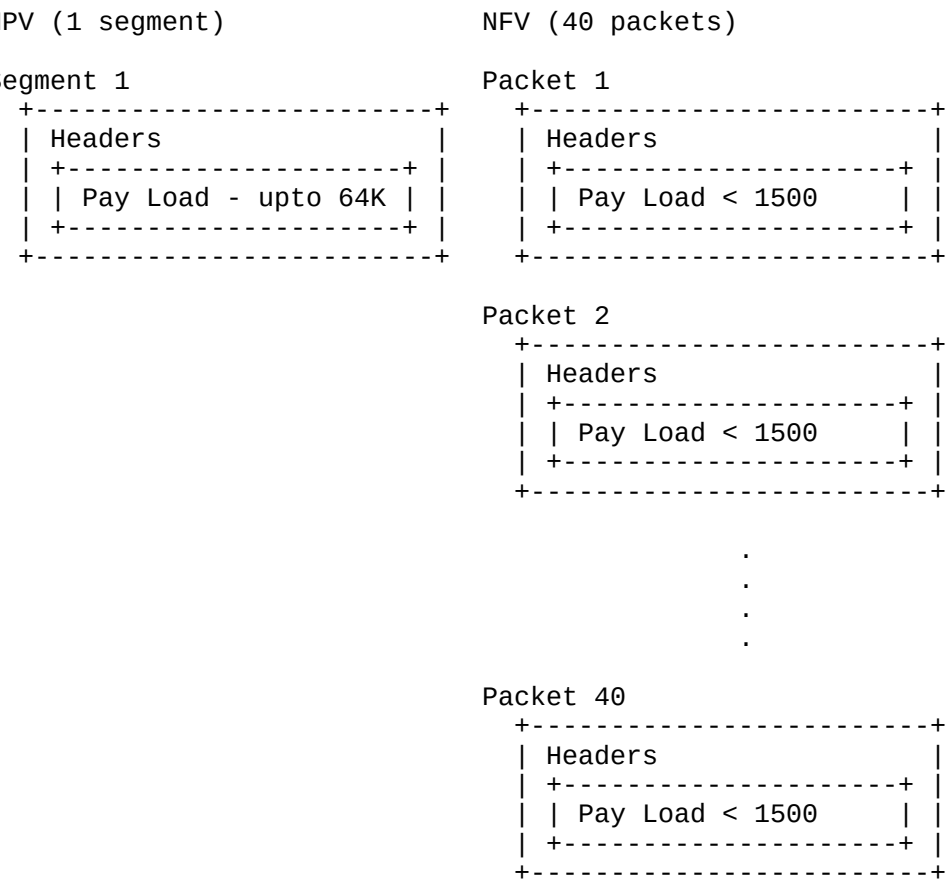


Figure 1 Payload NPV vs NFV

Hence, normal benchmarking methods are not relevant to the NVPs.

Instead, newer methods that take into account the built in advantages of TCP provided optimizations MUST be used for testing Network Virtualization Platforms.

3.3. Micro-services

Traditional monolithic application architectures such as the three tier web, app and db architectures are hitting scale and deployment limits for the modern use cases.

Micro-services make use of classic unix style of small app with single responsibility.

These small apps are designed with the following characteristics:

Each application only does one thing - like unix tools

Small enough that you could rewrite instead of maintain

Embedded with a simple web container

Packaged as a single executable

Installed as daemons

Each of these applications are completely separate

Interact via uniform interface

REST (over HTTP/HTTPS) being the most common

With Micro-services architecture, a single web app of the three tier application model could now have 100s of smaller apps dedicated to do just one job.

These 100s of small one responsibility only services will MUST be secured into their own segment - hence pushing the scale boundaries of the overlay from both simple segmentation perspective and also from a security perspective

4. Scope

This document does not address Network Function Virtualization has been covered already by previous IETF documents (https://datatracker.ietf.org/doc/draft-ietf-bmwg-virtual-net/?include_text=1) the focus of this document is Network Virtualization Platform where the network functions are an intrinsic part of the hypervisor’s TCP stack, working closer to the application layer and leveraging performance optimizations such TSO/RSS provided by the TCP stack and the underlying hardware.

4.1. Virtual Networking for Datacenter Applications

While virtualization is growing beyond the datacenter, this document focuses on the virtual networking for east-west traffic within the datacenter applications only. For example, in a three tier app such web, app and db, this document focuses on the east-west traffic between web and app. It does not address north-south web traffic accessed from outside the datacenter. A future document would address north-south traffic flows.

This document addresses scale requirements for modern application architectures such as Micro-services to consider whether the proposed solution is able to scale up to the demands of micro-services application models that basically have 100s of small services communicating on some standard ports such as http/https using protocols such as REST

4.2. Interaction with Physical Devices

Virtual network components cannot be tested independent of other components within the system. Example, unlike a physical router or a firewall, where the tests can be focused directly solely on the device, when testing a virtual router or firewall, multiple other devices may become part of the system under test. Hence the characteristics of these other traditional networking switches and routers, LB, FW etc. MUST be considered.

- Hashing method used
- Over-subscription rate
- Throughput available
- Latency characteristics

5. Interaction with Physical Devices

In virtual environments, System Under Test (SUT) may often share resources and reside on the same Physical hardware with other components involved in the tests. Hence SUT MUST be clearly defined. In this tests, a single hypervisor may host multiple servers, switches, routers, firewalls etc.,

Intra host testing: Intra host testing helps in reducing the number of components involved in a test. For example, intra host testing would help focus on the System Under Test, logical switch and the hardware that is running the hypervisor that hosts the logical switch, and eliminate other components. Because of the nature of virtual infrastructures and multiple elements being hosted on the same physical infrastructure, influence from other components cannot be completely ruled out. For example, unlike in physical infrastructures, logical routing or distributed firewall MUST NOT be benchmarked independent of logical switching. System Under Test definition MUST include all components involved with that particular test.

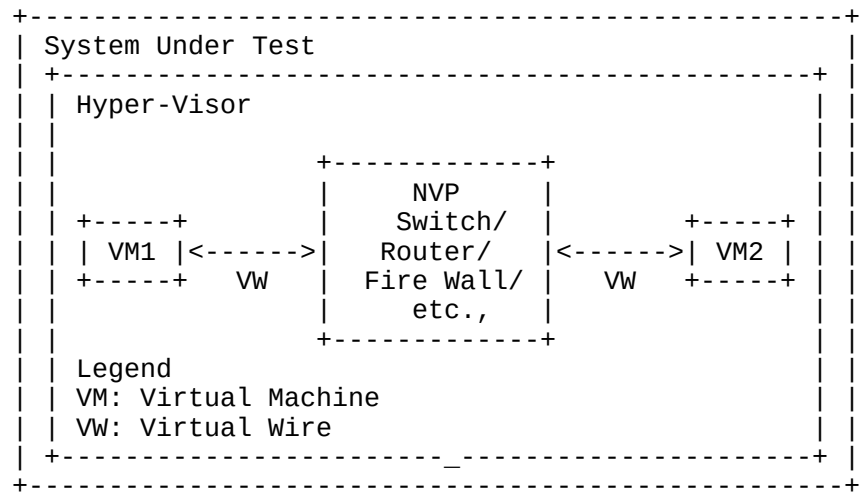


Figure 2 Intra-Host System Under Test

Inter host testing: Inter host testing helps in profiling the underlying network interconnect performance. For example, when testing Logical Switching, inter host testing would not only test the logical switch component but also any other devices that are part of the physical data center fabric that connects the two hypervisors. System Under Test MUST be well defined to help with repeatability of tests. System Under Test definition in the case of inter host testing, MUST include all components, including the underlying network fabric.

Figure 2 is a visual representation of system under test for inter-host testing

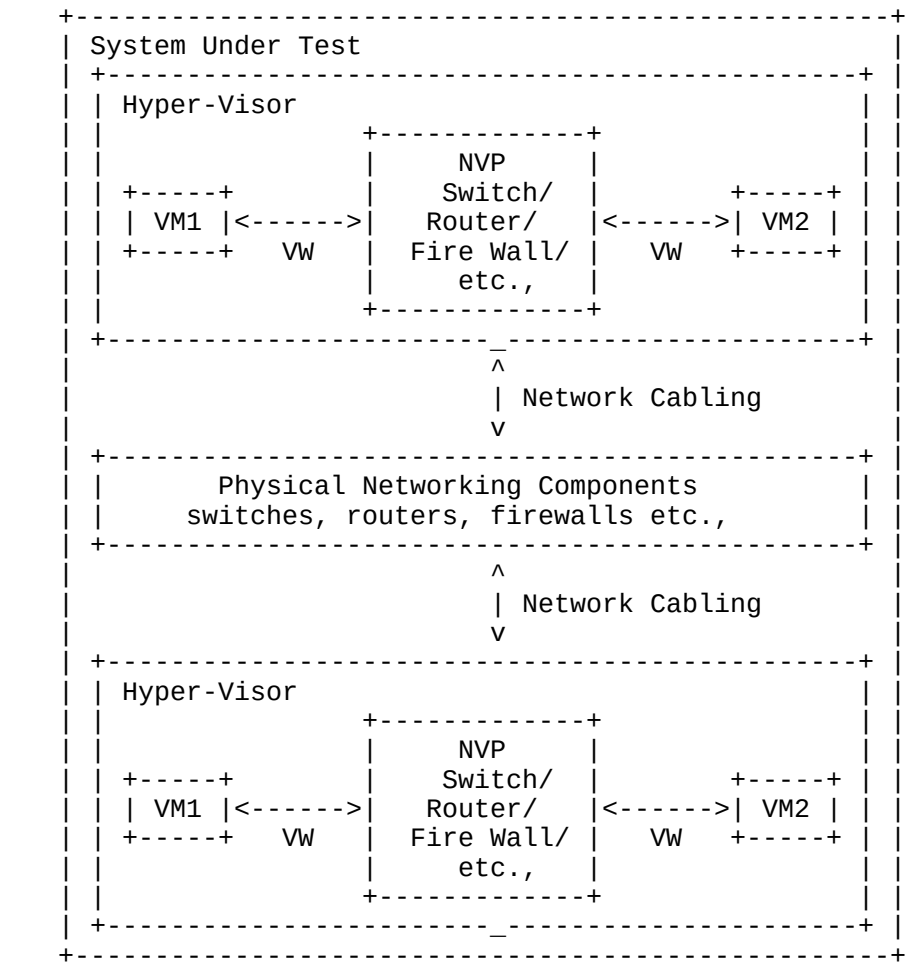


Figure 3 Inter-Host System Under Test

Virtual components have a direct dependency on the physical infrastructure that is hosting these resources. Hardware characteristics of the physical host impact the performance of the virtual components. The components that are being tested and the impact of the other hardware components within the hypervisor on the performance of the SUT MUST be documented. Virtual component performance is influenced by the physical hardware components within the hypervisor. Access to various offloads such as TCP segmentation

offload, may have significant impact on performance. Firmware and driver differences may also significantly impact results based on whether the specific driver leverages any hardware level offloads offered. Hence, all physical components of the physical server running the hypervisor that hosts the virtual components MUST be documented along with the firmware and driver versions of all the components used to help ensure repeatability of test results. For example, BIOS configuration of the server MUST be documented as some of those changes are designed to improve performance. Please refer to Appendix A for a partial list of parameters to document.

5.1. Server Architecture Considerations

When testing physical networking components, the approach taken is to consider the device as a black-box. With virtual infrastructure, this approach would no longer help as the virtual networking components are an intrinsic part of the hypervisor they are running on and are directly impacted by the server architecture used. Server hardware components define the capabilities of the virtual networking components. Hence, server architecture MUST be documented in detail to help with repeatability of tests. And the entire hardware and software components become the SUT.

5.1.1. Frame format/sizes within the Hypervisor

Maximum Transmission Unit (MTU) limits physical network component's frame sizes. The most common max supported MTU for physical devices is 9000. However, 1500 MTU is the standard. Physical network testing and NFV uses these MTU sizes for testing. However, the virtual networking components that live inside a hypervisor, may work with much larger segments because of the availability of hardware and software based offloads. Hence, the normal smaller packets based testing is not relevant for performance testing of virtual networking components. All the TCP related configuration such as TSO size, number of RSS queues MUST be documented along with any other physical NIC related configuration.

Virtual network components work closer to the application layer then the physical networking components. Hence virtual network components work with type and size of segments that are often not the same type and size that the physical network works with. Hence, testing virtual network components MUST be done with application layer segments instead of the physical network layer packets.

5.1.2. Baseline testing with Logical Switch

Logical switch is often an intrinsic component of the test system along with any other hardware and software components used for

testing. Also, other logical components cannot be tested independent of the Logical Switch.

5.1.3. Repeatability

To ensure repeatability of the results, in the physical network component testing, much care is taken to ensure the tests are conducted with exactly the same parameters. Parameters such as MAC addresses used etc.,

When testing NPV components with an application layer test tool, there may be a number of components within the system that may not be available to tune or to ensure they maintain a desired state. Example: housekeeping functions of the underlying Operating System.

Hence, tests MUST be repeated a number of times and each test case MUST be run for at least 2 minutes if test tool provides such an option. Results SHOULD be derived from multiple test runs. Variance between the tests SHOULD be documented.

5.1.4. Tunnel encap/decap outside the hypervisor

Logical network components may also have performance impact based on the functionality available within the physical fabric. Physical fabric that supports NVO encap/decap is one such case that has considerable impact on the performance. Any such functionality that exists on the physical fabric MUST be part of the test result documentation to ensure repeatability of tests. In this case SUT MUST include the physical fabric

5.1.5. SUT Hypervisor Profile

Physical networking equipment has well defined physical resource characteristics such as type and number of ASICs/SoCs used, amount of memory, type and number of processors etc., Virtual networking components' performance is dependent on the physical hardware that hosts the hypervisor. Hence the physical hardware usage, which is part of SUT, for a given test MUST be documented. Example, CPU usage when running logical router.

CPU usage changes based on the type of hardware available within the physical server. For example, TCP Segmentation Offload greatly reduces CPU usage by offloading the segmentation process to the NIC card on the sender side. Receive side scaling offers similar benefit on the receive side. Hence, availability and status of such hardware MUST be documented along with actual CPU/Memory usage when the virtual networking components have access to such offload capable hardware.

Following is a partial list of components that MUST be documented - both in terms of what's available and also what's used by the SUT -

- o CPU - type, speed, available instruction sets (e.g. AES-NI)
- o Memory - type, amount
- o Storage - type, amount
- o NIC Cards - type, number of ports, offloads available/used, drivers, firmware (if applicable), HW revision
- o Libraries such as DPDK if available and used
- o Number and type of VMs used for testing and
  - o vCPUs
  - o RAM
  - o Storage
  - o Network Driver
  - o Any prioritization of VM resources
  - o Operating System type, version and kernel if applicable
  - o TCP Configuration Changes - if any
  - o MTU
- o Test tool
  - o Workload type
  - o Protocol being tested
  - o Number of threads
  - o Version of tool
- o For inter-hypervisor tests,
  - o Physical network devices that are part of the test

! Note: For inter-hypervisor tests, system under test is no longer only the virtual component that is being tested but the entire fabric that connects the

virtual components become part of the system under test.

6. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization of a Device Under Test/System Under Test (DUT/SUT) using controlled stimuli in a laboratory environment, with dedicated address space and the constraints specified in the sections above.

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the DUT/SUT.

Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

7. IANA Considerations

No IANA Action is requested at this time.

8. Conclusions

Network Virtualization Platforms, because of their proximity to the application layer and since they can take advantage of TCP stack optimizations, do not function on packets/sec basis. Hence, traditional benchmarking methods, while still relevant for Network Function Virtualization, are not designed to test Network Virtualization Platforms. Also, advances in application architectures such as micro-services, bring new challenges and need benchmarking not just around throughput and latency but also around scale. New benchmarking methods that are designed to take advantage of the TCP optimizations or needed to accurately benchmark performance of the Network Virtualization Platforms

9. References

9.1. Normative References

[RFC7364] T. Narten, E. Gray, D. Black, L. Fang, L. Kreeger, M. Napierala, "Problem Statement: Overlays for Network Virtualization", RFC 7364, October 2014, https://datatracker.ietf.org/doc/rfc7364/

[nv03] IETF, WG, Network Virtualization Overlays, < https://datatracker.ietf.org/wg/nvo3/documents/>

9.2. Informative References

[1] A. Morton " Considerations for Benchmarking Virtual Network Functions and Their Infrastructure", draft-ietf-bmwg-virtual-net-03, < https://datatracker.ietf.org/doc/draft-ietf-bmwg-virtual-net/?include_text=1>

Appendix A. Partial List of Parameters to Document

A.1. CPU

CPU Vendor

CPU Number

CPU Architecture

# of Sockets (CPUs)

# of Cores

Clock Speed (GHz)

Max Turbo Freq. (GHz)

Cache per CPU (MB)

# of Memory Channels

Chipset

Hyperthreading (BIOS Setting)

Power Management (BIOS Setting)

VT-d

A.2. Memory

Memory Speed (MHz)

DIMM Capacity (GB)

# of DIMMs

DIMM configuration

Total DRAM (GB)

A.3. NIC

Vendor

Model

Port Speed (Gbps)

Ports

PCIe Version

PCIe Lanes

Bonded

Bonding Driver

Kernel Module Name

Driver Version

VXLAN TSO Capable

VXLAN RSS Capable

Ring Buffer Size RX

Ring Buffer Size TX

A.4. Hypervisor

Hypervisor Name

Version/Build

Based on

Hotfixes/Patches

OVS Version/Build

IRQ balancing

vCPUs per VM

Modifications to HV

Modifications to HV TCP stack

Number of VMs

IP MTU

Flow control TX (send pause)

Flow control RX (honor pause)

Encapsulation Type

A.5. Guest VM

Guest OS & Version

Modifications to VM

IP MTU Guest VM (Bytes)

Test tool used

Number of NetPerf Instances

Total Number of Streams

Guest RAM (GB)

A.6. Overlay Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

A.7. Gateway Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

Authors' Addresses

Samuel Kommu
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: skommu@vmware.com

Jacob Rapp
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: jrapp@vmware.com

ï»¿INTERNET-DRAFT

BMWG S. Kommu
Internet-Draft V Mware
Intended status: Informational J. Rapp
Expires: Sep 2019 V Mware

Mar 11, 2019

Considerations for Benchmarking Network Virtualization Platforms
draft-skommu-bmwg-nvp-03.txt

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on September 11, 2019.

Copyright Notice

Abstract

Current network benchmarking methodologies are focused on physical networking components and do not consider the actual application layer traffic patterns and hence do not reflect the traffic that virtual networking components work with when using network virtualization overlays (NVO3). The purpose of this document is to distinguish and highlight benchmarking considerations when testing and evaluating virtual networking components in the data center.

Table of Contents

1. Introduction

Datacenter virtualization that includes both compute and network virtualization is growing rapidly as the industry continues to look for ways to improve productivity, flexibility and at the same time cut costs. Network virtualization is comparatively new and expecte
d
to grow tremendously similar to compute virtualization. There are multiple vendors and solutions out in the market. Each vendor ofte
n
has their own recommendations on how to benchmark their solutions thus making it difficult to perform a apples-to-apples comparision between different solutions. Hence, the need for a vendor, product and cloud agnostic way to benchmark network virtualization solution
s
to help with comparison and make informed decisions when it comes t
o
selecting the right network virtualization solution.

Applications traditionally have been segmented using VLANs and ACLs between the VLANs. This model does not scale because of the 4K sca
le
limitations of VLANs. Overlays such as VXLAN were designed to address the limitations of VLANs.

With VXLAN, applications are segmented based on VXLAN encapsulation (specifically the VNI field in the VXLAN header), which is similar
to
VLAN ID in the 802.1Q VLAN tag, however without the 4K scale limitations of VLANs. For a more detailed discussion on this subje

ct

please refer RFC 7364 ’Problem Statement: Overlays for Network Virtualization’.

VXLAN is just one of several Network Virtualization Overlays (NVO). Some of the others include STT, Geneve and NVGRE. STT and Geneve have expanded on the capabilities of VXLAN. Please refer IETF’s nv

o3

working group < https://datatracker.ietf.org/wg/nvo3/documents/> for more information.

Modern application architectures, such as Micro-services, because of IP based connectivity within the app, place high demands on the networking and security when compared to the traditional three tier app models such as web, app and db. Benchmarks MUST consider whether the proposed solution is able to scale up to the demands of such applications and not just a three-tier architecture.

The benchmarks will be utilizing the various terminology and definitions from the NVO3 working group including RFC 8014 and RFC 8394.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Definitions

3.1. System Under Test (SUT)

Traditional hardware based networking devices generally use the device under test (DUT) model of testing. In this model, apart from any allowed configuration, the DUT is a black box from a testing perspective. This method works for hardware based networking devices since the device itself is not influenced by any other components outside the DUT.

Virtual networking components cannot leverage DUT model of testing as the DUT is not just the virtual device but includes the hardware components that were used to host the virtual device

Hence System Under Test (SUT) model MUST be used instead of the traditional device under test

With SUT model, the virtual networking component along with all software and hardware components that host the virtual networking component MUST be considered as part of the SUT.

Virtual networking components, because of their dependency on the underlying hardware and other software components, may end up leveraging NIC offload benefits, such as TCP Segmentation Offload (TSO), Large Receive Offload (LRO) and Rx / Tx Filters. Such underlying hardware and software level features, even though they may not be part of virtual networking stack itself, MUST be considered and documented. Note: Physical switches and routers, including the ones that act as initiators for NVOs, work with L2/L3 packets and may not be able to leverage TCP enhancements such as TSO.

Please refer to section 5 Figure 1 for a visual representation of System Under Test in the case of Intra-Host testing and section 5 Figure 2 for System Under Test in the case of Inter-Host testing.

3.2. Network Virtualization Platform

This document focuses on the Network Virtualization Overlay platform as outlined in RFC 8014 and use cases from RFC 8394.

Network Virtualization platforms, function closer to the application layer and are able to work with not only L2/L3 packets but also segments that leverage TCP optimizations such as Large Segment Offload (LSO).

NVPs leverage TCP stack optimizations such as TCP Segmentation Offload (TSO) and Large Receive Offload (LRO) that enables NVPs to work with much larger payloads of up to 64K unlike their counterparts such as NFVs.

Because of the difference in the payload, which translates into one operation per 64K of payload in NVP verses ~40 operations for the same amount of payload in NFV because of having to divide it to MTU sized packets, results in considerable difference in performance between NFV and NVP.

Please refer to figure 1 for a pictorial representation of this primary difference between NPV and NFV for a 64K payload segment/packet running on network set to 1500 bytes MTU.

Note: Payload sizes in figure 1 are approximates.

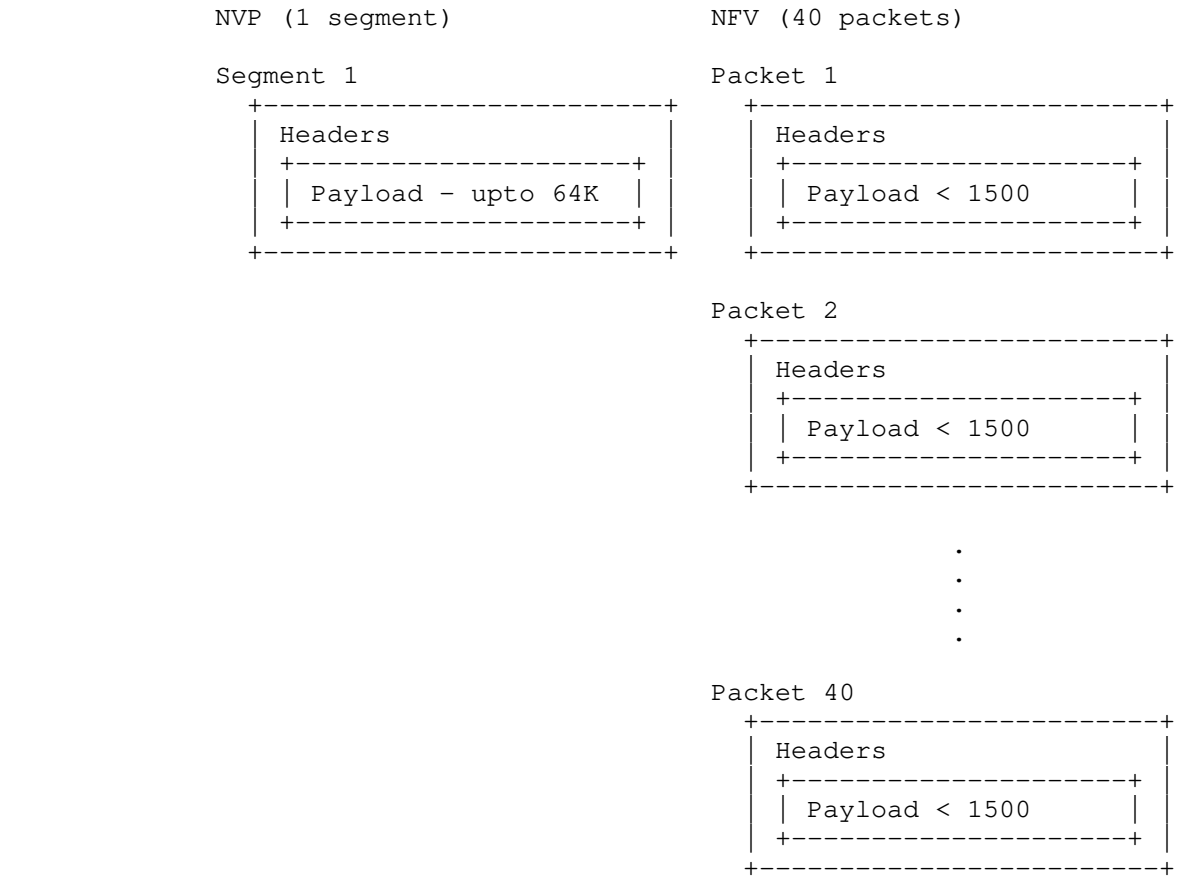


Figure 1! Payload NPV vs NFV

Hence, normal benchmarking methods are not relevant to the NVPs.

Instead, newer methods that leverage TCP optimizations MUST be used for testing Network Virtualization Platforms.

3.3. Microservices

Moving from traditional monolithic application architectures such as

the three tier web, app and db architectures to microservices model open up networking and security stacks to new scale and performance

related challenges. At a high level, in a microservices model, a traditional monolithic app that may use few IPs is broken down into 100s of individual one-responsibility-only applications where each application has connectivity and security related requirements. These 100s of small one-responsibility-only micro-services need their own IP and also secured into their own segments, hence pushing the scale boundaries of the overlay from both simple segmentation perspective and also from a security perspective.

For more details regarding microservices, please refer to wiki on microservices: https://en.wikipedia.org/wiki/Microservices

4. Scope

Focus of this document is the Network Virtualization Platform in two separate scenarios as outlined in RFC 8014 section 4, Network Virtualization Edge (NVE) and RFC 8394 section 1.1 Split-NVE and the associated learning phase:

4.1.1. Scenario 1

RFC 8014 Section 4.1 "NVE Co-located with server hypervisor": Where the entire NVE functionality will typically be implemented as part of the hypervisor and/or virtual switch on the server.

4.1.2. Scenario 2

RFC 8394 Section 1.1 "Split-NVE: A type of NVE (Network Virtualization Edge) where the functionalities are split across an end device supporting virtualization and an external network device."

4.1.3. Learning

Address learning rate is a key contributor to the overall performance of SUT specially in microservices type of use cases where a large amount of end-points are created and destroyed on demand.

4.1.4. Flow Optimization

There are several flow optimization algorithms that are designed to help improve latency or throughput. These optimizations MUST be documented.

4.1.5. Out of scope

This document does not address Network Function Virtualization which has been covered already by previous IETF documents

(https://datatracker.ietf.org/doc/draft-ietf-bmwg-virtual-net/?include_text=1).

Network Function Virtualization (NFV) focuses on being independent of networking hardware while providing the same functionality. In the case of NFV, traditional benchmarking methodologies recommended by IETF may be used. Considerations for Benchmarking Virtual Network Functions and Their Infrastructure IETF document addresses benchmarking NFVs.

Typical NFV implementations emulate in software, the characteristics and features of physical switches. They are similar to any physical L2/L3 switch from the perspective of the packet size, which is typically enforced based on the maximum transmission unit used.

4.2. Virtual Networking for Datacenter Applications

This document focuses on the virtual networking for east-west traffic within on-prem datacenter and/or cloud. For example, in a three tier app such web, app and db, this document focuses on the east-west traffic between web and app.

This document addresses scale requirements for modern application architectures such as Micro-services to consider whether the proposed solution is able to scale up to the demands of micro- services application models that basically have 100s of small services communicating on some standard ports such as http/https using protocols such as REST.

4.3. Interaction with Physical Devices

Virtual network components MUST NOT be tested independent of other components within the system. Example, unlike a physical router or a firewall, where the tests can be focused solely on the device, when testing a virtual router or firewall, multiple other devices may become part of the SUT. Hence the characteristics of these other traditional networking switches and routers, LB, FW etc. MUST be considered.

- Hashing method used
- Over-subscription rate
- Throughput available
- Latency characteristics

5. NVP Benchmarking Considerations

In virtual environments, the SUT may often share resources and resi
de
on the same physical hardware with other components involved in the tests. Hence SUT MUST be clearly documented. In these tests, a single hypervisor may host multiple servers, switches, routers, firewalls etc.

Intra host testing: Intra host testing helps in reducing the numbe
r
of components involved in a test. For example, intra host testing would help focus on the System Under Test, logical switch and the hardware that is running the hypervisor that hosts the logical switch, and eliminate other components. Because of the nature of virtual infrastructures and multiple elements being hosted on the same physical infrastructure, influence from other components canno
t
be completely ruled out. For example, unlike in physical infrastructures, logical routing or distributed firewall MUST NOT b
e
benchmarked independent of logical switching. System Under Test definition MUST include all components involved with that particula
r
test.

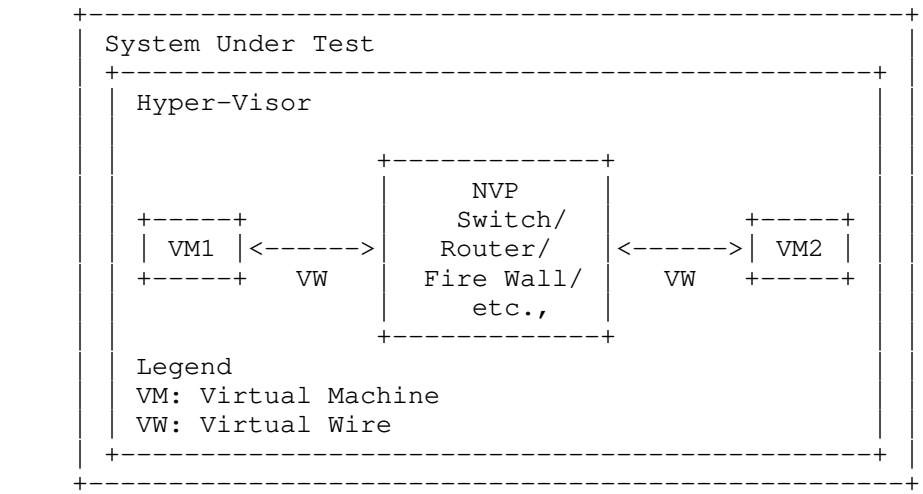


Figure 2! Intra-Host System Under Test

In the above figure, we only address the NVE co-located with the hypervisor.

Inter-host testing: Inter-host testing helps in profiling the underlying network interconnect performance. For example, when testing Logical Switching, inter host testing would not only test t
he

logical switch component but also any other devices that are part of the physical data center fabric that connects the two hypervisors. System Under Test MUST be well defined to help with repeatability of tests. System Under Test definition in the case of inter host testing, MUST include all components, including the underlying network fabric.

Figure 2 is a visual representation of system under test for inter-host testing.

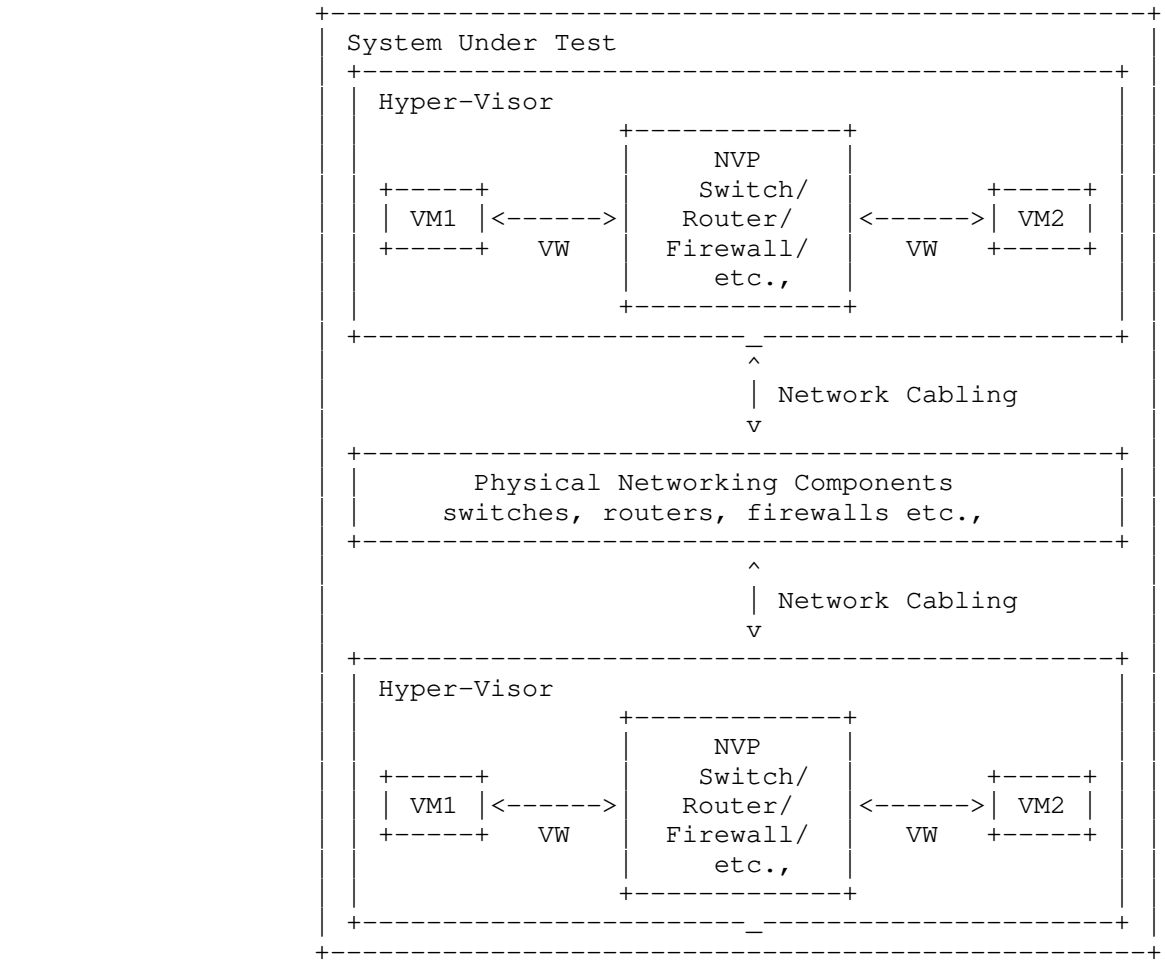


Figure 3! Inter-Host System Under Test

Virtual components have a direct dependency on the physical infrastructure that is hosting these resources. Hardware characteristics of the physical host impact the performance of the virtual components. The components that are being tested and the impact of the other hardware components within the hypervisor on th
e

performance of the SUT MUST be documented. Virtual component performance is influenced by the physical hardware components withi
n

the hypervisor. Access to various offloads such as TCP segmentation offload, may have significant impact on performance. Firmware and driver differences may also significantly impact results based on whether the specific driver leverages any hardware level offloads offered. Packet processing could be executed on shared or dedicated cores on the main processor or via a dedicated co-processor or embedded processor on NIC.

Hence, all physical components of the physical server running the hypervisor that hosts the virtual components MUST be documented along with the firmware and driver versions of all the components used to help ensure repeatability of test results. For example, BIOS configuration of the server MUST be documented as some of those changes are designed to improve performance. Please refer to Appendix A for a partial list of parameters to document.

5.1. Learning

SUT needs to learn all the addresses before running any tests. Address learning rate MUST be considered in the overall performance metrics because address learning rate has a high impact in microservices based use cases where there is huge churn of end points as they are created and destroyed on demand. In these cases, both the throughput at stable state, and the time taken to get to stable state MUST be tested and documented.

5.2. Traffic Flow Optimizations

Several mechanisms are employed to optimize traffic flows. Following are some examples:

5.2.1. Fast Path

A single flow may go through various switching, routing and firewalling decisions. While in the standard model, every single packet has to go through the entire process/pipeline, some optimizations help make this decision for the first packet, store the final state for that packet, and leverage it to skip the process for rest of the packets that are part of the same flow.

5.2.2. Dedicated cores / Co-processors

Packet processing is a CPU intensive workload. Some NVE's may use dedicated cores or a co-processor primarily for packet processing instead of sharing the cores used for the actual workloads. Such cases MUST be documented. Tests MUST be performed with both shared

and dedicated cores. Results and differences in results MUST be documented.

5.2.3. Prioritizing and de-prioritizing active flows

Certain algorithms may prioritize or de-prioritize traffic flows based on purely their network characteristics such as the length of the flow. For example, de-prioritize a long-lived flow. This coul
d
result in changing the performance of a flow over a period of time. Such optimizations MUST be documented, and tests MUST consist of long-lived flows to help capture the change in performance for such flows. Tests MUST note the point at which performance changes.

5.3. Server Architecture Considerations

When testing physical networking components, the approach taken is
to
consider the device as a black-box. With virtual infrastructure, this approach would no longer help as the virtual networking components are an intrinsic part of the hypervisor they are running on and are directly impacted by the server architecture used. Serve
r
hardware components define the capabilities of the virtual networki
ng
components. Hence, server architecture MUST be documented in detai
l
to help with repeatability of tests. And the entire hardware and software components become the SUT.

5.3.1. NVE Component considerations

5.3.1.1. NVE co-located

Components of NVE co-located may be hypervisor based or offloaded entirely to the NIC card or a hybrid model. In the case of hypervisor-based model, they may be running in user space or kernel space. Further, they may use dedicated cores, shared cores or in some cases dedicated co-processors. All the components and the process used MUST be documented.

5.3.1.2. NVE split

NVE split scenario generally has three primary components as documented per RFC 8394.

"tNVE: Terminal-side NVE. The portion of Split-NVE functionalitie
s
located on the end device supporting virtualization. The tNVE interacts with a Tenant System through an internal interface in the end device." tNVE may be made of either hypervisor controlled components such as hypervisor provided switches or NVE controlled

components where the network functionality is not provided by the hypervisor. In either case, the components used MUST be documented
.

"nNVE: Network-side NVE. The portion of Split-NVE functionalities located on the network device that is directly or indirectly connected to the end device that contains the corresponding NVE. Th
e
nNVE normally performs encapsulation to and decapsulation from the overlay network." All the functionality provided by the nNVE MUST
be
documented.

"External NVE: The physical network device that contains the nNVE.
"
Networking device hardware specs MUST be documented. Please use Apendix A for an example of the specs that MUST be documented.

In either case, NVE co-located or NVE split all the components MUST be documented. Where possible, individual components MUST be teste
d
independent of the entire system. For example, where possible, hypervisor provided switching functionality MUST be tested independent of the NVE.

Per RFC 8014, "For the split-NVE case, protocols will be needed tha
t
allow the hypervisor and NVE to negotiate and set up the necessary state so that traffic sent across the access link between a server and the NVE can be associated with the correct virtual network instance." Supported VM lifecycle events, from RFC 8394 section 2, MUST be documented as part of the benchmark process. This process MUST also include how the hypervisor and the external NVE have signaled each other to reach an agreement. Example, see section 2.
1
of RFC 8394 "VM creation event". The process used to update agreement status MUST also be documented.

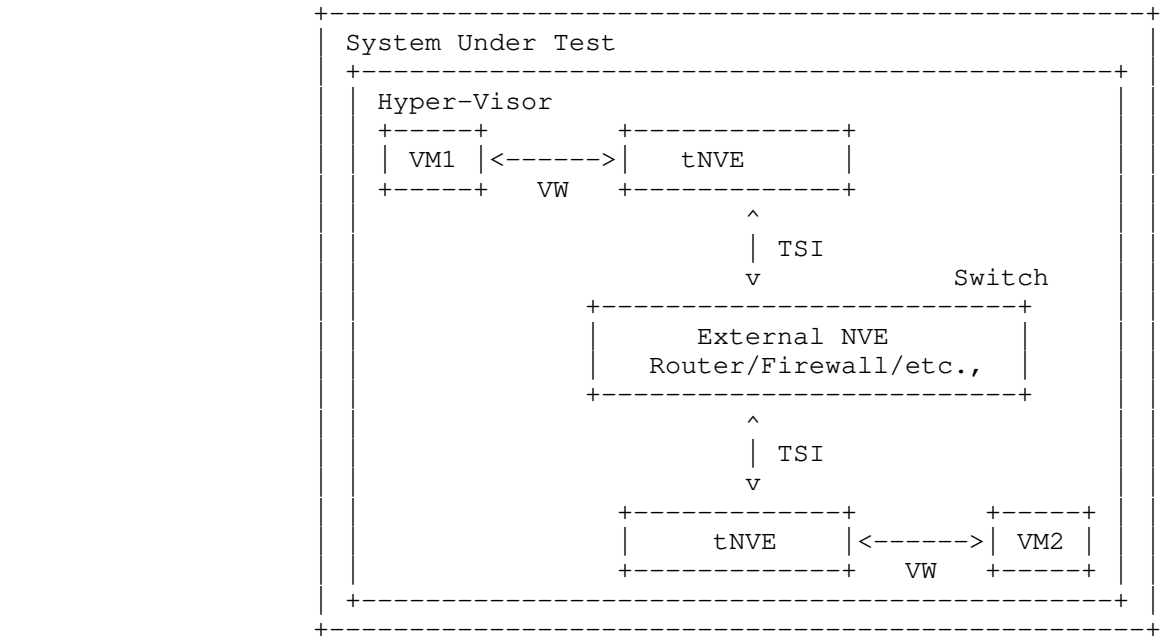


Figure 4 NVE Split collocated - System Under Test

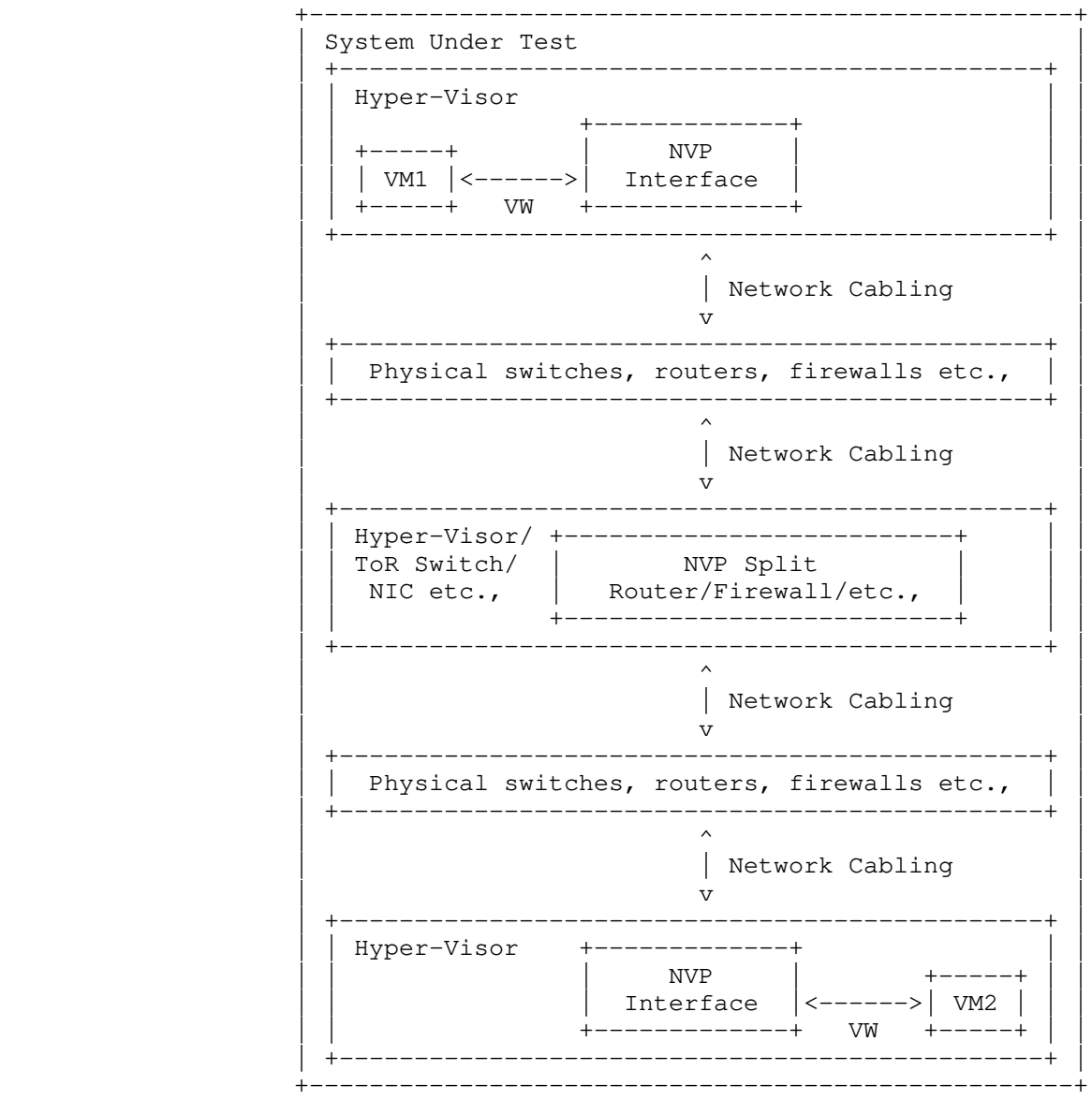


Figure 5 NVE Split not collocated - System Under Test

5.3.2. Frame format/sizes within the Hypervisor

Maximum Transmission Unit (MTU) limits physical network component's frame sizes. The most common max supported MTU for physical devices is 9000. However, 1500 MTU is the standard. Physical network testing and NFV uses these MTU sizes for testing. However, the virtual networking components that live inside a hypervisor, may work with much larger segments because of the availability of hardware and software based offloads. Hence, the normal smaller packets based testing is not relevant for performance testing of virtual networking components. All the TCP related configuration such as TSO size, number of RSS queues MUST be documented along with any other physical NIC related configuration.

NVE co-located may have a different performance profile when compared with NVE split because, the NVE co-located may have access to offloads that may not be available when the packet has to traverse the physical link. Such differences MUST be documented.

5.3.3. Baseline testing with Logical Switch

Logical switch is often an intrinsic component of the test system along with any other hardware and software components used for testing. Also, other logical components cannot be tested independent of the Logical Switch.

5.3.4. Repeatability

To ensure repeatability of the results, in the physical network component testing, much care is taken to ensure the tests are conducted with exactly the same parameters. Example parameters such as MAC addresses used.

When testing NVP components with an application layer test tool, there may be a number of components within the system that may not be available to tune or to ensure they maintain a desired state. Example: housekeeping functions of the underlying Operating System.

Hence, tests MUST be repeated a number of times and each test case MUST be run for at least 2 minutes if test tool provides such an option. Results SHOULD be derived from multiple test runs. Variance between the tests SHOULD be documented.

5.3.5. Tunnel encap/decap outside the Hypervisor

Logical network components may also have performance impact based on the functionality available within the physical fabric. Physical

fabric that supports NVO encap/decap is one such case that may have a different performance profile. Any such functionality that exists on the physical fabric MUST be part of the test result documentation to ensure repeatability of tests. In this case SUT MUST include the physical fabric if its being used for encap/decap operations.

5.3.6. SUT Hypervisor Profile

Physical networking equipment has well defined physical resource characteristics such as type and number of ASICs/SoCs used, amount of memory, type and number of processors etc., Virtual networking components performance is dependent on the physical hardware that hosts the hypervisor. Hence the physical hardware usage, which is part of SUT, for a given test MUST be documented, for example, CPU usage when running logical router.

CPU usage, changes based on the type of hardware available within the physical server. For example, TCP Segmentation Offload greatly reduces CPU usage by offloading the segmentation process to the NIC card on the sender side. Receive side scaling offers similar benefit on the receive side. Hence, availability and status of such hardware MUST be documented along with actual CPU/Memory usage when the virtual networking components have access to such offload capable hardware.

Following is a partial list of components that MUST be documented both in terms of what is available and also what is used by the SUT

- CPU - type, speed, available instruction sets (e.g. AES-NI)
- Memory - type, amount
- Storage - type, amount
- NIC Cards -
  - Type
  - number of ports
  - offloads available/used - following is a partial list of possible features
    - TCP Segmentation Offload
    - Large Receive Offload

- o Checksum Offloads
- o Receive Side Scaling
- o Other Queuing Mechanisms

  * drivers, firmware (if applicable)
  * HW revision

- o Libraries such as DPDK if available and used
- o Number and type of VMs used for testing and
  * vCPUs
  * RAM
  * Storage
  * Network Driver
  * Any prioritization of VM resources
  * Operating System type, version and kernel if applicable
  * TCP Configuration Changes - if any
  * MTU
- o Test tool
  * Workload type
  * Protocol being tested
  * Number of threads
  * Version of tool
- o For inter-hypervisor tests,
  * Physical network devices that are part of the test
    - o Note: For inter-hypervisor tests, system under test is no longer only the virtual component that is being

tested but the entire fabric that connects the virtual components become part of the system under test.

5.4. Benchmarking Tools Considerations

5.4.1. Considerations for NVE

Virtual network components in NVE work closer to the application layer then the physical networking components, which enables the virtual network components to take advantage of TCP optimizations such as TCP Segmentation Offload (TSO) and Large Receive Offload (LRO). Because of this optimizations, virtual network components work with type and size of segments that are often not the same type and size that the physical network works with. Hence, testing virtual network components MUST be done with application layer segments instead of the physical network layer packets. Testing MUST be done with application layer testing tools such as iperf, netperf etc.,

5.4.2. Considerations for Split-NVE

In the case of Split-NVE, since they may not leverage any TCP related optimizations, typical network test tools focused on packet processing MUST be used. However, the tools used MUST be able to leverage Receive Side Scaling (RSS).

6. Control Plane Scale Considerations

For a holistic approach to performance testing, control plane performance must also be considered. While the previous sections focused on performance tests after the SUT has come to a steady state, the following section focusses on tests to measure the time taken to bring the SUT to steady state.

In a physical network infrastructure world view, this could be various stages such as boot up time, time taken to apply configuration, BGP convergence time etc., In a virtual infrastructure world, this involves lot more components which may also be distributed across multiple hosts. Some of the components are:

- o VM Creation Event
- o VM Migration Event
- i How many total VMs can the SUT support

- What is the rate at which the SUT would allow creation of VMs

Please refer to section 2 of RFC 8394 for various VM events and their definitions. In the following section we further clarify some of the terms used in the above RFC.

VM Creation

For the purposes of NVP control plane testing, VM Creation event is when a VM starts participating for the first time on a NVP provided network. This involves various actions on the tNVE and NVP. Please refer to 2.1 "VM Creation Event" of RFC 8394 for more details.

In order to rule out any Hypervisor imposed limitations, System Under Test must first be profiled and baselined with-out the use of NVP components. For the purposes of baselining control plane, the VM used may have very small footprint such as DSL Linux which runs in 16MB RAM.

Once a baseline has been established for a single HV, a similar exercise MUST be done on multiple HVs to establish a baseline for the entire hypervisor domain. However, it may not be practical to have physical hosts and hence nested hosts may be used for this purpose

6.1.1. VM Events

Performance of various control plane activities which are associated with the System Under Test, MUST BE documented.

- VM Creation: Time taken to join the VMs to the SUT provided network
- Policy Realization: Time taken for policy realization on the VM
- VM Migration: Time taken to migrate a VM from one SUT provided network to another SUT provided network

For the test itself, the following process could be use:

1 API to call to join VM on the SUT provided network

2 Loop while incrementing a timer - till the VM comes online on the SUT provided network

Similarly, policy realization and VM migration may also be tested with a check on whether the VM is available or not available based on the type of policy that is applied.

6.1.2. Scale

SUT must also be tested to determine the maximum scale supported. Scale can be multi-faceted such as the following:

- Total # of VMs per Host
- Total # of VMs per one SUT Domain
- Total # of Hosts per one SUT Domain
- Total # of Logical Switches per one SUT Domain
    - Total # of VMs per one SUT provided Logical Switch
        - Per Host
        - Per SUT Domain
- Total # of Logical Routers per one SUT Domain
    - Total # of Logical Switches per one Logical Router
    - Total # of VMs on a single Logical Router
- Total # of Firewall Sections
- Total # of Firewall Rules per Section
- Total # of Firewall Rules applied per VM
- Total # of Firewall Rules applied per Host
- Total # of Firewall Rules per SUT

6.1.3. Control Plane Performance at Scale

Benchmarking MUST also test and document the control performance at scale. That is,

- Total # VMs that can be created in parallel
    - How long does the action take
- Total # of VMs that can be migrated in parallel
    - How long does the action take

- Total amount of time taken to apply 1 firewall across the entire VMs under a SUT
- Time taken to apply 1000s rules on a SUT

7. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization of a Device Under Test/System Under Test (DUT/SUT) using controlled stimuli in a laboratory environment, with dedicated address space and the constraints specified in the sections above.

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network.

Further, benchmarking is performed on a 'black-box' basis, relying solely on measurements observable external to the DUT/SUT.

Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

8. IANA Considerations

No IANA Action is requested at this time.

9. Conclusions

Network Virtualization Platforms, because of their proximity to the application layer and since they can take advantage of TCP stack optimizations, do not function on packets/sec basis. Hence, traditional benchmarking methods, while still relevant for Network Function Virtualization, are not designed to test Network Virtualization Platforms. Also, advances in application architectures such as micro-services, bring new challenges and need benchmarking not just around throughput and latency but also around scale. New benchmarking methods that are designed to take advantage of the TCP optimizations or needed to accurately benchmark performance of the Network Virtualization Platforms

10. References

10.1. Normative References

[RFC7364] T. Narten, E. Gray, D. Black, L. Fang, L. Kreeger, M. Napierala, 'Problem Statement: Overlays for Network Virtualization', RFC 7364, October 2014, https://datatracker.ietf.org/doc/rfc7364/

[RFC8014] D. Black, J. Hudson, L. Kreeger, M. Lasserre, T. Narten 'An Architecture for Data-Center Network Virtualization over Layer 3 (NVO3)', RFC 8014, December 2016, https://tools.ietf.org/html/rfc8014

[RFC8394] Y. Li, D. Eastlake 3rd, L. Kreeger, T. Narten, D. Black 'Split Network Virtualization Edge (Split-NVE) Control-Plane Requirements', RFC 8394, May 2018, https://tools.ietf.org/html/rfc8394

[nv03] IETF, WG, Network Virtualization Overlays, <https://datatracker.ietf.org/wg/nvo3/documents/>

10.2. Informative References

[RFC8172] A. Morton 'Considerations for Benchmarking Virtual Network Functions and Their Infrastructure', RFC 8172, July 2017, https://tools.ietf.org/html/rfc8172

11. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

Appendix A.!Partial List of Parameters to Document

A.1. CPU

CPU Vendor

CPU Number

CPU Architecture

# of Sockets (CPUs)

# of Cores

Clock Speed (GHz)

Max Turbo Freq. (GHz)

Cache per CPU (MB)

# of Memory Channels

Chipset

Hyperthreading (BIOS Setting)

Power Management (BIOS Setting)

VT-d

Shared vs Dedicated packet processing

User space vs Kernel space packet processing

A.2. Memory

Memory Speed (MHz)

DIMM Capacity (GB)

# of DIMMs

DIMM configuration

Total DRAM (GB)

A.3. NIC

Vendor

Model

Port Speed (Gbps)

Ports

PCIe Version

PCIe Lanes

Bonded

Bonding Driver

Kernel Module Name

Driver Version

VXLAN TSO Capable

VXLAN RSS Capable

Ring Buffer Size RX

Ring Buffer Size TX

A.4. Hypervisor

Hypervisor Name

Version/Build

Based on

Hotfixes/Patches

OVS Version/Build

IRQ balancing

vCPUs per VM

Modifications to HV

Modifications to HV TCP stack

Number of VMs

IP MTU

Flow control TX (send pause)

Flow control RX (honor pause)

Encapsulation Type

A.5. Guest VM

Guest OS & Version

Modifications to VM

IP MTU Guest VM (Bytes)

Test tool used

Number of NetPerf Instances

Total Number of Streams

Guest RAM (GB)

A.6. Overlay Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

A.7. Gateway Network Physical Fabric

Vendor

Model

and Type of Ports

Software Release

Interface Configuration

Interface/Ethernet MTU (Bytes)

Flow control TX (send pause)

Flow control RX (honor pause)

A.8. Metrics

Drops on the virtual infrastructure

Drops on the physical underlay infrastructure

Authors' Addresses

Samuel Kommu
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: skommu@vmware.com

Jacob Rapp
VMware
3401 Hillview Ave
Palo Alto, CA, 94304

Email: jrapp@vmware.com

Benchmarking Workgroup
Internet-Draft
Intended status: Informational
Expires: May 19, 2018

S. Wu
Juniper Networks

November 15, 2017

Network Service Layer Abstract Model
draft-xwu-bmwg-nslam-00

Abstract

While the networking technologies have evolved over the years, the layered approach has been dominant in many network solutions. Each layer may have multiple interchangeable, competing alternatives that deliver a similar set of functionality. In order to provide an objective benchmarking data among various implementations, the need arises for a common abstract model for each network service layer, with a set of required and optional specifications in respective layers. Many overlay and or underlay solutions can be described using these models.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 19, 2018.

Table of Contents

1. Introduction

This document provides a reference model for common network service framework. The main purpose is to abstract service model for each network layer with a small set of key specifications. This is essential to characterize the capability and capacity of a production network, a target network design. A complete service model mainly includes

Infrastructure - devices, links, and other equipment.

Services - network applications provisioned. It is often defined as device configuration and or resource allocation.

Capacity - A set of objects dynamically created for both control and forwarding planes, such as routes, traffic, subscribers and

etc. In some cases, the amount and types of traffic may impact control plane objects, such as multicast or ethernet networks.

Performance Metrics - infrastructure resource utilization.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Purpose of The Document

Many efforts to YANG model and OpenConfig collaboration are well under way. This document specifies a higher layer abstraction that reuses a small subset of YANG keywords for service description purpose. It SHALL NOT be used for production provisioning purpose. Instead, it can be adopted for design spec, capacity planning, product benchmarking and test setup.

The specification described in this document SHALL be used for outline service requirements from customer perspective, instead of network implementation mechanism from operators perspective.

1.3. Conventions Used in This Document

Descriptive terms can quickly become overloaded. For consistency, the following definitions are used.

- Node - The name for an attrubute.
- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

2. Network Service Framework

The network service layer abstract model is illustrated by Figure 1. This shows a stack of components to enable end-to-end services. Not all components are required for a given service. A common use case is to pick one component as target service with a detailed profile, with the remaining components as supporting technologies using default profiles.

Network Service Layer Abstract Model

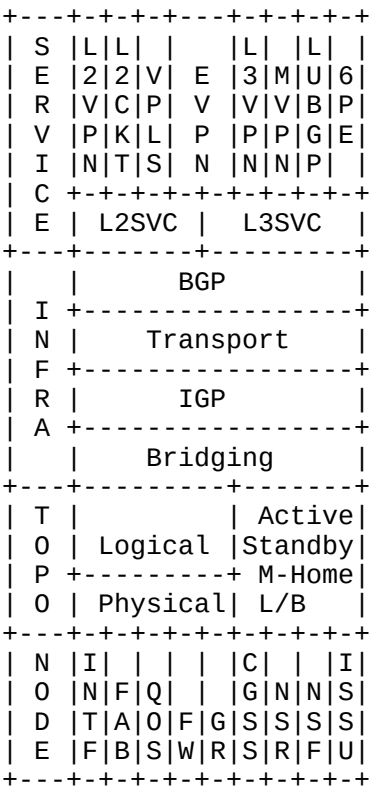


Figure 1

2.1. Node

A network node or a network device processes and forwards traffic based on predefined or dynamically learned policies. This section covers standalone features like the following:

- INTF - Network interfaces that provides internal or external connectivity to adjacent devices. Only physical properties of the interfaces are of concern in this level. The interfaces can be physical or logical, wired or wireless.

- FAB - Fabric capacity. It provides redundancy and cross connect within the same network device among various linecards or interfaces

- QOS - Quality of Services. Traffic queuing, buffering, and congestion management technologies are used in this level

- FW - Firewall filters or access control list. This commonly refers to stateless packet inspection and filtering. Stateful firewall is out of scope of this document. Number of filters daisy chained on a given protocol family, number of terms within same filter, and depth of packet inspection can all affect speed and latency of traffic forwarding. It also provides necessary security protection of node, where protocol traffic may be affected.

- GR - Graceful Restart per protocol. It needs to cooperate with adjacent node

- CGSS - Controller Graceful / Stateful Switchover. A network device often has two redundant controllers to minimize the disruption in event of catastrophic failure on the primary controller. This is accomplished via real time state synchronization from the primary to the backup controller. It, however, should be used along with either NSR or NSF to achieve optimal redundancy.

- NSR - Non-Stop Routing - hitless failover of route processor. It maintains an active copy of route information base (RIB) as well as state for protocol exchange so that the protocol adjacency is not reset.

- NSF - Non-Stop Forwarding for layer 2 traffic, including layer 2 protocols such as spanning tree state

- ISSU - In Service Software Upgrade - Sub-second traffic loss in many modern routing platforms. The demand for this feature continues to grow from the field. Some study shows that the downtime due to software upgrade is greater than that caused by unplanned outages.

2.2. Topology

Placement of network devices and corresponding links plays an important role for optimal traffic forwarding. There are two types of topologies:

- Physical Topology - Actual physical connectivity via fiber, coax, cat5 or even wireless. That could be a ring, bus, star or matrix topology. Even though all can be modeled using point-to-point connections.

- Logical Topology - With aggregated ethernet, extended dot1q tunneling, or VxLAN, a logical or virtual topology can be easily created spanning across geography boundaries. Recent development of multi-chassis, virtual-chassis, node-slicing technologies, and multiple logical units within a single physical node have enabled logical topology deployment more flexible and agile.

With various topology, the following funtionalities need to be taken into consideration for feature design and validation.

- Active-Standby - 1:1 or 1:n support. The liveness detection is essential to trigger failover.

- M-Home - Multi-homing support. A customer edge (CE) device can be homed to 2 or more Provider Edge (PE) devices at the same time. This is a common redundancy design in layer 2 service offering

- L/B - Load Balancing - When multiple diverse paths exist for a given destination, it is important to achieve load balancing based on multiple criteria, such as per packet, or per prefix. Sometimes, cascading effect can make issues more complex and harder to resolve

The topology, regardless of physical or virtual, can be better depicted using a collection of nodes and point-to-point links. Some broadcast network, or ring topology, can also be abstracted using same collection of point-to-point links. For example, in a wireless LAN network, each client is a node with wireless LAN NIC as its physical interface. The access point is the node, at which all WLAN cients terminate with airwaves. The Service Set IDentifier (SSID) on this access point can be considered as part of broadcast domain with many pseudo-ports taking airwave terminations from clients.

The default link id can use srcnode-dstnode-linkseq to uniquely identify a link in this topology. If this is a link connecting two ports on the same node, it can use link-id of srcnode-srcnode-linkseq-portseq. Additional attributes of the node can be added with proper placement for auto topology diagram.

Network Topology Definition

```
node-id-1 {
    maker: maker_name,
    model: model_name,
    controller: controller-type,
    mgmt_ip: mgmt_ip_address,
    links: {
        link-id-1 {
            name: link_name,
            connector: 'sfpp',
            attr: ['10G', 'Ethernet'],
            node_dst: destination node-id,
            link_seq: sequence number for links between the node pair
            ...
        }
    }
    ...
}
node-id-2 {
    ...
}
```

Figure 2

2.3. Infrastructure

Network infrastructure here refers to a list of protocols and policies for a data center network, an enterprise network, or a core backbone in a service provider network.

- Bridging - Spanning Tree Protocol (STP) and its various flavors, 802.1q tunneling, Q-in-Q, VRRP and etc

- IGP - Interior Gateway Protocol - some common choices are OSPF, IS-IS, RIP, RIPng. For multicast, choices are PIM and its various flavors including MSDP, Bootstrap, DVMRP

- Transport - Tunnel technologies including

  * MPLS - Multi-Protocol Label Switching - most commonly used in service provider network

    + LDP - Label distribution protocol - including mLDP and LDP Tunneling through RSVP LSPs

    + RSVP - Resource Reservation Protocol - including P2MP and its various features like Fast ReRoute - FRR.

    * IPSec - IPSec Tunnel with AH or ESP

    * GRE - Generic Routing Encapsulation (GRE) tunnels provides a flexible direct adjacency between two remote routers

    * VxLAN - In data center interconnect (DCI) solutions, VxLAN encapsulation provides data plane for layer 2 frames

  o BGP - Define families and their sub-SAFI deployed, as well as route reflector topology.

2.4. Services

Previous sections mostly outline an operator's implementation of the network, while customers may not necessarily care about these. This section defines service profiles from customer's view.

  o Layer 2 Services

    * Layer 2 VPN - RFC 6624

    * Martini Layer 2 Circuit - RFC 4906

    * Virtual Private LAN Services - RFC 4761

    * Ethernet VPN - RFC 7432

  o Layer 3 Services

    * Type 5 Route for EVPN - draft-ietf-bess-evpn-prefix-advertisement-05

    * Layer 3 VPN - RFC 4364

    * Labled Unicast BGP - RFC 3107

    * Draft Rosen MVPN - RFC 6037

    * NG MVPN - RFC 6513

    * 6PE - RFC 4798

In next section, an abstract model is proposed to identify key metrics for both layer 2 and layer 3 model

3. Service Models

A service model is a high level abstraction of network deployment from bottom up. It defines a set of common key characteristics of customer traffic profile in both control and forwarding planes. The network itself should be considered as a blackbox and deliver the services regardless of types of network equipment vendor or network technologies.

The abstraction removes some details like specific IP address assignment, and favors address range and its distribution. The goal is to describe aggregated network behavior instead of granular network element configuration. It is up to implementation to map aggregated metrics to actual configuration for the network devices, protocol emulator and traffic genrator.

A single network may be comprised of multiple instances of service models defined below.

3.1. Layer 2 Ethernet Service Model

The metrics outlined below are for layer 2 network services typically within a data center, data center interconnect, metro ethernet, or layer 2 domain over WAN or even inter-carrier.

- service-type: identityref, ELAN, ELINE, ETREE
- sites-per-instance: uint32, an avrage number of sites a layer 2 instance may span across
- global-mac-count: uint32, Global MAC from all attachment circuits, local and remote. This is probably the most important metric that determines the capacity requirements in layer 2 for both control and forwarding planes
- interface-mac-max: uint32, maxiumum number of locally learned MAC addresses per logical interface, aka attachment circuit
- single-home-segments: uint32, number of single homed ethernet segments per service instance
- multi-home-segments: uint32, number of multi homed ethernet segments per layer 2 service instance
- service-instance-count: uint32, total number of layer 2 service instances. Typically, one customer is
- traffic-type: list, {known-unicast: %, multicast, %, broadcast: %, unknown-unicast: %}
- traffic-frame-size: list, predefined mixture of traffic frame size distribution
- traffic-load: speed of traffic being sent towards the network. This can be defined as frame per second (fps), or actual speed in bps. This is particular important whenever some component along

forwarding path is implemented in software, the throughput might be affected significantly at high speed
- traffic-flow: A distribution of flows. This may affect efficient use of load-balancing techniques and resource consumption. More details discussed in later section of this document.
- layer3-gateway-count: uint32, number of layer 2 service instances that also provide layer 3 gateway service
- arpnp-table-size: uint32. This is only relevant with presence of layer 3 gateway

Integrated routing and bridging (IRB) and EVPN Type 5 route have blurred boundaries between layer 2 and layer 3 services.

3.2. Layer 3 Service Model

This section outlines traffic type, layer 3 protocol families, layer 3 prefixes distribution, layer 3 traffic flow and packet size distributions.

- proto-family: protocol family are defined with three sub-attributes. The list may grow as the complexity
  - proto - list: inet, inet6, iso
  - type - list: unicast, mcast, segment, labeled
  - vpn - list, true, false
- prefix-count, uint32, total unique prefixes
- prefix-distrib, list of prefix length size and percentage. This could be a distribution pattern, such as uniform, random. Or simply top representation of prefix lengths
- bgp-path-count, uint32, total BGP paths
- bgp-path-distrib, top representation of number of paths per prefix
- traffic-frame-size, similar to traffic-frame-size in layer 2 model. The focus is on the MTU size on each protocol interfaces and the impact of fragmentation
- traffic-flow, similar to traffic-flow in layer 2 model, it focuses on a set of labels, source and destination addresses as well as ports
- traffic-load, similar to traffic-load in layer 2 model
- ifl-count, uint32,
- vpn-count, uint32,

3.3. Infrastructure Service Model

- bgp-peer-ext-count, uint32, number of eBGP peers
- bgp-peer-int-count, uint32, number of iBGP peers
- bgp-path-mtu, list, true or false. Larger path mtu helps convergence

- o bgp-hold-time-distrib, list of top hold-time values and their respective percentage out of all peers.
- o bgp-as-path-distrib, list of top as-path lengths and their respective percentage of all BGP paths
- o bgp-community-distrib, list of top community size and their respective percentage out of all BGP paths
- o mpls-sig, list, MPLS signaling protocol, rsvp or ldp
- o rsvp-lsp-count-ingress, uint32, total ingress lsp count
- o rsvp-lsp-count-transit, uint32, total transit lsp count
- o rsvp-lsp-count-egress, uint32, total egress lsp count
- o ldp-fec-count, uint32, total forwarding equivalence class
- o rsvp-lsp-protection, list, link-node, link, frr
- o ospf-interface-type, list, point-to-point, broadcast, non-broadcast multi-access
- o ospf-lsa-distrib, list. OSPF Link Statement Advertisement distribution is comprised of those for core router in backbone area, and internal router in non-Backbone areas. A common modeling can include number of LSAs per OSPF LSA type
- o ospf-route-count, list, total OSPF routes in both backbone and non-backbone areas
- o isis-lsp-distrib, list, similar to ospf-lsa-distrib
- o isis-route-count, list, total IS-IS routes in both level-1 and level-2 areas

TODO: bridging, OAM, EOAM, BFD and etc

3.4. Node Level Features

TODO: node level feature set

3.5. Common Service Specification

For most network services, regardless of layer 2 or layer 3, protocol families, the following needs to be considered when measuring network capacity and baseline.

- o rib-learning-time, uint32 in seconds. This indicates show quickly the route processor learns routing objects either locally and remotely
- o fib-learning-time. In large routing system, forwarding engine residing on separate hardware from controller, takes additional time to install all forwarding entries learned by controller.
- o convergence-time, this is could be as a result of many events, such as uplink failure, ae member link failure, fast reroute, local repair, upstream node failure and etc
- o multihome-failover-time, this refers to traffic convergence in a topology where a customer edge (CE) device is connected to two or more provider edge (PE) devices.

- o issu-dark-window-size. Unlike NSR, the goal of ISSU is not zero packet loss. Instead, there will be a few seconds, or in some cases, sub-second dark window where it sees both total packet loss for both transit and or host bound protocol traffic.
- o cpu-util, total CPU utilization of the controllers in stead state
- o cpu-util-peak, Peak CPU utilization on the controller in event of failure, and convergence
- o mem-util, total memory utilization of the controllers in steady state
- o mem-util-peak, total memory utilization on the controller in event of failure and convergence
- o processes, list of top processes running on the controllers with their CPU and memory utilization.
- o lc-cpu-util, top CPU utilization on the line cards
- o lc-cpu-util-peak, maximum peak CPU utilization among all line cards in event of failure and convergence
- o lc-mem-util, top memory utilization on the line cards
- o lc-mem-util-peak, maximum peak memory utilization among all line cards in event of failure and convergence
- o throughout, in both pps and bps. This is measured with zero packet loss. For virtualized environment, throughput is sometimes measured with a small loss tolerance given the nature of shared resource
- o traffic performance, in both pps and bps. It is measured the rate of traffic received by pumping oversubscribed traffic at ingress
- o latency in us. this is more important within a local data center environment rather than DCI over wide area network. Use of extensive firewall filter or access control lists may affect latency
- o Out of Order Packet - This can happen in intra-node or over ECMP where different paths have large latency/delay variations.

The list of metrics can be used for network monitoring during network resiliency test. This is to understand how quickly a network service can restore during various events and failures

3.6. Comment Network Events

TODO: A list of events to be defined to characterize network resiliency. These attributes require the provider networks have diverse paths and node redundancy built-in. These numbers affect service level agreement and network availability

4. Use of Network Service Layer Abstract Model

The primary goal is to characterize and document a complex network using a simplified service model. While eliminating many details such as address assignment, actual route or mac entries, it retains a

set of key network information, including services, scale, and performance profiles. This can be used to validate how well each underlying solution performs when delivering same set of services.

The model can also be used to build a virtualized topology with both static and dynamic scale closely resemble to a real network. This eases network design and benchmarking, and helps capacity planning by studying the impact with changes to a specific dimension.

5. Acknowledgements

The authors appreciate and acknowledge comments from Al Morton and others based on initial discussions.

6. IANA Considerations

This memo includes no request to IANA.

All drafts are required to have an IANA considerations section (see Guidelines for Writing an IANA Considerations Section in RFCs [RFC5226] for a guide). If the draft does not require IANA to do anything, the section contains an explicit statement that this is the case (as above). If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

7. Security Considerations

All drafts are required to have a security considerations section. See RFC 3552 [RFC3552] for a guide.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

8.2. Informative References

[L2SM] B. Wu et al, "A Yang Data Model for L2VPN Service Delivery", 2017, <https://www.ietf.org/id/draft-ietf-l2sm-l2vpn-service-model-02.txt>.

[RFC8049] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8049, DOI 10.17487/RFC8049, February 2017, <https://www.rfc-editor.org/info/rfc8049>.

Author’s Address

Sean Wu
Juniper Networks
2251 Corporate Park Dr.
Suite #200
Herndon, VA 20171
US

Phone: +1 571 203 1898
Email: xwu@juniper.net

Benchmarking Workgroup
Internet-Draft
Intended status: Informational
Expires: November 19, 2018

S. Wu
Juniper Networks

May 18, 2018

Network Service Layer Abstract Model
draft-xwu-bmwg-nslam-01

Abstract

While the networking technologies have evolved over the years, the layered approach has been dominant in many network solutions. Each layer may have multiple interchangeable, competing alternatives that deliver a similar set of functionality. In order to provide an objective benchmarking data among various implementations, the need arises for a common abstract model for each network service layer, with a set of required and optional specifications in respective layers. Many overlay and or underlay solutions can be described using these models.

Table of Contents

1. Introduction

This document provides a reference model for common network service framework. The main purpose is to abstract service model for each network layer with a small set of key specifications. This is essential to characterize the capability and capacity of a production network, a target network design. A complete service model mainly includes

Infrastructure - devices, links, and other equipment.

Services - network applications provisioned. It is often defined as device configuration and or resource allocation.

Capacity - A set of objects dynamically created for both control and forwarding planes, such as routes, traffic, subscribers and etc. In some cases, the amount and types of traffic may impact control plane objects, such as multicast or ethernet networks.

Performance Metrics - infrastructure resource utilization.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Purpose of The Document

Many efforts to YANG model and OpenConfig collaboration are well under way. This document specifies a higher layer abstraction that reuses a small subset of YANG keywords for service description purpose. It SHALL NOT be used for production provisioning purpose. Instead, it can be adopted for design spec, capacity planning, product benchmarking and test setup.

The specification described in this document SHALL be used for outline service requirements from customer perspective, instead of network implementation mechanism from operators perspective.

1.3. Conventions Used in This Document

Descriptive terms can quickly become overloaded. For consistency, the following definitions are used.

- Node - The name for an attrubute.
- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

2. Network Service Framework

The network service layer abstract model is illustrated by Figure 1. This shows a stack of components to enable end-to-end services. Not all components are required for a given service. A common use case is to pick one component as target service with a detailed profile, with the remaining components as supporting technologies using default profiles.

Network Service Layer Abstract Model

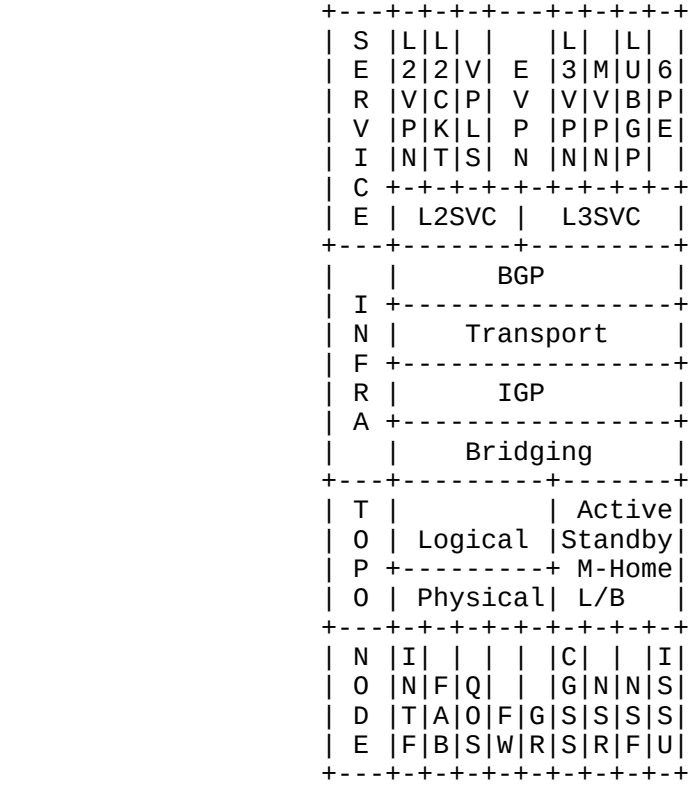


Figure 1

2.1. Node

A network node or a network device processes and forwards traffic based on predefined or dynamically learned policies. This section covers standalone features like the following:

- INTF - Network interfaces that provides internal or external connectivity to adjacent devices. Only physical properties of the interfaces are of concern in this level. The interfaces can be physical or logical, wired or wireless.

- FAB - Fabric capacity. It provides redundancy and cross connect within the same network device among various linecards or interfaces

- QOS - Quality of Services. Traffic queuing, buffering, and congestion management technologies are used in this level

- FW - Firewall filters or access control list. This commonly refers to stateless packet inspection and filtering. Stateful firewall is out of scope of this document. Number of filters daisy chained on a given protocol family, number of terms within same filter, and depth of packet inspection can all affect speed and latency of traffic forwarding. It also provides necessary security protection of node, where protocol traffic may be affected.

- GR - Graceful Restart per protocol. It needs to cooperate with adjacent node

- CGSS - Controller Graceful / Stateful Switchover. A network device often has two redundant controllers to minimize the disruption in event of catastrophic failure on the primary controller. This is accomplished via real time state synchronization from the primary to the backup controller. It, however, should be used along with either NSR or NSF to achieve optimal redundancy.

- NSR - Non-Stop Routing - hitless failover of route processor. It maintains an active copy of route information base (RIB) as well as state for protocol exchange so that the protocol adjacency is not reset.

- NSF - Non-Stop Forwarding for layer 2 traffic, including layer 2 protocols such as spanning tree state

- ISSU - In Service Software Upgrade - Sub-second traffic loss in many modern routing platforms. The demand for this feature continues to grow from the field. Some study shows that the downtime due to software upgrade is greater than that caused by unplanned outages.

2.2. Topology

Placement of network devices and corresponding links plays an important role for optimal traffic forwarding. There are two types of topologies:

- Physical Topology - Actual physical connectivity via fiber, coax, cat5 or even wireless. That could be a ring, bus, star or matrix topology. Even though all can be modeled using point-to-point connections.

- Logical Topology - With aggregated ethernet, extended dot1q tunneling, or VxLAN, a logical or virtual topology can be easily created spanning across geography boundaries. Recent development of multi-chassis, virtual-chassis, node-slicing technologies, and multiple logical units within a single physical node have enabled logical topology deployment more flexible and agile.

With various topology, the following functionalities need to be taken into consideration for feature design and validation.

- Active-Standby - 1:1 or 1:n support. The liveness detection is essential to trigger failover.

- M-Home - Multi-homing support. A customer edge (CE) device can be homed to 2 or more Provider Edge (PE) devices at the same time. This is a common redundancy design in layer 2 service offering

- L/B - Load Balancing - When multiple diverse paths exist for a given destination, it is important to achieve load balancing based on multiple criteria, such as per packet, or per prefix. Sometimes, cascading effect can make issues more complex and harder to resolve

The topology, regardless of physical or virtual, can be better depicted using a collection of nodes and point-to-point links. Some broadcast network, or ring topology, can also be abstracted using same collection of point-to-point links. For example, in a wireless LAN network, each client is a node with wireless LAN NIC as its physical interface. The access point is the node, at which all WLAN cients terminate with airwaves. The Service Set IDentifier (SSID) on this access point can be considered as part of broadcast domain with many pseudo-ports taking airwave terminations from clients.

The default link id can use srcnode-dstnode-linkseq to uniquely identify a link in this topology. If this is a link connecting two ports on the same node, it can use link-id of srcnode-srcnode-

linkseq-portseq. Additional attributes of the node can be added with proper placement for auto topology diagram.

Network Topology Definition

```
node-id-1 {
    maker: maker_name,
    model: model_name,
    controller: controller-type,
    mgmt_ip: mgmt_ip_address,
    links: {
        link-id-1 {
            name: link_name,
            connector: 'sfpp',
            attr: ['10G', 'Ethernet'],
            node_dst: destination node-id,
            link_seq: sequence number for links between the node pair
            ...
        }
    }
    ...
}
node-id-2 {
    ...
}
```

Figure 2

2.3. Infrastructure

Network infrastructure here refers to a list of protocols and policies for a data center network, an enterprise network, or a core backbone in a service provider network.

- Bridging - Spanning Tree Protocol (STP) and its various flavors, 802.1q tunneling, Q-in-Q, VRRP and etc
- IGP - Interior Gateway Protocol - some common choices are OSPF, IS-IS, RIP, RIPng. For multicast, choices are PIM and its various flavors including MSDP, Bootstrap, DVMRP
- Transport - Tunnel technologies including
  - MPLS - Multi-Protocol Label Switching - most commonly used in service provider network
    - LDP - Label distribution protocol - including mLDP and LDP Tunneling through RSVP LSPs

        + RSVP - Resource Reservation Protocol - including P2MP and its various features like Fast ReRoute - FRR.

      * IPSec - IPSec Tunnel with AH or ESP

      * GRE - Generic Routing Encapsulation (GRE) tunnels provides a flexible direct adjacency between two remote routers

      * VxLAN - In data center interconnect (DCI) solutions, VxLAN encapsulation provides data plane for layer 2 frames

   o BGP - Define families and their sub-SAFI deployed, as well as route reflector topology.

2.4. Services

Previous sections mostly outline an operator's implementation of the network, while customers may not necessarily care about these. This section defines service profiles from customer's view.

   o Layer 2 Services

      * Layer 2 VPN - RFC 6624

      * Martini Layer 2 Circuit - RFC 4906

      * Virtual Private LAN Services - RFC 4761

      * Ethernet VPN - RFC 7432

   o Layer 3 Services

      * Type 5 Route for EVPN - draft-ietf-bess-evpn-prefix-advertisement-05

      * Layer 3 VPN - RFC 4364

      * Labled Unicast BGP - RFC 3107

      * Draft Rosen MVPN - RFC 6037

      * NG MVPN - RFC 6513

      * 6PE - RFC 4798

In next section, an abstract model is proposed to identify key metrics for both layer 2 and layer 3 model

3. Service Models

A service model is a high level abstraction of network deployment from bottom up. It defines a set of common key characteristics of customer traffic profile in both control and forwarding planes. The network itself should be considered as a blackbox and deliver the services regardless of types of network equipment vendor or network technologies.

The abstraction removes some details like specific IP address assignment, and favors address range and its distribution. The goal is to describe aggregated network behavior instead of granular network element configuration. It is up to implementation to map aggregated metrics to actual configuration for the network devices, protocol emulator and traffic genrator.

A single network may be comprised of multiple instances of service models defined below.

3.1. Layer 2 Ethernet Service Model

The metrics outlined below are for layer 2 network services typically within a data center, data center interconnect, metro ethernet, or layer 2 domain over WAN or even inter-carrier.

- service-type: identityref, ELAN, ELINE, ETREE
- sites-per-instance: uint32, an average number of sites a layer 2 instance may span across
- global-mac-count: uint32, Global MAC from all attachment circuits, local and remote. This is probably the most important metric that determines the capacity requirements in layer 2 for both control and forwarding planes
- interface-mac-max: uint32, maximum number of locally learned MAC addresses per logical interface, aka attachment circuit
- single-home-segments: uint32, number of single homed ethernet segments per service instance
- multi-home-segments: uint32, number of multi homed ethernet segments per layer 2 service instance
- service-instance-count: uint32, total number of layer 2 service instances. Typically, one customer is
- traffic-type: list, {known-unicast: %, multicast, %, broadcast: %, unknown-unicast: %}
- traffic-frame-size: list, predefined mixture of traffic frame size distribution
- traffic-load: speed of traffic being sent towards the network. This can be defined as frame per second (fps), or actual speed in bps. This is particular important whenever some component along

forwarding path is implemented in software, the throughput might be affected significantly at high speed
- traffic-flow: A distribution of flows. This may affect efficient use of load-balancing techniques and resource consumption. More details discussed in later section of this document.
- layer3-gateway-count: uint32, number of layer 2 service instances that also provide layer 3 gateway service
- arpnp-table-size: uint32. This is only relevant with presence of layer 3 gateway

Integrated routing and bridging (IRB) and EVPN Type 5 route have blurred boundaries between layer 2 and layer 3 services.

3.2. Layer 3 Service Model

This section outlines traffic type, layer 3 protocol families, layer 3 prefixes distribution, layer 3 traffic flow and packet size distributions.

- proto-family: protocol family are defined with three sub-attributes. The list may grow as the complexity
  - proto - list: inet, inet6, iso
  - type - list: unicast, mcast, segment, labeled
  - vpn - list, true, false
- prefix-count, uint32, total unique prefixes
- prefix-distrib, list of prefix length size and percentage. This could be a distribution pattern, such as uniform, random. Or simply top representation of prefix lengths
- bgp-path-count, uint32, total BGP paths
- bgp-path-distrib, top representation of number of paths per prefix
- traffic-frame-size, similar to traffic-frame-size in layer 2 model. The focus is on the MTU size on each protocol interfaces and the impact of fragmentation
- traffic-flow, similar to traffic-flow in layer 2 model, it focuses on a set of labels, source and destination addresses as well as ports
- traffic-load, similar to traffic-load in layer 2 model
- ifl-count, uint32,
- vpn-count, uint32,

3.3. Infrastructure Service Model

- bgp-peer-ext-count, uint32, number of eBGP peers
- bgp-peer-int-count, uint32, number of iBGP peers
- bgp-path-mtu, list, true or false. Larger path mtu helps convergence

- o bgp-hold-time-distrib, list of top hold-time values and their respective percentage out of all peers.
- o bgp-as-path-distrib, list of top as-path lengths and their respective percentage of all BGP paths
- o bgp-community-distrib, list of top community size and their respective percentage out of all BGP paths
- o mpls-sig, list, MPLS signaling protocol, rsvp or ldp
- o rsvp-lsp-count-ingress, uint32, total ingress lsp count
- o rsvp-lsp-count-transit, uint32, total transit lsp count
- o rsvp-lsp-count-egress, uint32, total egress lsp count
- o ldp-fec-count, uint32, total forwarding equivalence class
- o rsvp-lsp-protection, list, link-node, link, frr
- o ospf-interface-type, list, point-to-point, broadcast, non-broadcast multi-access
- o ospf-lsa-distrib, list. OSPF Link Statement Advertisement distribution is comprised of those for core router in backbone area, and internal router in non-Backbone areas. A common modeling can include number of LSAs per OSPF LSA type
- o ospf-route-count, list, total OSPF routes in both backbone and non-backbone areas
- o isis-lsp-distrib, list, similar to ospf-lsa-distrib
- o isis-route-count, list, total IS-IS routes in both level-1 and level-2 areas

TODO: bridging, OAM, EOAM, BFD and etc

3.4. Node Level Features

TODO: node level feature set

3.5. Common Service Specification

For most network services, regardless of layer 2 or layer 3, protocol families, the following needs to be considered when measuring network capacity and baseline.

- o rib-learning-time, uint32 in seconds. This indicates show quickly the route processor learns routing objects either locally and remotely
- o fib-learning-time. In large routing system, forwarding engine residing on separate hardware from controller, takes additional time to install all forwarding entries learned by controller.
- o convergence-time, this is could be as a result of many events, such as uplink failure, ae member link failure, fast reroute, local repair, upstream node failure and etc
- o multihome-failover-time, this refers to traffic convergence in a topology where a customer edge (CE) device is connected to two or more provider edge (PE) devices.

- o issu-dark-window-size. Unlike NSR, the goal of ISSU is not zero packet loss. Instead, there will be a few seconds, or in some cases, sub-second dark window where it sees both total packet loss for both transit and or host bound protocol traffic.
- o cpu-util, total CPU utilization of the controllers in stead state
- o cpu-util-peak, Peak CPU utilization on the controller in event of failure, and convergence
- o mem-util, total memory utilization of the controllers in steady state
- o mem-util-peak, total memory utilization on the controller in event of failure and convergence
- o processes, list of top processes running on the controllers with their CPU and memory utilization.
- o lc-cpu-util, top CPU utilization on the line cards
- o lc-cpu-util-peak, maximum peak CPU utilization among all line cards in event of failure and convergence
- o lc-mem-util, top memory utilization on the line cards
- o lc-mem-util-peak, maximum peak memory utilization among all line cards in event of failure and convergence
- o throughout, in both pps and bps. This is measured with zero packet loss. For virtualized environment, throughput is sometimes measured with a small loss tolerance given the nature of shared resource
- o traffic performance, in both pps and bps. It is measured the rate of traffic received by pumping oversubscribed traffic at ingress
- o latency in us. this is more important within a local data center environment rather than DCI over wide area network. Use of extensive firewall filter or access control lists may affect latency
- o Out of Order Packet - This can happen in intra-node or over ECMP where different paths have large latency/delay variations.

The list of metrics can be used for network monitoring during network resiliency test. This is to understand how quickly a network service can restore during various events and failures

3.6. Common Network Events

A list of events is defined to characterize network resiliency. These attributes require that the provider networks have diverse paths and node redundancy built-in. They directly affect service level agreement and network availability.

3.6.1. Event Attributes

Each network or system event may each be defined with the following aspects

- o event-iteration, uint16, event to be repeated
- o event-interval, unit16, seconds in between consecutive events
- o event-dist, list, random, equal, or other type of event scheduling
- o event-timeout, uint16, seconds when a single event is expected to complete
- o event-convergence, uint16, seconds before the network can be recovered

3.6.2. Hardware Related

Some hardware failures can not easily replicated, or even simulated in a lab environment, like the memory errors. A system or network should be equipped to monitor, detect and contain the impact to avoid global catastrohpic failure that may proagate beyond a single node or the regional network.

- o hw-mod-yank: hardware module removal and insertion.
- o hw-interface: Transceiver on/off or any other simulated link failures.
- o hw-storage: storage failure, ether local or network attached.
- o hw-power: unplanned power failure.
- o hw-controller: Controller failure
- o hw-memory: memory errors

3.6.3. Software Component

- o sw-daemon-watchdog-loss: Induced CPU hog that trigger watchdog failure
- o sw-daemon-restart-graceful: Graceful software daemon restart.
- o sw-daemon-restart-kill: The process is killed and the daemon was forced to restart
- o sw-daemon-panic: Sometimes a panic can introduced to trigger a coredump of software daemon along with restart.
- o sw-os-panic: network operating system may panic under various situations. Many network products with a console access support OS panic with a special sequence keystroke. Sometimes it may also generate a coredump for further debug
- o sw-upgrade: In many provider networks, the most downtime actually come from scheduled maintenance, especially software or firmware upgrade to provide better feature set or as a result of security patch. It is important to understand the downtime requirement for a routine software upgrade on a given network device or the devices in the network. This often presents a challenge to the access network.

3.6.4. Protocol Events

- protocol-keepalive-loss: Loss of keepalive, such as hellos for routing-protocols like OSPF and BGP
- oam-keepalive-loss: there are many OAM protocols, such as EOAM, MPLS OAM, LACP, one of their main purposes is to detect reachability. This is different from routing protocols keepalive
- protocol-adjacency-reset: clear all protocol neighbors and any routes or link states learned from the neighbor
- protocol-db-purge: remove all database objects learned from a particular neighbor, or a group of neighbors, or all neigbors. The database maybe original set of state learned from neighbors, or the consolidated database.

3.6.5. Redundancy Failover

The network protocols and design have a lot of redundancy built-in. It is important to benchmark their effectiveness.

- ha-lag-links: measure packet loss in milliseconds when member link(s) of a link aggregation group is torn down. In case of protected interface, traffic should failover seamlessly to the backup interface in event of primary link failure
- ha-controller: In a system with redundant controller, measure the network recovery time when the primary controller fails. If the advanced non-stop routing/forwarding is enabled, the network should only experience zero or sub-second traffic loss.
- ha-multihome: in addition to device level redundancy, many protocols support network layer redundancy though multihoming such as EVPN.
- ha-mpls-frr: MPLS RSVP Fast ReRoute provides core network redundancy
- ha-uplink: the core network is typically designed to provide path diversity for edge devices, at either layer 2 and layer 3 connectivity. The resiliency of network is measured by how fast the system detects the failure and reroute the traffic

4. Use of Network Service Layer Abstract Model

The primary goal is to characterize and document a complex network using a simplified service model. While eliminating many details such as address assignment, actual route or mac entries, it retains a set of key network information, including services, scale, and performance profiles. This can be used to validate how well each underlying solution performs when delivering same set of services.

The model can also be used to build a virtualized topology with both static and dynamic scale closely resemble to a real network. This

eases network design and benchmarking, and helps capacity planning by studying the impact with changes to a specific dimension.

5. Acknowledgements

The authors appreciate and acknowledge comments from Al Morton and others based on initial discussions.

6. IANA Considerations

This memo includes no request to IANA.

All drafts are required to have an IANA considerations section (see Guidelines for Writing an IANA Considerations Section in RFCs [RFC5226] for a guide). If the draft does not require IANA to do anything, the section contains an explicit statement that this is the case (as above). If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

7. Security Considerations

All drafts are required to have a security considerations section. See RFC 3552 [RFC3552] for a guide.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

8.2. Informative References

[L2SM] B. Wu et al, "A Yang Data Model for L2VPN Service Delivery", 2017, <https://www.ietf.org/id/draft-ietf-l2sm-l2vpn-service-model-02.txt>.

[RFC8049] Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8049, DOI 10.17487/RFC8049, February 2017, <https://www.rfc-editor.org/info/rfc8049>.

Author’s Address

Sean Wu
Juniper Networks
2251 Corporate Park Dr.
Suite #200
Herndon, VA 20171
US

Phone: +1 571 203 1898
Email: xwu@juniper.net