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.

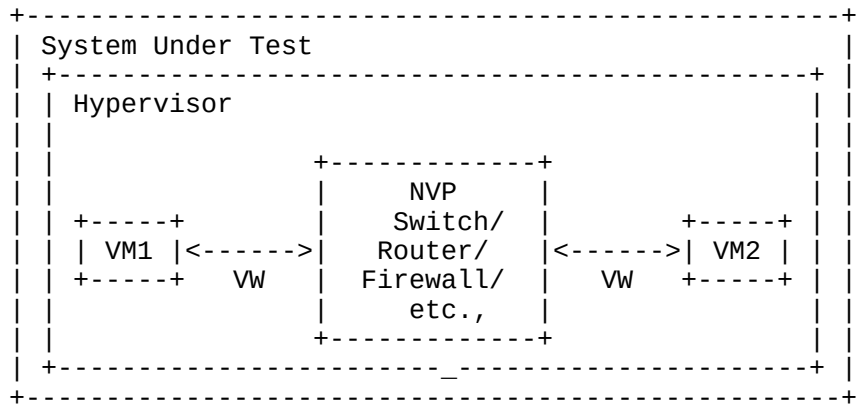
- Hashing method used
- Over-subscription rate
- Throughput available
- Latency characteristics

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.

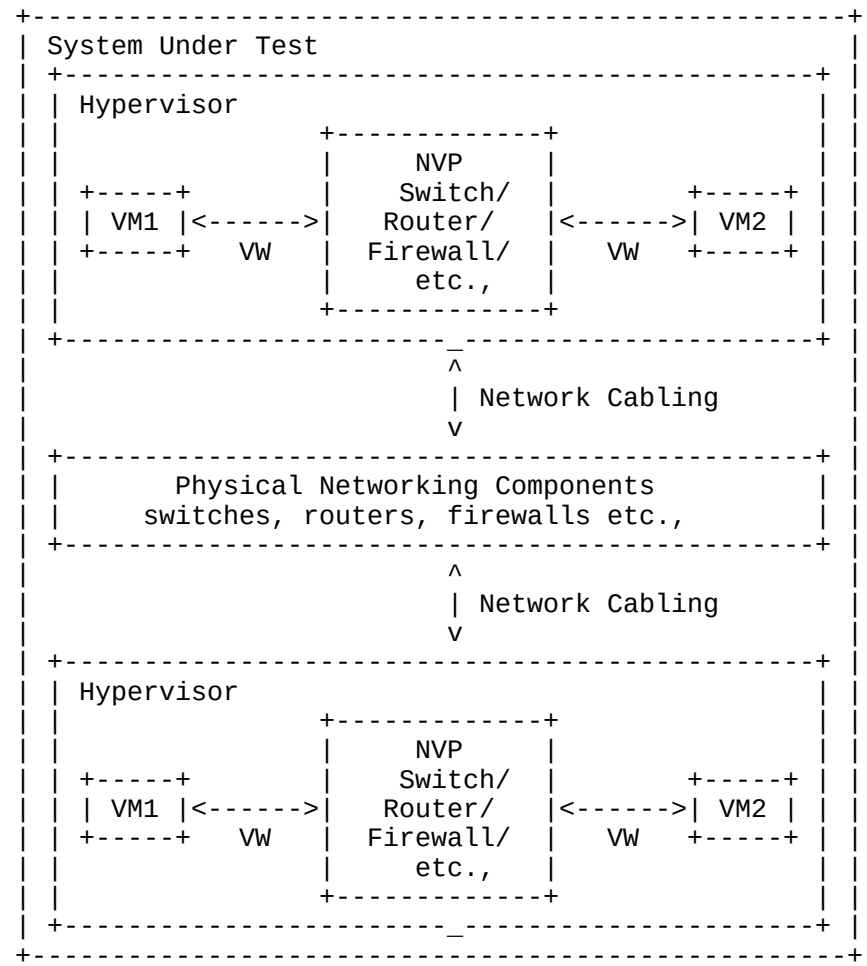


Legend
VM: Virtual Machine
VW: Virtual Wire

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

Internet Engineering Task Force
INTERNET-DRAFT, Intended Status: Informational
Expires December 23,2017
June 21, 2017

L. Avramov
Google
J. Rapp
VMware

Data Center Benchmarking Methodology
draft-ietf-bmwg-dcbench-methodology-18

Abstract

The purpose of this informational document is to establish test and evaluation methodology and measurement techniques for physical network equipment in the data center. A pre-requisite to this publication is the terminology document [draft-ietf-bmwg-dcbench-terminology]. Many of these terms and methods may be applicable beyond this publication's scope as the technologies originally applied in the data center are deployed elsewhere.

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."

Copyright Notice

Table of Contents

1. Introduction

Traffic patterns in the data center are not uniform and are constantly changing. They are dictated by the nature and variety of applications utilized in the data center. It can be largely east-west traffic flows (server to server inside the data center) in one data center and north-south (outside of the data center to server) in another, while others may combine both. Traffic patterns can be bursty in nature and contain many-to-one, many-to-many, or one-to-many flows. Each flow may also be small and latency sensitive or large and throughput sensitive while containing a mix of UDP and TCP traffic. All of these can coexist in a single cluster and flow through a single network device simultaneously. Benchmarking of

network devices have long used [RFC1242], [RFC2432], [RFC2544], [RFC2889] and [RFC3918] which have largely been focused around various latency attributes and Throughput [RFC2889] of the Device Under Test (DUT) being benchmarked. These standards are good at measuring theoretical Throughput, forwarding rates and latency under testing conditions; however, they do not represent real traffic patterns that may affect these networking devices.

Currently, typical data center networking devices are characterized by:

-High port density (48 ports of more)

-High speed (up to 100 GB/s currently per port)

-High throughput (line rate on all ports for Layer 2 and/or Layer 3)

-Low latency (in the microsecond or nanosecond range)

-Low amount of buffer (in the MB range per networking device)

-Layer 2 and Layer 3 forwarding capability (Layer 3 not mandatory)

This document provides a methodology for benchmarking Data Center physical network equipment DUT including congestion scenarios, switch buffer analysis, microburst, head of line blocking, while also using a wide mix of traffic conditions. The terminology document [draft-ietf-bmwg-dcbench-terminology] is a pre-requisite.

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. Methodology format and repeatability recommendation

The format used for each section of this document is the following:

-Objective

-Methodology

-Reporting Format

For each test methodology described, it is critical to obtain repeatability in the results. The recommendation is to perform enough iterations of the given test and to make sure the result is consistent. This is especially important for section 3, as the buffering testing has been historically the least reliable. The number of iterations SHOULD be explicitly reported. The relative standard deviation SHOULD be below 10%.

2. Line Rate Testing

2.1 Objective

Provide a maximum rate test for the performance values for Throughput, latency and jitter. It is meant to provide the tests to perform, and methodology to verify that a DUT is capable of forwarding packets at line rate under non-congested conditions.

2.2 Methodology

A traffic generator SHOULD be connected to all ports on the DUT. Two tests MUST be conducted: a port-pair test [RFC 2544/3918 section 15 compliant] and also in a full mesh type of DUT test [2889/3918 section 16 compliant].

For all tests, the test traffic generator sending rate MUST be less than or equal to 99.98% of the nominal value of Line Rate (with no further PPM adjustment to account for interface clock tolerances), to ensure stressing the DUT in reasonable worst case conditions (see RFC [draft-ietf-bmwg-dcbench-terminology] section 5 for more details -- note to RFC Editor, please replace all [draft-ietf-bmwg-dcbench-

terminology] references in this document with the future RFC number of that draft). Tests results at a lower rate MAY be provided for better understanding of performance increase in terms of latency and jitter when the rate is lower than 99.98%. The receiving rate of the traffic SHOULD be captured during this test in % of line rate.

The test MUST provide the statistics of minimum, average and maximum of the latency distribution, for the exact same iteration of the test.

The test MUST provide the statistics of minimum, average and maximum of the jitter distribution, for the exact same iteration of the test.

Alternatively when a traffic generator can not be connected to all ports on the DUT, a snake test MUST be used for line rate testing, excluding latency and jitter as those became then irrelevant. The snake test consists in the following method:

-connect the first and last port of the DUT to a traffic generator

-connect back to back sequentially all the ports in between: port 2 to 3, port 4 to 5 etc to port n-2 to port n-1; where n is the total number of ports of the DUT

-configure port 1 and 2 in the same vlan X, port 3 and 4 in the same vlan Y, etc. port n-1 and port n in the same vlan Z.

This snake test provides a capability to test line rate for Layer 2 and Layer 3 RFC 2544/3918 in instance where a traffic generator with only two ports is available. The latency and jitter are not to be considered with this test.

2.3 Reporting Format

The report MUST include:

-physical layer calibration information as defined into [draft-ietf-bmwg-dcbench-terminology] section 4.

-number of ports used

-reading for "Throughput received in percentage of bandwidth", while sending 99.98% of nominal value of Line Rate on each port, for each packet size from 64 bytes to 9216 bytes. As guidance, an increment of 64 byte packet size between each iteration being ideal, a 256 byte and 512 bytes being are also often used. The most common packets

sizes order for the report is:
64b,128b,256b,512b,1024b,1518b,4096,8000,9216b.

The pattern for testing can be expressed using [RFC 6985].

-Throughput needs to be expressed in % of total transmitted frames

-For packet drops, they MUST be expressed as a count of packets and SHOULD be expressed in % of line rate

-For latency and jitter, values expressed in unit of time [usually microsecond or nanosecond] reading across packet size from 64 bytes to 9216 bytes

-For latency and jitter, provide minimum, average and maximum values. If different iterations are done to gather the minimum, average and maximum, it SHOULD be specified in the report along with a justification on why the information could not have been gathered at the same test iteration

-For jitter, a histogram describing the population of packets measured per latency or latency buckets is RECOMMENDED

-The tests for Throughput, latency and jitter MAY be conducted as individual independent trials, with proper documentation in the report but SHOULD be conducted at the same time.

-The methodology makes an assumption that the DUT has at least nine ports, as certain methodologies require that number of ports or more.

3. Buffering Testing

3.1 Objective

To measure the size of the buffer of a DUT under typical|many|multiple conditions. Buffer architectures between multiple DUTs can differ and include egress buffering, shared egress buffering SoC (Switch-on-Chip), ingress buffering or a combination. The test methodology covers the buffer measurement regardless of buffer architecture used in the DUT.

3.2 Methodology

A traffic generator MUST be connected to all ports on the DUT.

The methodology for measuring buffering for a data-center switch is based on using known congestion of known fixed packet size along with maximum latency value measurements. The maximum latency will increase until the first packet drop occurs. At this point, the maximum latency value will remain constant. This is the point of inflection of this maximum latency change to a constant value. There MUST be multiple ingress ports receiving known amount of frames at a known fixed size, destined for the same egress port in order to create a known congestion condition. The total amount of packets sent from the oversubscribed port minus one, multiplied by the packet size represents the maximum port buffer size at the measured inflection point.

1) Measure the highest buffer efficiency

The tests described in this section have iterations called "first iteration", "second iteration" and, "last iteration". The idea is to show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

First iteration: ingress port 1 sending line rate to egress port 2, while port 3 sending a known low amount of over-subscription traffic (1% recommended) with a packet size of 64 bytes to egress port 2. Measure the buffer size value of the number of frames sent from the port sending the oversubscribed traffic up to the inflection point multiplied by the frame size.

Second iteration: ingress port 1 sending line rate to egress port 2, while port 3 sending a known low amount of over-subscription traffic (1% recommended) with same packet size 65 bytes to egress port 2. Measure the buffer size value of the number of frames sent from the port sending the oversubscribed traffic up to the inflection point multiplied by the frame size.

Last iteration: ingress port 1 sending line rate to egress port 2, while port 3 sending a known low amount of over-subscription traffic (1% recommended) with same packet size B bytes to egress port 2. Measure the buffer size value of the number of frames sent from the port sending the oversubscribed traffic up to the inflection point multiplied by the frame size.

When the B value is found to provide the largest buffer size, then size B allows the highest buffer efficiency.

2) Measure maximum port buffer size

The tests described in this section have iterations called "first

iteration", "second iteration" and, "last iteration". The idea is to show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

At fixed packet size B determined in procedure 1), for a fixed default Differentiated Services Code Point (DSCP)/Class of Service (COS) value of 0 and for unicast traffic proceed with the following:

First iteration: ingress port 1 sending line rate to egress port 2, while port 3 sending a known low amount of over-subscription traffic (1% recommended) with same packet size to the egress port 2. Measure the buffer size value by multiplying the number of extra frames sent by the frame size.

Second iteration: ingress port 2 sending line rate to egress port 3, while port 4 sending a known low amount of over-subscription traffic (1% recommended) with same packet size to the egress port 3. Measure the buffer size value by multiplying the number of extra frames sent by the frame size.

Last iteration: ingress port N-2 sending line rate traffic to egress port N-1, while port N sending a known low amount of over-subscription traffic (1% recommended) with same packet size to the egress port N. Measure the buffer size value by multiplying the number of extra frames sent by the frame size.

This test series MAY be repeated using all different DSCP/COS values of traffic and then using Multicast type of traffic, in order to find if there is any DSCP/COS impact on the buffer size.

3) Measure maximum port pair buffer sizes

The tests described in this section have iterations called "first iteration", "second iteration" and, "last iteration". The idea is to show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

First iteration: ingress port 1 sending line rate to egress port 2; ingress port 3 sending line rate to egress port 4 etc. Ingress port N-1 and N will respectively over subscribe at 1% of line rate egress port 2 and port 3. Measure the buffer size value by multiplying the number of extra frames sent by the frame size for each egress port.

Second iteration: ingress port 1 sending line rate to egress port 2; ingress port 3 sending line rate to egress port 4 etc. Ingress port N-1 and N will respectively over subscribe at 1% of line rate egress

port 4 and port 5. Measure the buffer size value by multiplying the number of extra frames sent by the frame size for each egress port.

Last iteration: ingress port 1 sending line rate to egress port 2; ingress port 3 sending line rate to egress port 4 etc. Ingress port N-1 and N will respectively over subscribe at 1% of line rate egress port N-3 and port N-2. Measure the buffer size value by multiplying the number of extra frames sent by the frame size for each egress port.

This test series MAY be repeated using all different DSCP/COS values of traffic and then using Multicast type of traffic.

4) Measure maximum DUT buffer size with many to one ports

The tests described in this section have iterations called "first iteration", "second iteration" and, "last iteration". The idea is to show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

First iteration: ingress ports 1,2,... N-1 sending each [(1/[N-1])*99.98]+[1/[N-1]] % of line rate per port to the N egress port.

Second iteration: ingress ports 2,... N sending each [(1/[N-1])*99.98]+[1/[N-1]] % of line rate per port to the 1 egress port.

Last iteration: ingress ports N,1,2...N-2 sending each [(1/[N-1])*99.98]+[1/[N-1]] % of line rate per port to the N-1 egress port.

This test series MAY be repeated using all different COS values of traffic and then using Multicast type of traffic.

Unicast traffic and then Multicast traffic SHOULD be used in order to determine the proportion of buffer for documented selection of tests. Also the COS value for the packets SHOULD be provided for each test iteration as the buffer allocation size MAY differ per COS value. It is RECOMMENDED that the ingress and egress ports are varied in a random, but documented fashion in multiple tests to measure the buffer size for each port of the DUT.

3.3 Reporting format

The report MUST include:

- The packet size used for the most efficient buffer used, along with DSCP/COS value

- The maximum port buffer size for each port

- The maximum DUT buffer size

- The packet size used in the test

- The amount of over-subscription if different than 1%

- The number of ingress and egress ports along with their location on the DUT

- The repeatability of the test needs to be indicated: number of iterations of the same test and percentage of variation between results for each of the tests (min, max, avg)

The percentage of variation is a metric providing a sense of how big the difference between the measured value and the previous ones.

For example, for a latency test where the minimum latency is measured, the percentage of variation of the minimum latency will indicate by how much this value has varied between the current test executed and the previous one.

PV=((x2-x1)/x1)*100 where x2 is the minimum latency value in the current test and x1 is the minimum latency value obtained in the previous test.

The same formula is used for max and avg variations measured.

4 Microburst Testing

4.1 Objective

To find the maximum amount of packet bursts a DUT can sustain under various configurations.

This test provides additional methodology to the other RFC tests:

-All bursts should be send with 100% intensity. Note: intensity is defined in [draft-ietf-bmwg-dcbench-terminology] section 6.1.1

-All ports of the DUT must be used for this test

-All ports are recommended to be testes simultaneously

4.2 Methodology

A traffic generator MUST be connected to all ports on the DUT. In order to cause congestion, two or more ingress ports MUST send bursts of packets destined for the same egress port. The simplest of the setups would be two ingress ports and one egress port (2-to-1).

The burst MUST be sent with an intensity of 100% (intensity is defined in [draft-ietf-bmwg-dcbench-terminology] section 6.1.1), meaning the burst of packets will be sent with a minimum inter-packet gap. The amount of packet contained in the burst will be trial variable and increase until there is a non-zero packet loss measured. The aggregate amount of packets from all the senders will be used to calculate the maximum amount of microburst the DUT can sustain.

It is RECOMMENDED that the ingress and egress ports are varied in multiple tests to measure the maximum microburst capacity.

The intensity of a microburst MAY be varied in order to obtain the microburst capacity at various ingress rates. Intensity of microburst is defined in [draft-ietf-bmwg-dcbench-terminology].

It is RECOMMENDED that all ports on the DUT will be tested simultaneously and in various configurations in order to understand all the combinations of ingress ports, egress ports and intensities.

An example would be:

First Iteration: N-1 Ingress ports sending to 1 Egress Ports

Second Iterations: N-2 Ingress ports sending to 2 Egress Ports

Last Iterations: 2 Ingress ports sending to N-2 Egress Ports

4.3 Reporting Format

The report MUST include:

- The maximum number of packets received per ingress port with the maximum burst size obtained with zero packet loss

- The packet size used in the test

- The number of ingress and egress ports along with their location on the DUT

- The repeatability of the test needs to be indicated: number of iterations of the same test and percentage of variation between results (min, max, avg)

5. Head of Line Blocking

5.1 Objective

Head-of-line blocking (HOLB) is a performance-limiting phenomenon that occurs when packets are held-up by the first packet ahead waiting to be transmitted to a different output port. This is defined in RFC 2889 section 5.5, Congestion Control. This section expands on RFC 2889 in the context of Data Center Benchmarking.

The objective of this test is to understand the DUT behavior under head of line blocking scenario and measure the packet loss.

Here are the differences between this HOLB test and RFC 2889:

-This HOLB starts with 8 ports in two groups of 4, instead of 4 RFC 2889

-This HOLB shifts all the port numbers by one in a second iteration of the test, this is new compared to RFC 2889. The shifting port numbers continue until all ports are the first in the group. The purpose is to make sure to have tested all permutations to cover differences of behavior in the SoC of the DUT

-Another test in this HOLB expands the group of ports, such that traffic is divided among 4 ports instead of two (25% instead of 50% per port)

-Section 5.3 adds additional reporting requirements from Congestion Control in RFC 2889

5.2 Methodology

In order to cause congestion in the form of head of line blocking, groups of four ports are used. A group has 2 ingress and 2 egress ports. The first ingress port MUST have two flows configured each going to a different egress port. The second ingress port will congest the second egress port by sending line rate. The goal is to measure if there is loss on the flow for the first egress port which is not over-subscribed.

A traffic generator MUST be connected to at least eight ports on the DUT and SHOULD be connected using all the DUT ports.

1) Measure two groups with eight DUT ports

The tests described in this section have iterations called "first iteration", "second iteration" and, "last iteration". The idea is to show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

First iteration: measure the packet loss for two groups with consecutive ports

The first group is composed of: ingress port 1 is sending 50% of traffic to egress port 3 and ingress port 1 is sending 50% of traffic to egress port 4. Ingress port 2 is sending line rate to egress port 4. Measure the amount of traffic loss for the traffic from ingress port 1 to egress port 3.

The second group is composed of: ingress port 5 is sending 50% of traffic to egress port 7 and ingress port 5 is sending 50% of traffic to egress port 8. Ingress port 6 is sending line rate to egress port 8. Measure the amount of traffic loss for the traffic from ingress port 5 to egress port 7.

Second iteration: repeat the first iteration by shifting all the ports from N to N+1.

The first group is composed of: ingress port 2 is sending 50% of traffic to egress port 4 and ingress port 2 is sending 50% of traffic to egress port 5. Ingress port 3 is sending line rate to egress port 5. Measure the amount of traffic loss for the traffic from ingress port 2 to egress port 4.

The second group is composed of: ingress port 6 is sending 50% of traffic to egress port 8 and ingress port 6 is sending 50% of traffic to egress port 9. Ingress port 7 is sending line rate to egress port 9. Measure the amount of traffic loss for the traffic from ingress port 6 to egress port 8.

Last iteration: when the first port of the first group is connected on the last DUT port and the last port of the second group is connected to the seventh port of the DUT.

Measure the amount of traffic loss for the traffic from ingress port N to egress port 2 and from ingress port 4 to egress port 6.

2) Measure with N/4 groups with N DUT ports

The tests described in this section have iterations called "first iteration", "second iteration" and, "last iteration". The idea is to

show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

The traffic from ingress split across 4 egress ports (100/4=25%).

First iteration: Expand to fully utilize all the DUT ports in increments of four. Repeat the methodology of 1) with all the group of ports possible to achieve on the device and measure for each port group the amount of traffic loss.

Second iteration: Shift by +1 the start of each consecutive ports of groups

Last iteration: Shift by N-1 the start of each consecutive ports of groups and measure the traffic loss for each port group.

5.3 Reporting Format

For each test the report MUST include:

- The port configuration including the number and location of ingress and egress ports located on the DUT

- If HOLB was observed in accordance with the HOLB test in section 5

- Percent of traffic loss

- The repeatability of the test needs to be indicated: number of iteration of the same test and percentage of variation between results (min, max, avg)

6. Incast Stateful and Stateless Traffic

6.1 Objective

The objective of this test is to measure the values for TCP Goodput [1] and latency with a mix of large and small flows. The test is designed to simulate a mixed environment of stateful flows that require high rates of goodput and stateless flows that require low latency. Stateful flows are created by generating TCP traffic and, stateless flows are created using UDP type of traffic.

6.2 Methodology

In order to simulate the effects of stateless and stateful traffic on

the DUT, there MUST be multiple ingress ports receiving traffic destined for the same egress port. There also MAY be a mix of stateful and stateless traffic arriving on a single ingress port. The simplest setup would be 2 ingress ports receiving traffic destined to the same egress port.

One ingress port MUST be maintaining a TCP connection trough the ingress port to a receiver connected to an egress port. Traffic in the TCP stream MUST be sent at the maximum rate allowed by the traffic generator. At the same time, the TCP traffic is flowing through the DUT the stateless traffic is sent destined to a receiver on the same egress port. The stateless traffic MUST be a microburst of 100% intensity.

It is RECOMMENDED that the ingress and egress ports are varied in multiple tests to measure the maximum microburst capacity.

The intensity of a microburst MAY be varied in order to obtain the microburst capacity at various ingress rates.

It is RECOMMENDED that all ports on the DUT be used in the test.

The tests described bellow have iterations called "first iteration", "second iteration" and, "last iteration". The idea is to show the first two iterations so the reader understands the logic on how to keep incrementing the iterations. The last iteration shows the end state of the variables.

For example:

Stateful Traffic port variation (TCP traffic):

TCP traffic needs to be generated in this section. During Iterations number of Egress ports MAY vary as well.

First Iteration: 1 Ingress port receiving stateful TCP traffic and 1 Ingress port receiving stateless traffic destined to 1 Egress Port

Second Iteration: 2 Ingress port receiving stateful TCP traffic and 1 Ingress port receiving stateless traffic destined to 1 Egress Port

Last Iteration: N-2 Ingress port receiving stateful TCP traffic and 1 Ingress port receiving stateless traffic destined to 1 Egress Port

Stateless Traffic port variation (UDP traffic):

UDP traffic needs to be generated for this test. During Iterations, the number of Egress ports MAY vary as well.

First Iteration: 1 Ingress port receiving stateful TCP traffic and 1 Ingress port receiving stateless traffic destined to 1 Egress Port

Second Iteration: 1 Ingress port receiving stateful TCP traffic and 2 Ingress port receiving stateless traffic destined to 1 Egress Port

Last Iteration: 1 Ingress port receiving stateful TCP traffic and N-2 Ingress port receiving stateless traffic destined to 1 Egress Port

6.3 Reporting Format

The report MUST include the following:

- Number of ingress and egress ports along with designation of stateful or stateless flow assignment.

- Stateful flow goodput

- Stateless flow latency

- The repeatability of the test needs to be indicated: number of iterations of the same test and percentage of variation between results (min, max, avg)

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 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.

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

8. IANA Considerations

NO IANA Action is requested at this time.

9. References

9.1. Normative References

[RFC1242] Bradner, S. "Benchmarking Terminology for Network Interconnection Devices", BCP 14, RFC 1242, DOI 10.17487/RFC1242, July 1991, <http://www.rfc-editor.org/info/rfc1242>

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", BCP 14, RFC 2544, DOI 10.17487/RFC2544, March 1999, <http://www.rfc-editor.org/info/rfc2544>

9.2. Informative References

[draft-ietf-bmwg-dcbench-terminology] Avramov L. and Rapp J., "Data Center Benchmarking Terminology", April 2017, RFC "draft-ietf-bmwg-dcbench-terminology", Date [to be fixed when the RFC is published and 1 to be replaced by the RFC number

[RFC2889] Mandeville R. and Perser J., "Benchmarking Methodology for LAN Switching Devices", RFC 2889, August 2000, <http://www.rfc-editor.org/info/rfc2889>

[RFC3918] Stopp D. and Hickman B., "Methodology for IP Multicast Benchmarking", RFC 3918, October 2004, <http://www.rfc-editor.org/info/rfc3918>

[RFC 6985] A. Morton, "IMIX Genome: Specification of Variable Packet Sizes for Additional Testing", RFC 6985, July 2013, <http://www.rfc-editor.org/info/rfc6985>

[1] Yanpei Chen, Rean Griffith, Junda Liu, Randy H. Katz, Anthony D. Joseph, "Understanding TCP Incast Throughput Collapse in Datacenter Networks, "http://yanpeichen.com/professional/usenixLoginIncastReady.pdf"

[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>

[RFC2432] Dubray, K., "Terminology for IP Multicast Benchmarking", BCP 14, RFC 2432, DOI 10.17487/RFC2432, October 1998, <http://www.rfc-editor.org/info/rfc2432>

9.2. Acknowledgements

The authors would like to thank Alfred Morton and Scott Bradner for their reviews and feedback.

Authors' Addresses

Lucien Avramov
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
United States
Phone: +1 408 774 9077
Email: lucien.avramov@gmail.com

Jacob Rapp
VMware
3401 Hillview Ave
Palo Alto, CA
United States
Phone: +1 650 857 3367
Email: jrapp@vmware.com

Internet Engineering Task Force
INTERNET-DRAFT, Intended status: Informational
Expires: December 24,2017
June 22, 2017

L. Avramov
Google
J. Rapp
VMware

Data Center Benchmarking Terminology
draft-ietf-bmwg-dcbench-terminology-19

Abstract

The purpose of this informational document is to establish definitions and describe measurement techniques for data center benchmarking, as well as it is to introduce new terminologies applicable to performance evaluations of data center network equipment. This document establishes the important concepts for benchmarking network switches and routers in the data center and, is a pre-requisite to the test methodology publication [draft-ietf-bmwg-dcbench-methodology]. Many of these terms and methods may be applicable to network equipment beyond this publication's scope as the technologies originally applied in the data center are deployed elsewhere.

Table of Contents

1. Introduction

Traffic patterns in the data center are not uniform and are constantly changing. They are dictated by the nature and variety of applications utilized in the data center. It can be largely east-west traffic flows (server to server inside the data center) in one data center and north-south (outside of the data center to server) in another, while some may combine both. Traffic patterns can be bursty in nature and contain many-to-one, many-to-many, or one-to-many flows. Each flow may also be small and latency sensitive or large and throughput sensitive while containing a mix of UDP and TCP traffic. One or more of these may coexist in a single cluster and flow through a single network device simultaneously. Benchmarking of network devices have long used [RFC1242], [RFC2432], [RFC2544], [RFC2889] and [RFC3918]. These benchmarks have largely been focused around various latency attributes and max throughput of the Device Under Test being benchmarked. These standards are good at measuring theoretical max throughput, forwarding rates and latency under testing conditions, but they do not represent real traffic patterns that may affect these networking devices. The data center networking devices covered are switches and routers.

Currently, typical data center networking devices are characterized by:

-High port density (48 ports of more)

-High speed (up to 100 GB/s currently per port)

-High throughput (line rate on all ports for Layer 2 and/or Layer 3)

-Low latency (in the microsecond or nanosecond range)

-Low amount of buffer (in the MB range per networking device)

-Layer 2 and Layer 3 forwarding capability (Layer 3 not mandatory)

The following document defines a set of definitions, metrics and terminologies including congestion scenarios, switch buffer analysis and redefines basic definitions in order to represent a wide mix of traffic conditions. The test methodologies are defined in [draft-ietf-bmwg-dcbench-methodology].

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. Definition format

Term to be defined. (e.g., Latency)

Definition: The specific definition for the term.

Discussion: A brief discussion about the term, its application and any restrictions on measurement procedures.

Measurement Units: Methodology for the measure and units used to report measurements of this term, if applicable.

2. Latency

2.1. Definition

Latency is a the amount of time it takes a frame to transit the Device Under Test (DUT). Latency is measured in units of time (seconds, milliseconds, microseconds and so on). The purpose of measuring latency is to understand the impact of adding a device in the communication path.

The Latency interval can be assessed between different combinations of events, regardless of the type of switching device (bit forwarding aka cut-through, or store-and-forward type of device). [RFC1242] defined Latency differently for each of these types of devices.

Traditionally the latency measurement definitions are:

FILO (First In Last Out)

The time interval starting when the end of the first bit of the input frame reaches the input port and ending when the last bit of the output frame is seen on the output port.

FIFO (First In First Out):

The time interval starting when the end of the first bit of the input frame reaches the input port and ending when the start of the first

bit of the output frame is seen on the output port. [RFC1242] Latency for bit forwarding devices uses these events.

LILO (Last In Last Out):

The time interval starting when the last bit of the input frame reaches the input port and the last bit of the output frame is seen on the output port.

LIFO (Last In First Out):

The time interval starting when the last bit of the input frame reaches the input port and ending when the first bit of the output frame is seen on the output port. [RFC1242] Latency for bit forwarding devices uses these events.

Another possibility to summarize the four different definitions above is to refer to the bit position as they normally occur: Input to output.

FILO is FL (First bit Last bit). FIFO is FF (First bit First bit). LILO is LL (Last bit Last bit). LIFO is LF (Last bit First bit).

This definition explained in this section in context of data center switching benchmarking is in lieu of the previous definition of Latency defined in RFC 1242, section 3.8 and is quoted here:

For store and forward devices: The time interval starting when the last bit of the input frame reaches the input port and ending when the first bit of the output frame is seen on the output port.

For bit forwarding devices: The time interval starting when the end of the first bit of the input frame reaches the input port and ending when the start of the first bit of the output frame is seen on the output port.

To accommodate both types of network devices and hybrids of the two types that have emerged, switch Latency measurements made according to this document MUST be measured with the FILO events. FILO will include the latency of the switch and the latency of the frame as well as the serialization delay. It is a picture of the 'whole' latency going through the DUT. For applications which are latency sensitive and can function with initial bytes of the frame, FIFO (or RFC 1242 Latency for bit forwarding devices) MAY be used. In all cases, the event combination used in Latency measurement MUST be reported.

2.2 Discussion

As mentioned in section 2.1, FILO is the most important measuring definition.

Not all DUTs are exclusively cut-through or store-and-forward. Data Center DUTs are frequently store-and-forward for smaller packet sizes and then adopting a cut-through behavior. The change of behavior happens at specific larger packet sizes. The value of the packet size for the behavior to change MAY be configurable depending on the DUT manufacturer. FILO covers all scenarios: Store-and-forward or cut-through. The threshold of behavior change does not matter for benchmarking since FILO covers both possible scenarios.

LIFO mechanism can be used with store forward type of switches but not with cut-through type of switches, as it will provide negative latency values for larger packet sizes because LIFO removes the serialization delay. Therefore, this mechanism MUST NOT be used when comparing latencies of two different DUTs.

2.3 Measurement Units

The measuring methods to use for benchmarking purposes are as follows:

1) FILO MUST be used as a measuring method, as this will include the latency of the packet; and today the application commonly needs to read the whole packet to process the information and take an action.

2) FIFO MAY be used for certain applications able to proceed the data as the first bits arrive, as for example for a Field-Programmable Gate Array (FPGA)

3) LIFO MUST NOT be used, because it subtracts the latency of the packet; unlike all the other methods.

3 Jitter

3.1 Definition

Jitter in the data center context is synonymous with the common term Delay variation. It is derived from multiple measurements of one-way delay, as described in RFC 3393. The mandatory definition of Delay Variation is the Packet Delay Variation (PDV) from section 4.2 of [RFC5481]. When considering a stream of packets, the delays of all packets are subtracted from the minimum delay over all packets in the stream. This facilitates assessment of the range of delay variation

(Max - Min), or a high percentile of PDV (99th percentile, for robustness against outliers).

When First-bit to Last-bit timestamps are used for Delay measurement, then Delay Variation MUST be measured using packets or frames of the same size, since the definition of latency includes the serialization time for each packet. Otherwise if using First-bit to First-bit, the size restriction does not apply.

3.2 Discussion

In addition to PDV Range and/or a high percentile of PDV, Inter-Packet Delay Variation (IPDV) as defined in section 4.1 of [RFC5481] (differences between two consecutive packets) MAY be used for the purpose of determining how packet spacing has changed during transfer, for example, to see if packet stream has become closely-spaced or "bursty". However, the Absolute Value of IPDV SHOULD NOT be used, as this collapses the "bursty" and "dispersed" sides of the IPDV distribution together.

3.3 Measurement Units

The measurement of delay variation is expressed in units of seconds. A PDV histogram MAY be provided for the population of packets measured.

4 Physical Layer Calibration

4.1 Definition

The calibration of the physical layer consists of defining and measuring the latency of the physical devices used to perform tests on the DUT.

It includes the list of all physical layer components used as listed here after:

-Type of device used to generate traffic / measure traffic

-Type of line cards used on the traffic generator

-Type of transceivers on traffic generator

-Type of transceivers on DUT

-Type of cables

-Length of cables

-Software name, and version of traffic generator and DUT

-List of enabled features on DUT MAY be provided and is recommended (especially the control plane protocols such as Link Layer Discovery Protocol, Spanning-Tree etc.). A comprehensive configuration file MAY be provided to this effect.

4.2 Discussion

Physical layer calibration is part of the end to end latency, which should be taken into acknowledgment while evaluating the DUT. Small variations of the physical components of the test may impact the latency being measured, therefore they MUST be described when presenting results.

4.3 Measurement Units

It is RECOMMENDED to use all cables of: The same type, the same length, when possible using the same vendor. It is a MUST to document the cables specifications on section 4.1 along with the test results. The test report MUST specify if the cable latency has been removed from the test measures or not. The accuracy of the traffic generator measure MUST be provided (this is usually a value in the 20ns range for current test equipment).

5 Line rate

5.1 Definition

The transmit timing, or maximum transmitted data rate is controlled by the "transmit clock" in the DUT. The receive timing (maximum ingress data rate) is derived from the transmit clock of the connected interface.

The line rate or physical layer frame rate is the maximum capacity to send frames of a specific size at the transmit clock frequency of the DUT.

The term "nominal value of Line Rate" defines the maximum speed capability for the given port; for example 1GE, 10GE, 40GE, 100GE etc.

The frequency ("clock rate") of the transmit clock in any two connected interfaces will never be precisely the same; therefore, a

tolerance is needed. This will be expressed by Parts Per Million (PPM) value. The IEEE standards allow a specific +/- variance in the transmit clock rate, and Ethernet is designed to allow for small, normal variations between the two clock rates. This results in a tolerance of the line rate value when traffic is generated from a testing equipment to a DUT.

Line rate SHOULD be measured in frames per second.

5.2 Discussion

For a transmit clock source, most Ethernet switches use "clock modules" (also called "oscillator modules") that are sealed, internally temperature-compensated, and very accurate. The output frequency of these modules is not adjustable because it is not necessary. Many test sets, however, offer a software-controlled adjustment of the transmit clock rate. These adjustments SHOULD be used to compensate the test equipment in order to not send more than the line rate of the DUT.

To allow for the minor variations typically found in the clock rate of commercially-available clock modules and other crystal-based oscillators, Ethernet standards specify the maximum transmit clock rate variation to be not more than +/- 100 PPM (parts per million) from a calculated center frequency. Therefore a DUT must be able to accept frames at a rate within +/- 100 PPM to comply with the standards.

Very few clock circuits are precisely +/- 0.0 PPM because:

1.The Ethernet standards allow a maximum of +/- 100 PPM (parts per million) variance over time. Therefore it is normal for the frequency of the oscillator circuits to experience variation over time and over a wide temperature range, among external factors.

2.The crystals, or clock modules, usually have a specific +/- PPM variance that is significantly better than +/- 100 PPM. Often times this is +/- 30 PPM or better in order to be considered a "certification instrument".

When testing an Ethernet switch throughput at "line rate", any specific switch will have a clock rate variance. If a test set is running +1 PPM faster than a switch under test, and a sustained line rate test is performed, a gradual increase in latency and eventually packet drops as buffers fill and overflow in the switch can be observed. Depending on how much clock variance there is between the two connected systems, the effect may be seen after the traffic

stream has been running for a few hundred microseconds, a few milliseconds, or seconds. The same low latency and no-packet-loss can be demonstrated by setting the test set link occupancy to slightly less than 100 percent link occupancy. Typically 99 percent link occupancy produces excellent low-latency and no packet loss. No Ethernet switch or router will have a transmit clock rate of exactly +/- 0.0 PPM. Very few (if any) test sets have a clock rate that is precisely +/- 0.0 PPM.

Test set equipment manufacturers are well-aware of the standards, and allow a software-controlled +/- 100 PPM "offset" (clock-rate adjustment) to compensate for normal variations in the clock speed of DUTs. This offset adjustment allows engineers to determine the approximate speed the connected device is operating, and verify that it is within parameters allowed by standards.

5.3 Measurement Units

"Line Rate" can be measured in terms of "Frame Rate":

Frame Rate = Transmit-Clock-Frequency / (Frame-Length*8 + Minimum_Gap + Preamble + Start-Frame Delimiter)

Minimum_Gap represents the inter frame gap. This formula "scales up" or "scales down" to represent 1 GB Ethernet, or 10 GB Ethernet and so on.

Example for 1 GB Ethernet speed with 64-byte frames: Frame Rate = 1,000,000,000 /(64*8 + 96 + 56 + 8) Frame Rate = 1,000,000,000 / 672 Frame Rate = 1,488,095.2 frames per second.

Considering the allowance of +/- 100 PPM, a switch may "legally" transmit traffic at a frame rate between 1,487,946.4 FPS and 1,488,244 FPS. Each 1 PPM variation in clock rate will translate to a 1.488 frame-per-second frame rate increase or decrease.

In a production network, it is very unlikely to see precise line rate over a very brief period. There is no observable difference between dropping packets at 99% of line rate and 100% of line rate.

Line rate can be measured at 100% of line rate with a -100PPM adjustment.

Line rate SHOULD be measured at 99,98% with 0 PPM adjustment.

The PPM adjustment SHOULD only be used for a line rate type of

measurement.

6 Buffering

6.1 Buffer

6.1.1 Definition

Buffer Size: The term buffer size represents the total amount of frame buffering memory available on a DUT. This size is expressed in B (byte); KB (kilobyte), MB (megabyte) or GB (gigabyte). When the buffer size is expressed it SHOULD be defined by a size metric stated above. When the buffer size is expressed, an indication of the frame MTU used for that measurement is also necessary as well as the cos (class of service) or dscp (differentiated services code point) value set; as often times the buffers are carved by quality of service implementation. Please refer to the buffer efficiency section for further details.

Example: Buffer Size of DUT when sending 1518 byte frames is 18 MB.

Port Buffer Size: The port buffer size is the amount of buffer for a single ingress port, egress port or combination of ingress and egress buffering location for a single port. The reason for mentioning the three locations for the port buffer is because the DUT buffering scheme can be unknown or untested, and so knowing the buffer location helps clarify the buffer architecture and consequently the total buffer size. The Port Buffer Size is an informational value that MAY be provided from the DUT vendor. It is not a value that is tested by benchmarking. Benchmarking will be done using the Maximum Port Buffer Size or Maximum Buffer Size methodology.

Maximum Port Buffer Size: In most cases, this is the same as the Port Buffer Size. In certain switch architecture called SoC (switch on chip), there is a port buffer and a shared buffer pool available for all ports. The Maximum Port Buffer Size , in terms of an SoC buffer, represents the sum of the port buffer and the maximum value of shared buffer allowed for this port, defined in terms of B (byte), KB (kilobyte), MB (megabyte), or GB (gigabyte). The Maximum Port Buffer Size needs to be expressed along with the frame MTU used for the measurement and the cos or dscp bit value set for the test.

Example: A DUT has been measured to have 3KB of port buffer for 1518 frame size packets and a total of 4.7 MB of maximum port buffer for 1518 frame size packets and a cos of 0.

Maximum DUT Buffer Size: This is the total size of Buffer a DUT can

be measured to have. It is, most likely, different than than the Maximum Port Buffer Size. It can also be different from the sum of Maximum Port Buffer Size. The Maximum Buffer Size needs to be expressed along with the frame MTU used for the measurement and along with the cos or dscp value set during the test.

Example: A DUT has been measured to have 3KB of port buffer for 1518 frame size packets and a total of 4.7 MB of maximum port buffer for 1518 B frame size packets. The DUT has a Maximum Buffer Size of 18 MB at 1500 B and a cos of 0.

Burst: The burst is a fixed number of packets sent over a percentage of linerate of a defined port speed. The amount of frames sent are evenly distributed across the interval, T. A constant, C, can be defined to provide the average time between two consecutive packets evenly spaced.

Microburst: It is a burst. A microburst is when packet drops occur when there is not sustained or noticeable congestion upon a link or device. A characterization of microburst is when the Burst is not evenly distributed over T, and is less than the constant C [C= average time between two consecutive packets evenly spaced out].

Intensity of Microburst: This is a percentage, representing the level of microburst between 1 and 100%. The higher the number the higher the microburst is. I=[1-[(TP2-Tp1)+(Tp3-Tp2)+....(TpN-Tp(n-1)] / Sum(packets)]]*100

The above definitions are not meant to comment on the ideal sizing of a buffer, rather on how to measure it. A larger buffer is not necessarily better and can cause issues with buffer bloat.

6.1.2 Discussion

When measuring buffering on a DUT, it is important to understand the behavior for each and all ports. This provides data for the total amount of buffering available on the switch. The terms of buffer efficiency here helps one understand the optimum packet size for the buffer, or the real volume of the buffer available for a specific packet size. This section does not discuss how to conduct the test methodology; instead, it explains the buffer definitions and what metrics should be provided for a comprehensive data center device buffering benchmarking.

6.1.3 Measurement Units

When Buffer is measured:

-The buffer size MUST be measured

-The port buffer size MAY be provided for each port

-The maximum port buffer size MUST be measured

-The maximum DUT buffer size MUST be measured

-The intensity of microburst MAY be mentioned when a microburst test is performed

-The cos or dscp value set during the test SHOULD be provided

6.2 Incast

6.2.1 Definition

The term Incast, very commonly utilized in the data center, refers to the traffic pattern of many-to-one or many-to-many traffic patterns. It measures the number of ingress and egress ports and the level of synchronization attributed, as defined in this section. Typically in the data center it would refer to many different ingress server ports (many), sending traffic to a common uplink (many-to-one), or multiple uplinks (many-to-many). This pattern is generalized for any network as many incoming ports sending traffic to one or few uplinks.

Synchronous arrival time: When two, or more, frames of respective sizes L1 and L2 arrive at their respective one or multiple ingress ports, and there is an overlap of the arrival time for any of the bits on the Device Under Test (DUT), then the frames L1 and L2 have a synchronous arrival times. This is called Incast regardless of in many-to-one (simpler form) or, many-to-many.

Asynchronous arrival time: Any condition not defined by synchronous arrival time.

Percentage of synchronization: This defines the level of overlap [amount of bits] between the frames L1,L2..Ln.

Example: Two 64 bytes frames, of length L1 and L2, arrive to ingress port 1 and port 2 of the DUT. There is an overlap of 6.4 bytes between the two where L1 and L2 were at the same time on the respective ingress ports. Therefore the percentage of synchronization is 10%.

Stateful type traffic defines packets exchanged with a stateful protocol such as TCP.

Stateless type traffic defines packets exchanged with a stateless protocol such as UDP.

6.2.2 Discussion

In this scenario, buffers are solicited on the DUT. In an ingress buffering mechanism, the ingress port buffers would be solicited along with Virtual Output Queues, when available; whereas in an egress buffer mechanism, the egress buffer of the one outgoing port would be used.

In either case, regardless of where the buffer memory is located on the switch architecture, the Incast creates buffer utilization.

When one or more frames having synchronous arrival times at the DUT they are considered forming an Incast.

6.2.3 Measurement Units

It is a MUST to measure the number of ingress and egress ports. It is a MUST to have a non-null percentage of synchronization, which MUST be specified.

7 Application Throughput: Data Center Goodput

7.1. Definition

In Data Center Networking, a balanced network is a function of maximal throughput and minimal loss at any given time. This is captured by the Goodput [4]. Goodput is the application-level throughput. For standard TCP applications, a very small loss can have a dramatic effect on application throughput. [RFC2647] has a definition of Goodput; the definition in this publication is a variance.

Goodput is the number of bits per unit of time forwarded to the correct destination interface of the DUT, minus any bits retransmitted.

7.2. Discussion

In data center benchmarking, the goodput is a value that SHOULD be

measured. It provides a realistic idea of the usage of the available bandwidth. A goal in data center environments is to maximize the goodput while minimizing the loss.

7.3. Measurement Units

The Goodput, G, is then measured by the following formula:

G=(S/F) x V bytes per second

-S represents the payload bytes, which does not include packet or TCP headers

-F is the frame size

-V is the speed of the media in bytes per second

Example: A TCP file transfer over HTTP protocol on a 10GB/s media.

The file cannot be transferred over Ethernet as a single continuous stream. It must be broken down into individual frames of 1500B when the standard MTU (Maximum Transmission Unit) is used. Each packet requires 20B of IP header information and 20B of TCP header information; therefore 1460B are available per packet for the file transfer. Linux based systems are further limited to 1448B as they also carry a 12B timestamp. Finally, the date is transmitted in this example over Ethernet which adds a 26B overhead per packet.

G= 1460/1526 x 10 Gbit/s which is 9.567 Gbit per second or 1.196 GB per second.

Please note: This example does not take into consideration the additional Ethernet overhead, such as the interframe gap (a minimum of 96 bit times), nor collisions (which have a variable impact, depending on the network load).

When conducting Goodput measurements please document in addition to the 4.1 section the following information:

-The TCP Stack used

-OS Versions

-NIC firmware version and model

For example, Windows TCP stacks and different Linux versions can influence TCP based tests results.

8. 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 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.

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

9. IANA Considerations

NO IANA Action is requested at this time.

10. References

10.1. Normative References

[draft-ietf-bmwg-dcbench-methodology] Avramov L. and Rapp J., "Data Center Benchmarking Methodology", RFC "draft-ietf-bmwg-dcbench-methodology", DATE (to be updated once published)

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

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

[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>

[RFC5481] , Morton, A., "Packet Delay Variation Applicability Statement", BCP 14, RFC 5481, March 2009, <http://www.rfc-editor.org/info/rfc5481>

10.2. Informative References

[RFC2889] Mandeville R. and Perser J., "Benchmarking Methodology for LAN Switching Devices", RFC 2889, August 2000, <http://www.rfc-editor.org/info/rfc2889>

[RFC3918] Stopp D. and Hickman B., "Methodology for IP Multicast Benchmarking", RFC 3918, October 2004, <http://www.rfc-editor.org/info/rfc3918>

[4] Yanpei Chen, Rean Griffith, Junda Liu, Randy H. Katz, Anthony D. Joseph, "Understanding TCP Incast Throughput Collapse in Datacenter Networks, "http://yanpeichen.com/professional/usenixLoginIncastReady.pdf"

[RFC2432] Dubray, K., "Terminology for IP Multicast Benchmarking", BCP 14, RFC 2432, DOI 10.17487/RFC2432, October 1998, <http://www.rfc-editor.org/info/rfc2432>

[RFC2647] Newman D. ,"Benchmarking Terminology for Firewall Performance" BCP 14, RFC 2647, August 1999, <http://www.rfc-editor.org/info/rfc2647>

10.3. Acknowledgments

The authors would like to thank Alfred Morton, Scott Bradner, Ian Cox, Tim Stevenson for their reviews and feedback.

Authors' Addresses

Lucien Avramov
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043
United States
Phone: +1 408 774 9077
Email: lucien.avramov@gmail.com

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

Phone: +1 650 857 3367
Email: jrapp@vmware.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 2, 2017

W. Cerveny
Arbor Networks
R. Bonica
R. Thomas
Juniper Networks
March 1, 2017

Benchmarking The Neighbor Discovery Protocol
draft-ietf-bmwg-ipv6-nd-06

Abstract

This document provides benchmarking procedures for Neighbor Discovery Protocol (NDP). It also proposes metrics by which an NDP implementation's scaling capabilities can be measured.

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 September 2, 2017.

Copyright Notice

Table of Contents

1. Introduction

When an IPv6 node forwards a packet, it executes the following procedure:

- Identifies the outbound interface and IPv6 next-hop
- Queries a local Neighbor Cache (NC) to determine the IPv6 next-hop's link-layer address

- Encapsulates the packet in a link-layer header. The link-layer header includes the IPv6 next-hop's link-layer address
- Forwards the packet to the IPv6 next-hop

IPv6 nodes use the Neighbor Discovery Protocol (NDP) [RFC4861] to maintain the NC. Operational experience [RFC6583] shows that when an implementation cannot maintain a sufficiently complete NC, its ability to forward packets is impaired.

NDP, like any other protocol, consumes processing, memory, and bandwidth resources. Its ability to maintain a sufficiently complete NC depends upon the availability of the above-mentioned resources.

This document provides benchmarking procedures for NDP. Benchmarking procedures include a Baseline Test and an NDP Scaling Test. In both tests, the Device Under Test (DUT) is an IPv6 router. Two physical links (A and B) connect the DUT to a Tester. The Tester sends traffic through Link A to the DUT. The DUT forwards that traffic, through Link B, back to the Tester.

The above-mentioned traffic stream contains one or more interleaved flows. An IPv6 Destination Address uniquely identifies each flow. Or, said another way, every packet within a flow has the same IPv6 Destination Address.

In the Baseline Test, the traffic stream contains exactly one flow. Because every packet in the stream has the same IPv6 Destination Address, the DUT can forward the entire stream using exactly one NC entry. NDP is exercised minimally and no packet loss should be observed.

The NDP Scaling Test is identical to the Baseline Test, except that the traffic stream contains many flows. In order to forward the stream without loss, the DUT must maintain one NC entry for each flow. If the DUT cannot maintain one NC entry for each flow, packet loss will be observed and attributed to NDP scaling limitations.

This document proposes an NDP scaling metric, called NDP-MAX-NEIGHBORS. NDP-MAX-NEIGHBORS is the maximum number of neighbors to which an IPv6 node can send traffic during periods of high NDP activity.

The procedures described herein reveal how many IPv6 neighbors an NDP implementation can discover. They also provide a rough estimate of the time required to discover those neighbors. However, that estimate does not reflect the maximum rate at which the

implementation can discover neighbors. Maximum rate discovery is a topic for further exploration.

The test procedures described herein assume that NDP does not compete with other applications for resources on the DUT. When NDP competes for resources, its scaling characteristics may differ from those reported by the benchmarks described, and may vary over time.

2. Test Setup

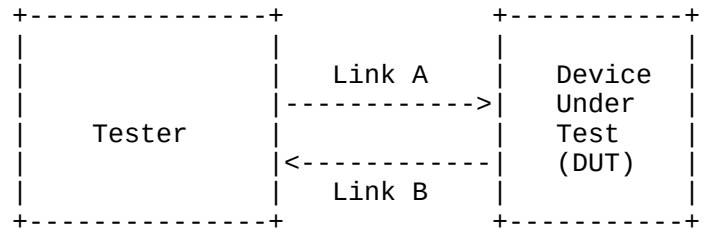


Figure 1: Test Setup

The DUT is an IPv6 router. Two links (A and B) connect the DUT to the Tester. Link A capabilities must be identical to Link B capabilities. For example, if the interface to Link A is a 10 Gigabit Ethernet port, the interface to Link B must also be a 10 Gigabit Ethernet port.

2.1. Device Under Test (DUT)

2.1.1. Interfaces

DUT interfaces are numbered as follows:

- Link A - 2001:2:0:0::2/64
- Link B- 2001:2:0:1::1/64

Both DUT interfaces should be configured with a 1500-byte MTU. However, if they cannot support a 1500-byte MTU, they may be configured with a 1280-byte MTU.

2.1.2. Neighbor Discovery Protocol (NDP)

NDP is enabled on both DUT interfaces. Therefore, the DUT emits both solicited and unsolicited Router Advertisement (RA) messages. The DUT emits an RA message at least once every 600 seconds and no more frequently than once every 200 seconds.

When the DUT sends an RA message, it includes the following information:

- Router Lifetime - 1800 seconds
- Reachable Time - 0 seconds
- Retrans Time - 0 seconds
- Source Link Layer Address - Link layer address of DUT interface
- M-bit is clear (0)
- O-bit is clear (0)

The above-mentioned values are chosen because they are the default values specified in RFC 4861.

NDP manages the NC. Each NC entry represents an on-link neighbor and is identified by the neighbor's on-link unicast IP address. As per RFC 4861, each NC entry needs to be refreshed periodically. NDP refreshes NC entries by exchanging Neighbor Solicitation (NS) and Neighbor Advertisement (NA) messages.

No static NC entries are configured on the DUT.

2.1.3. Routing

The DUT maintains a direct route to 2001:2:0:0/64 through Link A. It also maintains a direct route to 2001:2:0:1/64 through Link B. No static routes or dynamic routing protocols are configured on the DUT.

2.2. Tester

2.2.1. Interfaces

Interfaces are numbered as follows:

- Link A - 2001:2:0:0::1/64
- Link B - Multiple addresses are configured on Link B. These addresses are drawn sequentially from the 2001:2:0:1::/64 address block. The first address is 2001:2:0:1::2/64. Subsequent addresses are 2001:2:0:1::3/64, 2001:2:0:1::4/64, 2001:2:0:1::5/64, et cetera. The number of configured addresses should be the expected value of NDP-MAX-NEIGHBORS times 1.1.

Both Tester interfaces should be configured with a 1500-byte MTU. However, if they cannot support a 1500-byte MTU, they may be configured with a 1280-byte MTU.

2.2.2. Neighbor Discovery Protocol (NDP)

NDP is enabled on both Tester interfaces. Therefore, upon initiation, the Tester sends Router Solicitation (RS) messages and waits for Router Advertisement (RA) messages. The Tester also exchanges Neighbor Solicitation (NS) and Neighbor Advertisement (NA) messages with the DUT.

No static NC entries are configured on the Tester.

2.2.3. Routing

The Tester maintains a direct route to 2001:2:0:0/64 through Link A. It also maintains a direct route to 2001:2:0:1/64 through Link B. No static routes or dynamic routing protocols are configured on the Tester.

2.2.4. Test Traffic

The Tester sends a stream of test traffic through Link A to the DUT. The test traffic stream contains one or more interleaved flows. Flows are numbered 1 through N, sequentially.

Within each flow, each packet contains an IPv6 header and each IPv6 header contains the following information:

- Version - 6
- Traffic Class - 0
- Flow Label - 0
- Payload Length - 0
- Next Header - IPv6-NoNxt (59)
- Hop Limit - 255
- Source Address - 2001:2:0:0::1
- Destination Address - The first 64 bits of the Destination Address are 2001:2:0:1::. The next 64 are uniquely associated with the flow. Every packet in the first flow carries the Destination address 2001:2:0:1::2. Every subsequent flow has an IP address

one greater than the last (i.e., 2001:2:0:1::3, 2001:2:0:1::4, etc.)

In order to avoid link congestion, test traffic is offered at a rate not to exceed 50% of available link bandwidth. In order to avoid burstiness and buffer occupancy, every packet in the stream is exactly 40 bytes long (i.e., the length of an IPv6 header with no IPv6 payload). Furthermore, the gap between packets is identical.

During the course of a test, the number of flows that the test stream contains may increase. When this occurs, the rate at which test traffic is offered remains constant. For example, assume that a test stream is offered at a rate of 1,000 packets per second. This stream contains two flows, each contributing 500 packets per second to the 1,000 packet per second aggregate. When a third stream is added to the flow, all three streams must contribute 333 packets per second in order to maintain the 1,000 packet per second limit. (As in this example, rounding error is acceptable.)

The DUT attempts to forward every packet in the test stream through Link B to the Tester. It does this because:

- Every packet in the test stream has a destination address drawn from the 2001:2:0:1::/64 address block
- The DUT has a direct route to 2001:2:0:1/64 through Link B

2.2.5. Counters

On the Tester, two counters are configured for each flow. One counter, configured on Link A, increments when the Tester sends a packet belonging to the flow. The other counter, configured on Link B, increments when the Tester receives packet from the flow. In order for a packet to be associated with a flow, the following conditions must all be true:

- The IPv6 Destination Address must be that of the flow
- The IPv6 Next Header must be IPv6-NoNxt (59)

The following counters also are configured on both Tester Interfaces:

- RS packets sent
- RA packets received
- NS packets sent

- NS packets received
- NA packets sent
- NA packets received
- Total packets sent
- Total packets received

3. Tests

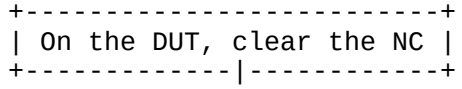
3.1. Baseline Test

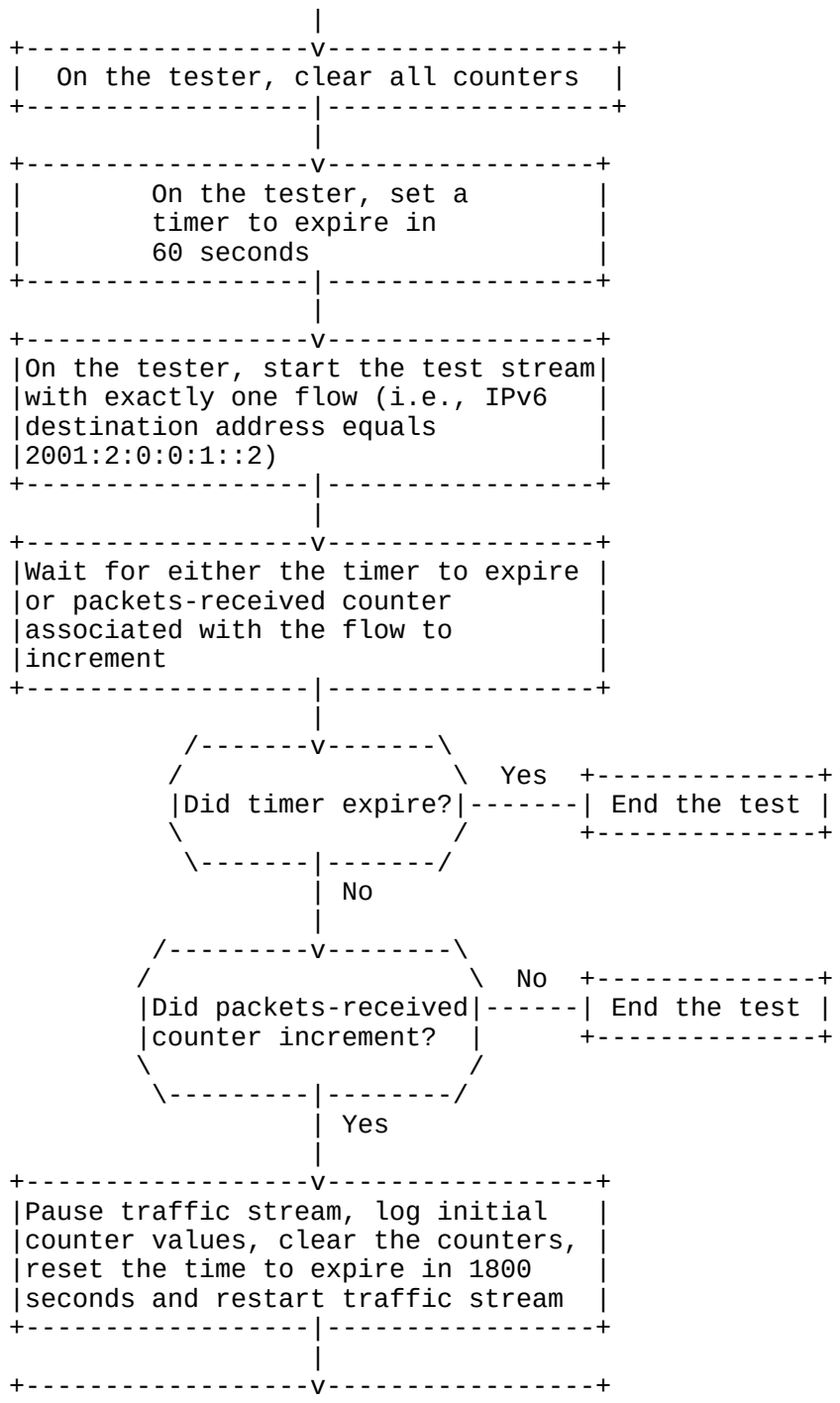
The purpose of the Baseline Test is to ensure that the DUT can forward every packet in the test stream, without loss, when NDP is minimally exercised and not operating near its scaling limit.

3.1.1. Procedure

- On the DUT, clear the NC
- On the Tester, clear all counters
- On the Tester, set a timer to expire in 60 seconds
- On the Tester, start the test stream with exactly one flow (i.e., IPv6 Destination Address equals 2001:2:0:1::2)
- Wait for either the timer to expire or the packets-received counter associated with the flow to increment
- If the timer expires, stop the test stream and end the test
- If the packets-received counter increments, pause the traffic stream, log the initial counter values, clear the counters, reset the timer to expire in 1800 seconds and restart the traffic stream
- When the timer expires, stop the test stream, wait sufficient time for any queued packets to exit, log the final counter values and end the test

3.1.2. Baseline Test Procedure Flow Chart




On the tester, clear all counters
On the tester, set a timer to expire in 60 seconds
On the tester, start the test stream with exactly one flow (i.e., IPv6 destination address equals 2001:2:0:0:1::2)
Wait for either the timer to expire or packets-received counter associated with the flow to increment
Did timer expire?
Yes
End the test
No
Did packets-received counter increment?
No
End the test
Yes
Pause traffic stream, log initial counter values, clear the counters, reset the time to expire in 1800 seconds and restart traffic stream

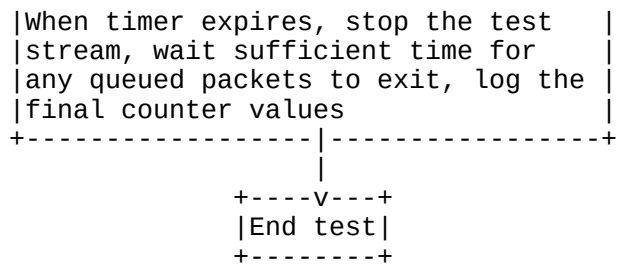


Figure 2: Baseline Test Procedure Flow Chart

3.1.3. Results

The log contains initial and final values for the following counters:

- packets-sent
- packets-received

The initial values of packets-sent and packets-received may be equal to one another. If these values are identical, none of the initial packets belonging to the flow were lost. However, if the initial value of packets-sent is greater than the initial value of packets-received, initial packets were lost. This loss of initial packets is acceptable.

The final values of packets-sent and packets-received should be equal to one another. If they are not, an error has occurred. Because this error is likely to affect Scaling Test results, the error must be corrected before the Scaling Test is executed.

3.2. Scaling Test

The purpose of the Scaling Test is to discover the number of neighbors to which an IPv6 node can send traffic during periods of high NDP activity. We call this number NDP-MAX-NEIGHBORS.

3.2.1. Procedure

Execute the following procedure:

- On the DUT, clear the NC
- On the Tester, clear all counters
- On the Tester, set a timer to expire in 60 seconds

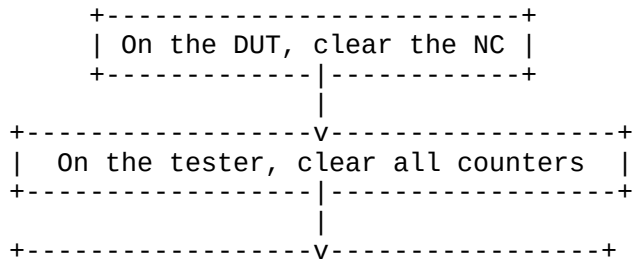
- On the Tester, start the test stream with exactly one flow (i.e., IPv6 Destination Address equals 2001:2:0:1::2)
- Wait for either the timer to expire or the packets-received counter associated with the flow to increment
- If the timer expires, stop the test stream and end the test
- If the packets-received counter increments, proceed as described below:

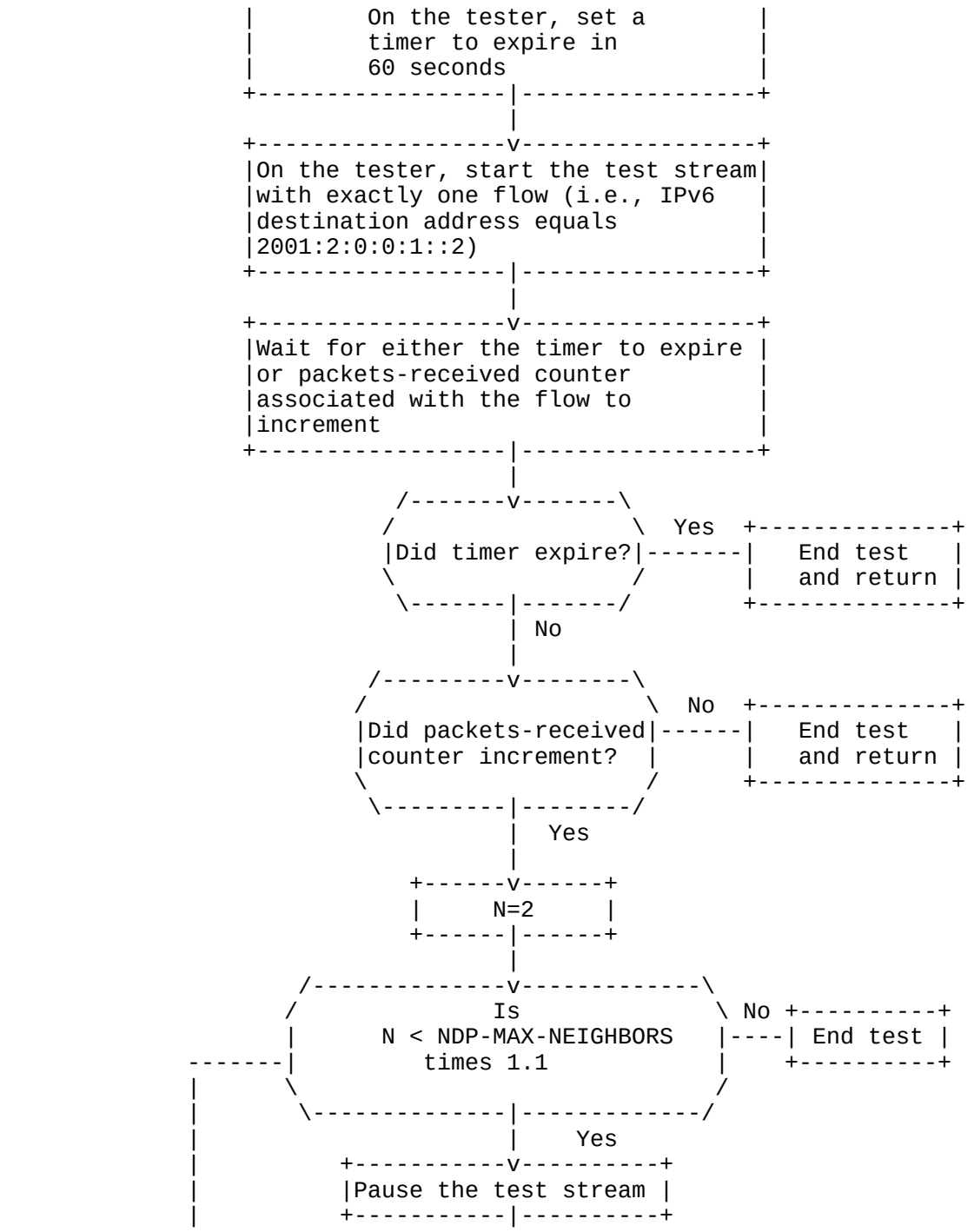
Execute the following procedure N times, starting at 2 and ending at the number of expected value of NDP-MAX-NEIGHBORS times 1.1.

- Pause the test stream
- Log the time and the value of N minus one
- Clear the packets-sent and packets-received counters associated with the previous flow (i.e., N minus one)
- Reset the timer to expire in 60 seconds
- Add the next flow to the test stream (i.e.,IPv6 Destination Address is a function of N)
- Restart the test stream
- Wait for either the timer to expire or the packets-received counter associated with the new flow to increment

After the above described procedure had been executed N times, clear the timer and reset it to expire in 1800 seconds. When the timer expires, stop the stream, log all counters and end the test (after waiting sufficient time for any queued packets to exit).

3.2.2. Scaling Test Procedure Flow Chart




On the tester, set a timer to expire in 60 seconds
On the tester, start the test stream with exactly one flow (i.e., IPv6 destination address equals 2001:2:0:0:1::2)
Wait for either the timer to expire or packets-received counter associated with the flow to increment
Did timer expire?
Yes
End test and return
No
Did packets-received counter increment?
No
End test and return
Yes
N=2
Is N < NDP-MAX-NEIGHBORS times 1.1
No
End test
Yes
Pause the test stream

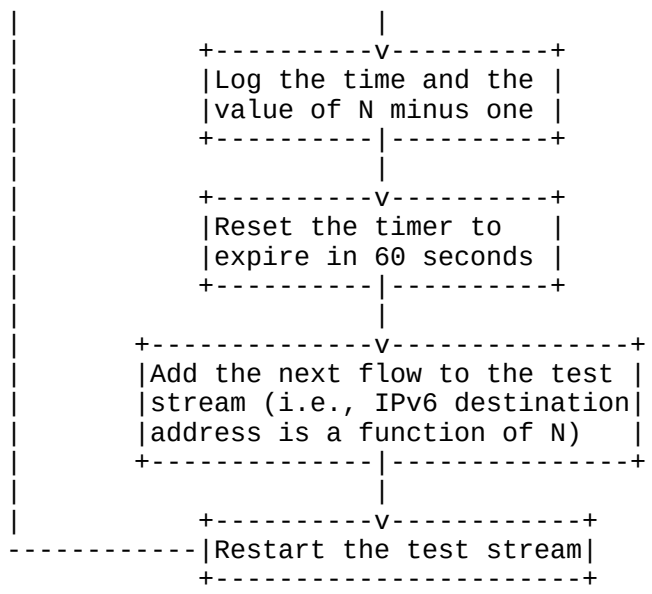


Figure 3: Scaling Test Procedure Flow Chart

3.2.3. Results

The test report includes the following:

- A description of the DUT (make, model, processor, memory, interfaces)
- Rate at which the Tester offers test traffic to the DUT (measured in packets per second)
- A log that records the time at which each flow was introduced to the test stream and the final value of all counters
- The expected value of NDP-MAX-NEIGHBORS
- The actual value of NDP-MAX-NEIGHBORS

NDP-MAX-NEIGHBORS is equal to the number of counter pairs where packets-sent is equal to packets-received. Two counters are members of a pair if they are both associated with the same flow. If packets-sent is equal to packets-recieved for every counter pair, the test should be repeated with a larger expected value of NDP-MAX-NEIGHBORS.

If an implementation abides by the recommendation of Section 7.1 of RFC 6583, for any given counter pair, packets-received will either be equal to zero or packets-sent.

The log documents the time at which each flow was introduced to the test stream. This log reveals the effect of NC size to the time required to discover a new IPv6 neighbor.

4. Measurements Explicitly Excluded

These are measurements which aren't recommended because of the itemized reasons below:

4.1. DUT CPU Utilization

This measurement relies on the DUT to provide utilization information, which is not externally observable (not black-box). However, some testing organizations may find the CPU utilization is useful auxiliary information specific to the DUT model, etc.

4.2. Malformed Packets

This benchmarking test is not intended to test DUT behavior in the presence of malformed packets.

5. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

6. 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 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. Acknowledgments

Helpful comments and suggestions were offered by Al Morton, Joel Jaeggli, Nalini Elkins, Scott Bradner, and Ram Krishnan, on the BMWG e-mail list and at BMWG meetings. Precise grammatical corrections and suggestions were offered by Ann Cerveny.

8. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, March 2012.

Authors' Addresses

Bill Cerveny
Arbor Networks
2727 South State Street
Ann Arbor, MI 48104
USA

Email: wcerveny@arbor.net

Ron Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, VA 20170
USA

Email: rbonica@juniper.net

Reji Thomas
Juniper Networks
Elnath-Exora Business Park Survey
Bangalore, KA 560103
India

Email: rejithomas@juniper.net

Benchmarking Working Group
Internet Draft
Intended status: Informational
Expires: April 2017

M. Georgescu
L. Pislaru
RCS&RDS
G. Lencse
Szechenyi Istvan University
October 27, 2016

Benchmarking Methodology for IPv6 Transition Technologies
draft-ietf-bmwg-ipv6-tran-tech-benchmarking-03.txt

Abstract

There are benchmarking methodologies addressing the performance of network interconnect devices that are IPv4- or IPv6-capable, but the IPv6 transition technologies are outside of their scope. This document provides complementary guidelines for evaluating the performance of IPv6 transition technologies. More specifically, this document targets IPv6 transition technologies that employ encapsulation or translation mechanisms, as dual-stack nodes can be very well tested using the recommendations of RFC2544 and RFC5180. The methodology also includes a tentative metric for benchmarking load scalability.

This Internet-Draft will expire on April 27, 2016.

Table of Contents

1. Introduction

The methodologies described in [RFC2544] and [RFC5180] help vendors and network operators alike analyze the performance of IPv4 and IPv6-capable network devices. The methodology presented in [RFC2544] is mostly IP version independent, while [RFC5180] contains complementary recommendations, which are specific to the latest IP version, IPv6. However, [RFC5180] does not cover IPv6 transition technologies.

IPv6 is not backwards compatible, which means that IPv4-only nodes cannot directly communicate with IPv6-only nodes. To solve this issue, IPv6 transition technologies have been proposed and implemented.

This document presents benchmarking guidelines dedicated to IPv6 transition technologies. The benchmarking tests can provide insights about the performance of these technologies, which can act as useful

feedback for developers, as well as for network operators going through the IPv6 transition process.

The document also includes an approach to quantify performance when operating in overload. Overload scalability can be defined as a system's ability to gracefully accommodate greater numbers of flows than the maximum number of flows which the DUT can operate normally. The approach taken here is to quantify the overload scalability by measuring the performance created by an excessive number of network flows, and comparing performance to the non-overloaded case.

1.1. IPv6 Transition Technologies

Two of the basic transition technologies, dual IP layer (also known as dual stack) and encapsulation are presented in [RFC4213]. IPv4/IPv6 Translation is presented in [RFC6144]. Most of the transition technologies employ at least one variation of these mechanisms. Some of the more complex ones (e.g. DSLite [RFC6333]) are using all three. In this context, a generic classification of the transition technologies can prove useful.

Tentatively, we can consider a production network transitioning to IPv6 as being constructed using the following IP domains:

- Domain A: IPvX specific domain
- Core domain: which may be IPvY specific or dual-stack(IPvX and IPvY)
- Domain B: IPvX specific domain

Note: X,Y are part of the set {4,6}, and X NOT.EQUAL Y.

According to the technology used for the core domain traversal the transition technologies can be categorized as follows:

1. Single Translation: In this case, the production network is assumed to have only two domains, Domain A and the Core domain. The core domain is assumed to be IPvY specific. IPvX packets are translated to IPvY at the edge between Domain A and the Core domain.
2. Dual-stack: the core domain devices implement both IP protocols

3. Encapsulation: The production network is assumed to have all three domains, Domains A and B are IPvX specific, while the core domain is IPvY specific. An encapsulation mechanism is used to traverse the core domain. The IPvX packets are encapsulated to IPvY packets at the edge between Domain A and the Core domain. Subsequently, the IPvY packets are de-encapsulated at the edge between the Core domain and Domain B.

4. Double translation: The production network is assumed to have all three domains, Domains A and B are IPvX specific, while the core domain is IPvY specific. A translation mechanism is employed for the traversal of the core network. The IPvX packets are translated to IPvY packets at the edge between Domain A and the Core domain. Subsequently, the IPvY packets are translated back to IPvX at the edge between the Core domain and Domain B.

The performance of Dual-stack transition technologies can be fully evaluated using the benchmarking methodologies presented by [RFC2544] and [RFC5180]. Consequently, this document focuses on the other 3 categories: Single translation, Encapsulation and Double translation transition technologies.

Another important aspect by which the IPv6 transition technologies can be categorized is their use of stateful or stateless mapping algorithms. The technologies that use stateful mapping algorithms (e.g. Stateful NAT64 [RFC6146]) create dynamic correlations between IP addresses or {IP address, transport protocol, transport port number} tuples, which are stored in a state table. For ease of reference, the IPv6 transition technologies which employ stateful mapping algorithms will be called stateful IPv6 transition technologies. The efficiency with which the state table is managed can be an important performance indicator for these technologies. Hence, for the stateful IPv6 transition technologies additional benchmarking tests are RECOMMENDED.

Table 1 contains the generic categories as well as associations with some of the IPv6 transition technologies proposed in the IETF.

Table 1. IPv6 Transition Technologies Categories

	Generic category	IPv6 Transition Technology
1	Dual-stack	Dual IP Layer Operations [RFC4213]
2	Single translation	NAT64 [RFC6146], IVI [RFC6219]
3	Double translation	464XLAT [RFC6877], MAP-T [RFC7599]
4	Encapsulation	DSLite[RFC6333], MAP-E [RFC7597] Lightweight 4over6 [RFC7596] 6RD [RFC5569]

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 [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 [RFC2119] significance.

Although these terms are usually associated with protocol requirements, in this document the terms are requirements for users and systems that intend to implement the test conditions and claim conformance with this specification.

3. Terminology

A number of terms used in this memo have been defined in other RFCs. Please refer to those RFCs for definitions, testing procedures and reporting formats.

Throughput (Benchmark) - [RFC2544]

Frame Loss Rate (Benchmark) - [RFC2544]

Back-to-back Frames (Benchmark) - [RFC2544]

System Recovery (Benchmark) - [RFC2544]

Reset (Benchmark) - [RFC6201]

Concurrent TCP Connection Capacity (Benchmark) - [RFC3511]

Maximum TCP Connection Establishment Rate (Benchmark) - [RFC3511]

4. Test Setup

The test environment setup options recommended for IPv6 transition technologies benchmarking are very similar to the ones presented in Section 6 of [RFC2544]. In the case of the tester setup, the options presented in [RFC2544] and [RFC5180] can be applied here as well. However, the Device under test (DUT) setup options should be explained in the context of the targeted categories of IPv6 transition technologies: Single translation, Double translation and Encapsulation transition technologies.

Although both single tester and sender/receiver setups are applicable to this methodology, the single tester setup will be used to describe the DUT setup options.

For the test setups presented in this memo, dynamic routing SHOULD be employed. However, the presence of routing and management frames can represent unwanted background data that can affect the benchmarking result. To that end, the procedures defined in [RFC2544] (Sections 11.2 and 11.3) related to routing and management frames SHOULD be used here as well. Moreover, the "Trial description" recommendations presented in [RFC2544] (Section 23) are valid for this memo as well.

In terms of route setup, the recommendations of [RFC2544] Section 13 are valid for this document as well assuming that an IPv6 version of the routing packets shown in appendix C.2.6.2 is used.

4.1. Single translation Transition Technologies

For the evaluation of Single translation transition technologies, a single DUT setup (see Figure 1) SHOULD be used. The DUT is responsible for translating the IPvX packets into IPvY packets. In this context, the tester device should be configured to support both IPvX and IPvY.

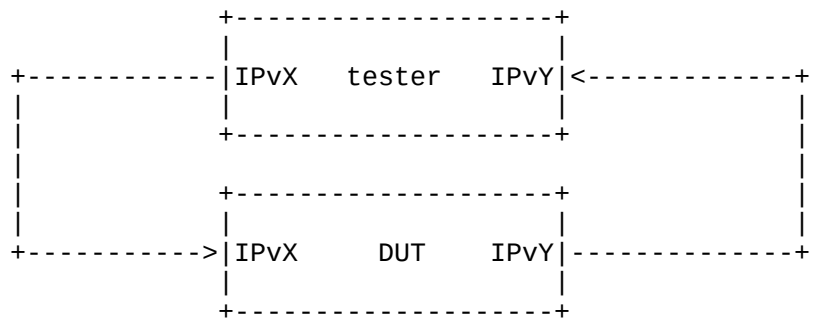


Figure 1. Test setup 1

4.2. Encapsulation/Double translation Transition Technologies

For evaluating the performance of Encapsulation and Double translation transition technologies, a dual DUT setup (see Figure 2) SHOULD be employed. The tester creates a network flow of IPvX packets. The first DUT is responsible for the encapsulation or translation of IPvX packets into IPvY packets. The IPvY packets are de-encapsulated/translated back to IPvX packets by the second DUT and forwarded to the tester.

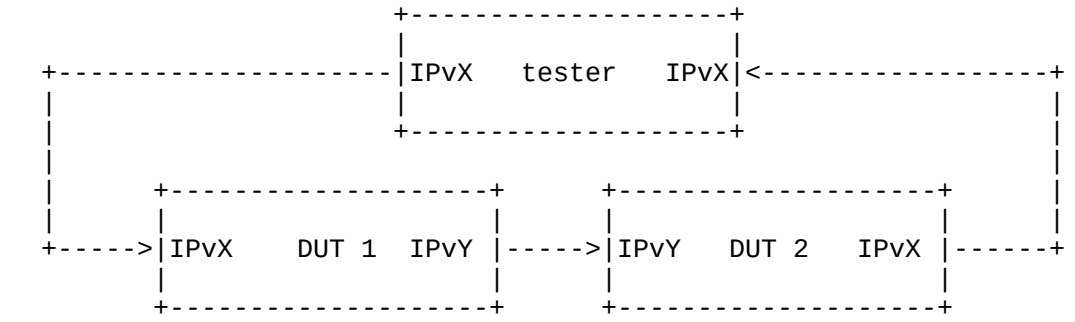


Figure 2. Test setup 2

One of the limitations of the dual DUT setup is the inability to reflect asymmetries in behavior between the DUTs. Considering this, additional performance tests SHOULD be performed using the single DUT setup.

Note: For encapsulation IPv6 transition technologies, in the single DUT setup, in order to test the de-encapsulation efficiency, the tester SHOULD be able to send IPvX packets encasulated as IPvY.

5. Test Traffic

The test traffic represents the experimental workload and SHOULD meet the requirements specified in this section. The requirements are dedicated to unicast IP traffic. Multicast IP traffic is outside of the scope of this document.

5.1. Frame Formats and Sizes

[RFC5180] describes the frame size requirements for two commonly used media types: Ethernet and SONET (Synchronous Optical Network). [RFC2544] covers also other media types, such as token ring and FDDI. The two documents can be referred for the dual-stack

transition technologies. For the rest of the transition technologies the frame overhead introduced by translation or encapsulation MUST be considered.

The encapsulation/translation process generates different size frames on different segments of the test setup. For instance, the single translation transition technologies will create different frame sizes on the receiving segment of the test setup, as IPvX packets are translated to IPvY. This is not a problem if the bandwidth of the employed media is not exceeded. To prevent exceeding the limitations imposed by the media, the frame size overhead needs to be taken into account when calculating the maximum theoretical frame rates. The calculation method for the Ethernet, as well as a calculation example are detailed in Appendix A. The details of the media employed for the benchmarking tests MUST be noted in all test reports.

In the context of frame size overhead, MTU recommendations are needed in order to avoid frame loss due to MTU mismatch between the virtual encapsulation/translation interfaces and the physical network interface controllers (NICs). To avoid this situation, the larger MTU between the physical NICs and virtual encapsulation/translation interfaces SHOULD be set for all interfaces of the DUT and tester. To be more specific, the minimum IPv6 MTU size (1280 bytes) plus the encapsulation/translation overhead is the RECOMMENDED value for the physical interfaces as well as virtual ones.

5.1.1. Frame Sizes to Be Used over Ethernet

Based on the recommendations of [RFC5180], the following frame sizes SHOULD be used for benchmarking IPvX/IPvY traffic on Ethernet links: 64, 128, 256, 512, 768 1024, 1280, 1518, 1522, 2048, 4096, 8192 and 9216.

The theoretical maximum frame rates considering an example of frame overhead are presented in Appendix A1.

5.2. Protocol Addresses

The selected protocol addresses should follow the recommendations of [RFC5180](Section 5) for IPv6 and [RFC2544](Section 12) for IPv4.

Note: testing traffic with extension headers might not be possible for the transition technologies, which employ translation. Proposed IPvX/IPvY translation algorithms such as IP/ICMP translation [RFC7915] do not support the use of extension headers.

5.3. Traffic Setup

Following the recommendations of [RFC5180], all tests described SHOULD be performed with bi-directional traffic. Uni-directional traffic tests MAY also be performed for a fine grained performance assessment.

Because of the simplicity of UDP, UDP measurements offer a more reliable basis for comparison than other transport layer protocols. Consequently, for the benchmarking tests described in Section 6 of this document UDP traffic SHOULD be employed.

Considering that a transition technology could process both native IPv6 traffic and translated/encapsulated traffic, the following traffic setups are recommended:

i) IPvX only traffic (where the IPvX traffic is to be translated/encapsulated by the DUT)
ii) 90% IPvX traffic and 10% IPvY native traffic
iii) 50% IPvX traffic and 50% IPvY native traffic
iv) 10% IPvX traffic and 90% IPvY native traffic

For the benchmarks dedicated to stateful IPv6 transition technologies, included in Section 8 of this memo (Concurrent TCP Connection Capacity and Maximum TCP Connection Establishment Rate), the traffic SHOULD follow the recommendations of [RFC3511], Sections 5.2.2.2 and 5.3.2.2.

6. Modifiers

The idea of testing under different operational conditions was first introduced in [RFC2544](Section 11) and represents an important aspect of benchmarking network elements, as it emulates to some extent the conditions of a production environment. Section 6 of [RFC5180] describes complementary testing conditions specific to IPv6. Their recommendations can be referred for IPv6 transition technologies testing as well.

7. Benchmarking Tests

The following sub-sections contain the list of all recommended benchmarking tests.

7.1. Throughput

Use Section 26.1 of RFC2544 unmodified.

7.2. Latency

Objective: To determine the latency. Typical latency is based on the definitions of latency from [RFC1242]. However, this memo provides a new measurement procedure.

Procedure: Similar to [RFC2544], the throughput for DUT at each of the listed frame sizes SHOULD be determined. Send a stream of frames at a particular frame size through the DUT at the determined throughput rate to a specific destination. The stream SHOULD be at least 120 seconds in duration.

Identifying tags SHOULD be included in at least 500 frames after 60 seconds. For each tagged frame, the time at which the frame was fully transmitted (timestamp A) and the time at which the frame was received (timestamp B) MUST be recorded. The latency is timestamp B minus timestamp A as per the relevant definition from RFC 1242, namely latency as defined for store and forward devices or latency as defined for bit forwarding devices.

From the resulted (at least 500) latencies, 2 quantities SHOULD be calculated. One is the typical latency, which SHOULD be calculated with the following formula:

TL=Median(Li)

Where: TL - the reported typical latency of the stream

Li -the latency for tagged frame i

The other measure is the worst case latency, which SHOULD be calculated with the following formula:

WCL=L99.9thPercentile

Where: WCL - The reported worst case latency

L99.9thPercentile - The 99.9th Percentile of the stream measured latencies

The test MUST be repeated at least 20 times with the reported value being the median of the recorded values for TL and WCL.

Reporting Format: The report MUST state which definition of latency (from RFC 1242) was used for this test. The summarized latency

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, the rate at which the latency test was run for that frame size, for the media types tested, and for the resultant typical latency and worst case latency values for each type of data stream tested. To account for the variation, the 1st and 99th percentiles of the 20 iterations MAY be reported in two separated columns. For a fine grain analysis, the histogram (as exemplified in [RFC5481] Section 4.4) of one of the iterations MAY be displayed as well.

7.3. Packet Delay Variation

Considering two of the metrics presented in [RFC5481], Packet Delay Variation (PDV) and Inter Packet Delay Variation (IPDV), it is RECOMMENDED to measure PDV. For a fine grain analysis of delay variation, IPDV measurements MAY be performed as well.

7.3.1. PDV

Objective: To determine the Packet Delay Variation as defined in [RFC5481].

Procedure: As described by [RFC2544], first determine the throughput for the DUT at each of the listed frame sizes. Send a stream of frames at a particular frame size through the DUT at the determined throughput rate to a specific destination. The stream SHOULD be at least 60 seconds in duration. Measure the One-way delay as described by [RFC3393] for all frames in the stream. Calculate the PDV of the stream using the formula:

PDV=D99.9thPercentile - Dmin

Where: D99.9thPercentile - the 99.9th Percentile (as it was described in [RFC5481]) of the One-way delay for the stream

Dmin - the minimum One-way delay in the stream

As recommended in [RFC2544], the test MUST be repeated at least 20 times with the reported value being the median of the recorded values. Moreover, the 1st and 99th percentiles SHOULD be calculated to account for the variation of the dataset.

Reporting Format: The PDV results SHOULD be reported in a table with a row for each of the tested frame sizes and columns for the frame size and the applied frame rate for the tested media types. Two columns for the 1st and 99th percentile values MAY as well be

displayed. Following the recommendations of [RFC5481], the RECOMMENDED units of measurement are milliseconds.

7.3.2. IPDV

Objective: To determine the Inter Packet Delay Variation as defined in [RFC5481].

Procedure: As described by [RFC2544], first determine the throughput for the DUT at each of the listed frame sizes. Send a stream of frames at a particular frame size through the DUT at the determined throughput rate to a specific destination. The stream SHOULD be at least 60 seconds in duration. Measure the One-way delay as described by [RFC3393] for all frames in the stream. Calculate the IPDV for each of the frames using the formula:

IPDV(i)=D(i) - D(i-1)

Where: D(i) - the One-way delay of the i th frame in the stream

Given the nature of IPDV, reporting a single number might lead to over-summarization. In this context, the report for each measurement SHOULD include 3 values: Dmin, Dmed, and Dmax

Where: Dmin - the minimum IPDV in the stream

Dmed - the median IPDV of the stream

Dmax - the maximum IPDVin the stream

As recommended in [RFC 2544], the test MUST be repeated at least 20 times. To summarize the 20 repetitions, for each of the 3 (Dmin, Dmed and Dmax) the median value SHOULD be reported.

Reporting format: The median for the 3 proposed values SHOULD be reported. The IPDV results SHOULD be reported in a table with a row for each of the tested frame sizes. The columns SHOULD include the frame size and associated frame rate for the tested media types and sub-columns for the three proposed reported values. Following the recommendations of [RFC5481], the RECOMMENDED units of measurement are milliseconds.

7.4. Frame Loss Rate

Use Section 26.3 of [RFC2544] unmodified.

7.5. Back-to-back Frames

Use Section 26.4 of [RFC2544] unmodified.

7.6. System Recovery

Use Section 26.5 of[RFC2544]unmodified.

7.7. Reset

Use Section 26.6 of [RFC2544] unmodified.

8. Additional Benchmarking Tests for Stateful IPv6 Transition Technologies

This section describes additional tests dedicated to the stateful IPv6 transition technologies. For the tests described in this section the DUT devices SHOULD follow the test setup and test parameters recommendations presented in [RFC3511] (Sections 4, 5).

In addition to the IPv4/IPv6 transition function, a network node can have a firewall function. This document is targeting only the network devices that do not have a firewall function, as this function can be benchmarked using the recommendations of [RFC3511]. Consequently, only the tests described in [RFC3511] (Sections 5.2, 5.3) are RECOMMENDED. Namely, the following additional tests SHOULD be performed:

8.1. Concurrent TCP Connection Capacity

Use Section 5.3 of [RFC3511] unmodified.

8.2. Maximum TCP Connection Establishment Rate

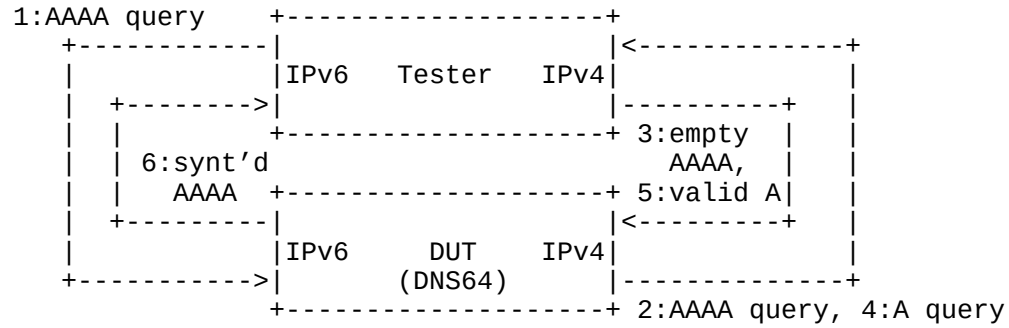
Use Section 5.3 of RFC3511 unmodified.

9. DNS Resolution Performance

This section describes benchmarking tests dedicated to DNS64 (see [RFC6147]), used as DNS support for single translation technologies such as NAT64.

9.1. Test and Traffic Setup

The test setup follows the setup proposed for single translation IPv6 transition technologies in Figure 1.



The test traffic SHOULD follow the following steps.

1. Query for the AAAA record of a domain name (from client to DNS64 server)

2. Query for the AAAA record of the same domain name (from DNS64 server to authoritative DNS server)

3. Empty AAAA record answer (from authoritative DNS server to DNS64 server)

4. Query for the A record of the same domain name (from DNS64 server to authoritative DNS server)

5. Valid A record answer (from authoritative DNS server to DNS64 server)

6. Synthesized AAAA record answer (from DNS64 server to client)

The Tester plays the role of DNS client as well as authoritative DNS server. It MAY be realized as a single physical device, or alternatively, two physical devices MAY be used.

Please note that:

- If the DNS64 server implements caching and there is a cache hit then step 1 is followed by step 6 (and steps 2 through 5 are omitted).
- If the domain name has an AAAA record then it is returned in step 3 by the authoritative DNS server, steps 4 and 5 are omitted, and the DNS64 server does not synthesizes an AAAA record, but returns the received AAAA record to the client.

- As for the IP version used between the tester and the DUT, IPv6 MUST be used between the client and the DNS64 server (as a DNS64 server provides service for an IPv6-only client), but either IPv4 or IPv6 MAY be used between the DNS64 server and the authoritative DNS server.

9.2. Benchmarking DNS Resolution Performance

Objective: To determine DNS64 performance by means of the maximum number of successfully processed DNS requests per second.

Procedure: Send a specific number of DNS queries at a specific rate to the DUT and then count the replies received in time (within a predefined timeout period from the sending time of the corresponding query, having the default value 1 second) from the DUT. If the count of sent queries is equal to the count of received replies, the rate of the queries is raised and the test is rerun. If fewer replies are received than queries were sent, the rate of the queries is reduced and the test is rerun. The duration of each trial SHOULD be at least 60 seconds to reduce the potential gain of a DNS64 server, which is able to exhibit higher performance by storing the requests and thus utilizing also the timeout time for answering them. For the same reason, no higher timeout time than 1 second SHOULD be used.

The maximum number of processed DNS queries per second is the fastest rate at which the count of DNS replies sent by the DUT is equal to the number of DNS queries sent to it by the test equipment.

The test SHOULD be repeated at least 20 times and the median and 1st /99th percentiles of the number of processed DNS queries per second SHOULD be calculated.

Details and parameters:

1. Caching
First, all the DNS queries MUST contain different domain names (or domain names MUST NOT be repeated before the cache of the DUT is exhausted). Then new tests MAY be executed with domain names, 20%, 40%, 60%, 80% and 100% of which are cached. We note that ensuring a record being cached requires repeating it both "late enough" after the first query to be already resolved and be present in the cache and "early enough" to be still present in the cache.

2. Existence of AAAA record
First, all the DNS queries MUST contain domain names which do not have an AAAA record and have exactly one A record.
Then new tests MAY be executed with domain names, 20%, 40%, 60%, 80% and 100% of which have an AAAA record.

Please note that the two conditions above are orthogonal, thus all their combinations are possible and MAY be tested. The testing with 0% cached domain names and with 0% existing AAAA record is REQUIRED and the other combinations are OPTIONAL. (When all the domain names are cached then the results do not depend on what percentage of the domain names have AAAA records, thus these combinations are not worth testing one by one.)

Reporting format: The primary result of the DNS64/DNS46 test is the average of the number of processed DNS queries per second measured with the above mentioned "0% + 0% combination". The average SHOULD be complemented with the margin of error to show the stability of the result. If optional tests are done, the median and the 1st and 99th percentiles MAY be presented in a two dimensional table where the dimensions are the proportion of the repeated domain names and the proportion of the DNS names having AAAA records. The two table headings SHOULD contain these percentage values. Alternatively, the results MAY be presented as the corresponding two dimensional graph, too. In this case the graph SHOULD show the median values with the percentiles as error bars. From both the table and the graph, one dimensional excerpts MAY be made at any given fixed percentage value of the other dimension. In this case, the fixed value MUST be given together with a one dimensional table or graph.

9.2.1. Requirements for the Tester

Before a Tester can be used for testing a DUT at rate r queries per second with t seconds timeout, it MUST perform a self-test in order to exclude the possibility that the poor performance of the Tester itself influences the results. For performing a self-test, the tester is looped back (leaving out DUT) and its authoritative DNS server subsystem is configured to be able to answer all the AAAA record queries. For passing the self-test, the Tester SHOULD be able to answer AAAA record queries at 2*(r+delta) rate within 0.25*t timeout, where the value of delta is at least 0.1.

Explanation: When performing DNS64 testing, each AAAA record query may result in at most two queries sent by the DUT, the first one of them is for an AAAA record and the second one is for an A record (the are both sent when there is no cache hit and also no AAAA record exists). The parameters above guarantee that the authoritative DNS server subsystem of the DUT is able to answer the queries at the required frequency using up not more than the half of the timeout time.

Remark: a sample open-source test program, dns64perf++ is available from [Dns64perf] and it is documented in [Lencse]. It implements only the client part of the Tester and it should be used together

with an authoritative DNS server implementation, e.g. BIND, NSD or YADIFA.

10. Overload Scalability

Scalability has been often discussed; however, in the context of network devices, a formal definition or a measurement method has not yet been proposed. In this context, we can define overload scalability as the ability of each transition technology to accommodate network growth. Poor scalability usually leads to poor performance. Considering this, overload scalability can be measured by quantifying the network performance degradation associated with an increased number of network flows.

The following subsections describe how the test setups can be modified to create network growth and how the associated performance degradation can be quantified.

10.1. Test Setup

The test setups defined in Section 3 have to be modified to create network growth.

10.1.1. Single Translation Transition Technologies

In the case of single translation transition technologies the network growth can be generated by increasing the number of network flows generated by the tester machine (see Figure 3).

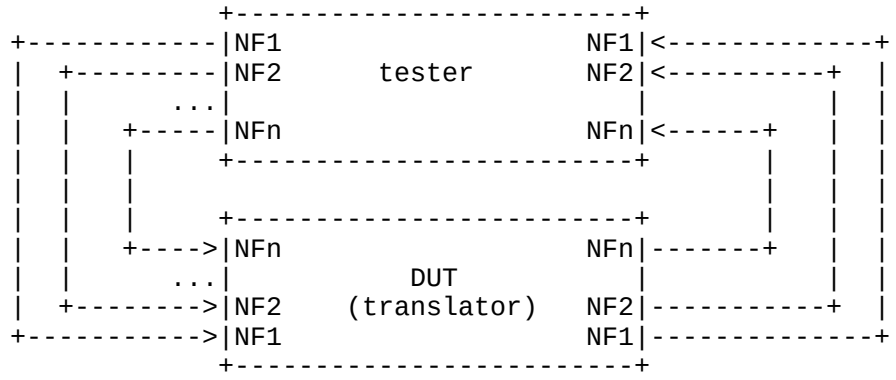


Figure 3. Test setup 3

10.1.2. Encapsulation/Double Translation Transition Technologies

Similarly, for the encapsulation/double translation technologies a multi-flow setup is recommended. Considering a multipoint-to-point scenario, for most transition technologies, one of the edge nodes is designed to support more than one connecting devices. Hence, the recommended test setup is a n:1 design, where n is the number of client DUTs connected to the same server DUT (See Figure 4).

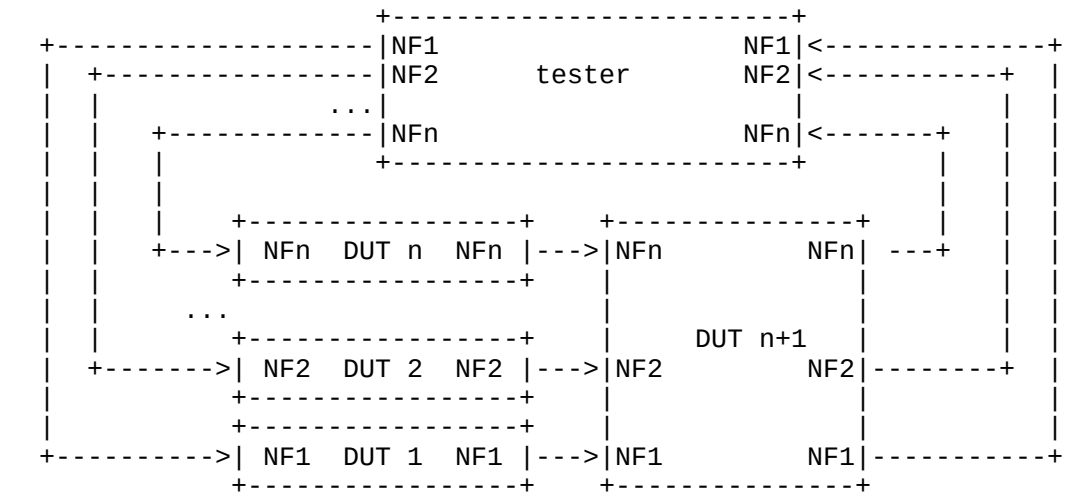


Figure 4. Test setup 4

This test setup can help to quantify the scalability of the server device. However, for testing the overload scalability of the client DUTs additional recommendations are needed.
For encapsulation transition technologies a m:n setup can be created, where m is the number of flows applied to the same client device and n the number of client devices connected to the same server device.
For the translation based transition technologies the client devices can be separately tested with n network flows using the test setup presented in Figure 3.

10.2. Benchmarking Performance Degradation

10.2.1. Network performance degradation with simultaneous load

Objective: To quantify the performance degradation introduced by n parallel and simultaneous network flows.

Procedure: First, the benchmarking tests presented in Section 6 have to be performed for one network flow.

The same tests have to be repeated for n network flows, where the network flows are started simultaneously. The performance degradation of the X benchmarking dimension SHOULD be calculated as relative performance change between the 1-flow (single flow) results and the n-flow results, using the following formula:

$$Xpd = \frac{Xn - X1}{X1} * 100, \text{ where: } X1 \text{ - result for 1-flow}, \; Xn \text{ - result for n-flows}$$

As a guideline for the maximum number of flows n, the value can be deduced by measuring the Concurrent TCP Connection Capacity as described by [RFC3511], following the test setups specified by Section 4.

Reporting Format: The performance degradation SHOULD be expressed as a percentage. The number of tested parallel flows n MUST be clearly specified. For each of the performed benchmarking tests, there SHOULD be a table containing a column for each frame size. The table SHOULD also state the applied frame rate.

10.2.2. Network performance degradation with incremental load

Objective: To quantify the performance degradation introduced by n parallel and incrementally started network flows.

Procedure: First, the benchmarking tests presented in Section 6 have to be performed for one network flow.

The same tests have to be repeated for n network flows, where the network flows are started incrementally in succession, each after time T. In other words, if flow I is started at time x, flow i+1 will be started at time x+T. Considering the time T, the time duration of each iteration must be extended with the time necessary to start all the flows, namely (n-1)xT. The measurement for the first flow SHOULD be at least 60 seconds in duration.

The performance degradation of the X benchmarking dimension SHOULD be calculated as relative performance change between the 1-flow results and the n-flow results, using the following formula presented in Section 9.2.1. Intermediary degradation points for 1/4*n, 1/2*n and 3/4*n MAY also be performed.

Reporting Format: The performance degradation SHOULD be expressed as a percentage. The number of tested parallel flows n MUST be clearly specified. For each of the performed benchmarking tests, there SHOULD be a table containing a column for each frame size. The table SHOULD also state the applied frame rate and time duration T, used as increment step between the network flows. The units of measurement for T SHOULD be seconds. A column for the intermediary degradation points MAY also be displayed.

11. NAT44 and NAT66

Although these technologies are not the primarily scope of this document, the benchmarking methodology associated with single translation technologies as defined in Section 4.1 can be employed to benchmark NAT44 (as defined by [RFC2663] with the behavior described by [RFC7857]) implementations and NAT66 (as defined by [RFC6296]) implementations.

12. Summarizing function and variation

To ensure the stability of the benchmarking scores obtained using the tests presented in Sections 6 through 9, multiple test iterations are RECOMMENDED. Using a summarizing function (or measure of central tendency) can be a simple and effective way to compare the results obtained across different iterations. However, over-summarization is an unwanted effect of reporting a single number.

Measuring the variation (dispersion index) can be used to counter the over-summarization effect. Empirical data obtained following the proposed methodology can also offer insights on which summarizing function would fit better.

To that end, data presented in [ietf95pres] indicate the median as suitable summarizing function and the 1st and 99th percentiles as variation measures for DNS Resolution Performance and PDV. The median and percentile calculation functions SHOULD follow the recommendations of [RFC2330] Section 11.3.

For a fine grain analysis of the frequency distribution of the data, histograms or cumulative distribution function plots can be employed.

13. 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 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.

14. IANA Considerations

The IANA has allocated the prefix 2001:2::/48 [RFC5180] for IPv6 benchmarking. For IPv4 benchmarking, the 198.18.0.0/15 prefix was reserved, as described in [RFC6890]. The two ranges are sufficient for benchmarking IPv6 transition technologies. Thus, no action is requested.

15. References

15.1. Normative References

[RFC1242] Bradner, S., "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, July 1991.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

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

[RFC2544] Bradner, S., and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.

[RFC2647] Newman, D., "Benchmarking Terminology for Firewall Devices", [RFC2647], August 1999.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, November 2002.

[RFC3511] Hickman, B., Newman, D., Tadjudin, S. and T. Martin, "Benchmarking Methodology for Firewall Performance", RFC 3511, April 2003.

[RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, May 2008.

[RFC5481] Morton, A., and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, March 2009.

[RFC6201] Asati, R., Pignataro, C., Calabria, F. and C. Olvera, "Device Reset Characterization ", RFC 6201, March 2011.

15.2. Informative References

[RFC2663] Srisuresh, P., and M. Holdrege. "IP Network Address Translator (NAT) Terminology and Considerations", RFC2663, August 1999.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

[RFC5569] Despres, R., "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)", RFC 5569, DOI 10.17487/RFC5569, January 2010, <http://www.rfc-editor.org/info/rfc5569>.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <http://www.rfc-editor.org/info/rfc6146>.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and Beijnum, I., "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, DOI 10.17487/RFC6147, April 2011, <http://www.rfc-editor.org/info/rfc6147>.

[RFC6219] Li, X., Bao, C., Chen, M., Zhang, H., and J. Wu, "The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition", RFC 6219, DOI 10.17487/RFC6219, May 2011, <http://www.rfc-editor.org/info/rfc6219>.

[RFC6296] Wasserman, M., and F. Baker. "IPv6-to-IPv6 network prefix translation." RFC6296, June 2011.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <http://www.rfc-editor.org/info/rfc6877>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC6890, April 2013.

[RFC7596] Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the Dual-Stack Lite Architecture", RFC 7596, DOI 10.17487/RFC7596, July 2015, <http://www.rfc-editor.org/info/rfc7596>.

[RFC7597] Troan, O., Ed., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, Ed., "Mapping of Address and Port with Encapsulation (MAP-E)", RFC 7597, DOI 10.17487/RFC7597, July 2015, <http://www.rfc-editor.org/info/rfc7597>.

[RFC7599] Li, X., Bao, C., Dec, W., Ed., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", RFC 7599, DOI 10.17487/RFC7599, July 2015, <http://www.rfc-editor.org/info/rfc7599>.

[RFC7857] Penno, R., Perreault, S., Boucadair, M., Sivakumar, S., and K. Naito "Updates to Network Address Translation (NAT) Behavioral Requirements" RFC 7857, April 2016.

[RFC7915] LBao, C., Li, X., Baker, F., Anderson, T., and F. Gont, "IP/ICMP Translation Algorithm", RFC 7915, DOI 10.17487/RFC7915, June 2016, <http://www.rfc-editor.org/info/rfc7915>.

[Dns64perf] Bakai, D., "A C++11 DNS64 performance tester", available: https://github.com/bakaid/dns64perfpp

[ietf95pres] Georgescu, M., "Benchmarking Methodology for IPv6 Transition Technologies", IETF 95, Buenos Aires, Argentina, April 2016, available: https://www.ietf.org/proceedings/95/slides/slides-95-bmwg-2.pdf

[Lencse] Lencse, G., Bakai, D, "Design and Implementation of a Test Program for Benchmarking DNS64 Servers", IEICE Transactions on Communications, conditionally accepted, revised version available: http://www.hit.bme.hu/~lencse/publications/IEICE-2016-dns64perfpp-revised.pdf

16. Acknowledgements

The authors would like to thank Youki Kadobayashi and Hiroaki Hazeyama for their constant feedback and support. The thanks should be extended to the NECOMA project members for their continuous support. The thank you list should also include Emanuel Popa and the RCS&RDS Backbone Team for their support and insights. We would also like to thank Scott Bradner for the useful suggestions. We also note that portions of text from Scott's documents were used in this memo (e.g. Latency section). A big thank you to Al Morton and Fred Baker for their detailed review of the draft and very helpful suggestions. Other helpful comments and suggestions were offered by Bhuvaneswaran Vengainathan, Andrew McGregor, Nalini Elkins, Kaname Nishizuka, Yasuhiro Ohara, Masataka Mawatari, Kostas Pentikousis, Bela Almasi, Tim Chown, Paul Emmerich and Stenio Fernandes. A special thank you to the RFC Editor Team for their thorough editorial review and helpful suggestions. This document was prepared using 2-Word-v2.0.template.dot.

Appendix A. Theoretical Maximum Frame Rates

This appendix describes the recommended calculation formulas for the theoretical maximum frame rates to be employed over Ethernet as example media. The formula takes into account the frame size overhead created by the encapsulation or the translation process. For example, the 6in4 encapsulation described in [RFC4213] adds 20 bytes of overhead to each frame.

Considering X to be the frame size and O to be the frame size overhead created by the encapsulation on translation process, the maximum theoretical frame rate for Ethernet can be calculated using the following formula:

```
        Line Rate (bps)
------------------------------
(8bits/byte)*(X+O+20)bytes/frame
```

The calculation is based on the formula recommended by RFC5180 in Appendix A1. As an example, the frame rate recommended for testing a 6in4 implementation over 10Mb/s Ethernet with 64 bytes frames is:

```
        10,000,000(bps)
------------------------------      = 12,019 fps
(8bits/byte)*(64+20+20)bytes/frame
```

The complete list of recommended frame rates for 6in4 encapsulation can be found in the following table:

Frame size (bytes)	10 Mb/s (fps)	100 Mb/s (fps)	1000 Mb/s (fps)	10000 Mb/s (fps)
64	12,019	120,192	1,201,923	12,019,231
128	7,440	74,405	744,048	7,440,476
256	4,223	42,230	422,297	4,222,973
512	2,264	22,645	226,449	2,264,493
1024	1,175	11,748	117,481	1,174,812
1280	947	9,470	94,697	946,970
1518	802	8,023	80,231	802,311
1522	800	8,003	80,026	800,256
2048	599	5,987	59,866	598,659
4096	302	3,022	30,222	302,224
8192	152	1,518	15,185	151,846
9216	135	1,350	13,505	135,048

Authors' Addresses

Marius Georgescu
RCS&RDS
Strada Dr. Nicolae D. Staicovici 71-75
Bucharest 030167
Romania

Phone: +40 31 005 0979
Email: marius.georgescu@rcs-rds.ro

Liviu Pislaru
RCS&RDS
Strada Dr. Nicolae D. Staicovici 71-75
Bucharest 030167
Romania

Phone: +40 31 005 0979
Email: liviu.pislaru@rcs-rds.ro

Gabor Lencse
Szechenyi Istvan University
Egyetem ter 1.
Gyor
Hungary

Phone: +36 20 775 8267
Email: lencse@sze.hu

Benchmarking Working Group
Internet Draft
Intended status: Informational
Expires: December 2017

M. Georgescu
L. Pislaru
RCS&RDS
G. Lencse
Szechenyi Istvan University
June 12, 2017

Benchmarking Methodology for IPv6 Transition Technologies
draft-ietf-bmwg-ipv6-tran-tech-benchmarking-08.txt

Abstract

There are benchmarking methodologies addressing the performance of network interconnect devices that are IPv4- or IPv6-capable, but the IPv6 transition technologies are outside of their scope. This document provides complementary guidelines for evaluating the performance of IPv6 transition technologies. More specifically, this document targets IPv6 transition technologies that employ encapsulation or translation mechanisms, as dual-stack nodes can be very well tested using the recommendations of RFC2544 and RFC5180. The methodology also includes a metric for benchmarking load scalability.

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 December 12, 2016.

Copyright Notice

Table of Contents

1. Introduction

The methodologies described in [RFC2544] and [RFC5180] help vendors and network operators alike analyze the performance of IPv4 and IPv6-capable network devices. The methodology presented in [RFC2544] is mostly IP version independent, while [RFC5180] contains complementary recommendations, which are specific to the latest IP version, IPv6. However, [RFC5180] does not cover IPv6 transition technologies.

IPv6 is not backwards compatible, which means that IPv4-only nodes cannot directly communicate with IPv6-only nodes. To solve this issue, IPv6 transition technologies have been proposed and implemented.

This document presents benchmarking guidelines dedicated to IPv6 transition technologies. The benchmarking tests can provide insights about the performance of these technologies, which can act as useful feedback for developers, as well as for network operators going through the IPv6 transition process.

The document also includes an approach to quantify performance when operating in overload. Overload scalability can be defined as a system's ability to gracefully accommodate greater numbers of flows than the maximum number of flows which the Device under test (DUT) can operate normally. The approach taken here is to quantify the overload scalability by measuring the performance created by an excessive number of network flows, and comparing performance to the non-overloaded case.

1.1. IPv6 Transition Technologies

Two of the basic transition technologies, dual IP layer (also known as dual stack) and encapsulation are presented in [RFC4213]. IPv4/IPv6 Translation is presented in [RFC6144]. Most of the transition technologies employ at least one variation of these mechanisms. In this context, a generic classification of the transition technologies can prove useful.

We can consider a production network transitioning to IPv6 as being constructed using the following IP domains:

- Domain A: IPvX specific domain
- Core domain: which may be IPvY specific or dual-stack(IPvX and IPvY)
- Domain B: IPvX specific domain

Note: X,Y are part of the set {4,6}, and X NOT.EQUAL Y.

According to the technology used for the core domain traversal the transition technologies can be categorized as follows:

1. Dual-stack: the core domain devices implement both IP protocols.
2. Single Translation: In this case, the production network is assumed to have only two domains, Domain A and the Core domain. The core domain is assumed to be IPvY specific. IPvX packets are translated to IPvY at the edge between Domain A and the Core domain.
3. Double translation: The production network is assumed to have all three domains; Domains A and B are IPvX specific, while the core domain is IPvY specific. A translation mechanism is employed for the traversal of the core network. The IPvX packets are translated to IPvY packets at the edge between Domain A and the Core domain. Subsequently, the IPvY packets are translated back to IPvX at the edge between the Core domain and Domain B.

4. Encapsulation: The production network is assumed to have all three domains; Domains A and B are IPvX specific, while the core domain is IPvY specific. An encapsulation mechanism is used to traverse the core domain. The IPvX packets are encapsulated to IPvY packets at the edge between Domain A and the Core domain. Subsequently, the IPvY packets are de-encapsulated at the edge between the Core domain and Domain B.

The performance of Dual-stack transition technologies can be fully evaluated using the benchmarking methodologies presented by [RFC2544] and [RFC5180]. Consequently, this document focuses on the other 3 categories: Single translation, Encapsulation and Double translation transition technologies.

Another important aspect by which the IPv6 transition technologies can be categorized is their use of stateful or stateless mapping algorithms. The technologies that use stateful mapping algorithms (e.g. Stateful NAT64 [RFC6146]) create dynamic correlations between IP addresses or {IP address, transport protocol, transport port number} tuples, which are stored in a state table. For ease of reference, the IPv6 transition technologies which employ stateful mapping algorithms will be called stateful IPv6 transition technologies. The efficiency with which the state table is managed can be an important performance indicator for these technologies. Hence, for the stateful IPv6 transition technologies additional benchmarking tests are RECOMMENDED.

Table 1 contains the generic categories as well as associations with some of the IPv6 transition technologies proposed in the IETF. Please note that the list is not exhaustive.

Table 1. IPv6 Transition Technologies Categories

	Generic category	IPv6 Transition Technology
1	Dual-stack	Dual IP Layer Operations [RFC4213]
2	Single translation	NAT64 [RFC6146], IVI [RFC6219]
3	Double translation	464XLAT [RFC6877], MAP-T [RFC7599]
4	Encapsulation	DSLite[RFC6333], MAP-E [RFC7597]
		Lightweight 4over6 [RFC7596]
		6RD [RFC5569], 6PE [RFC4798], 6VPE
		6VPE [RFC4659]

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 [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 [RFC2119] significance.

Although these terms are usually associated with protocol requirements, in this document the terms are requirements for users and systems that intend to implement the test conditions and claim conformance with this specification.

3. Terminology

A number of terms used in this memo have been defined in other RFCs. Please refer to those RFCs for definitions, testing procedures and reporting formats.

Throughput (Benchmark) - [RFC2544]

Frame Loss Rate (Benchmark) - [RFC2544]

Back-to-back Frames (Benchmark) - [RFC2544]

System Recovery (Benchmark) - [RFC2544]

Reset (Benchmark) - [RFC6201]

Concurrent TCP Connection Capacity (Benchmark) - [RFC3511]

Maximum TCP Connection Establishment Rate (Benchmark) - [RFC3511]

4. Test Setup

The test environment setup options recommended for IPv6 transition technologies benchmarking are very similar to the ones presented in Section 6 of [RFC2544]. In the case of the tester setup, the options presented in [RFC2544] and [RFC5180] can be applied here as well. However, the Device under test (DUT) setup options should be explained in the context of the targeted categories of IPv6 transition technologies: Single translation, Double translation and Encapsulation transition technologies.

Although both single tester and sender/receiver setups are applicable to this methodology, the single tester setup will be used to describe the DUT setup options.

For the test setups presented in this memo, dynamic routing SHOULD be employed. However, the presence of routing and management frames can represent unwanted background data that can affect the benchmarking result. To that end, the procedures defined in [RFC2544] (Sections 11.2 and 11.3) related to routing and management frames SHOULD be used here. Moreover, the "Trial description" recommendations presented in [RFC2544] (Section 23) are also valid for this memo.

In terms of route setup, the recommendations of [RFC2544] Section 13 are valid for this document assuming that IPv6 capable routing protocols are used..

4.1. Single translation Transition Technologies

For the evaluation of Single translation transition technologies, a single DUT setup (see Figure 1) SHOULD be used. The DUT is responsible for translating the IPvX packets into IPvY packets. In this context, the tester device SHOULD be configured to support both IPvX and IPvY.

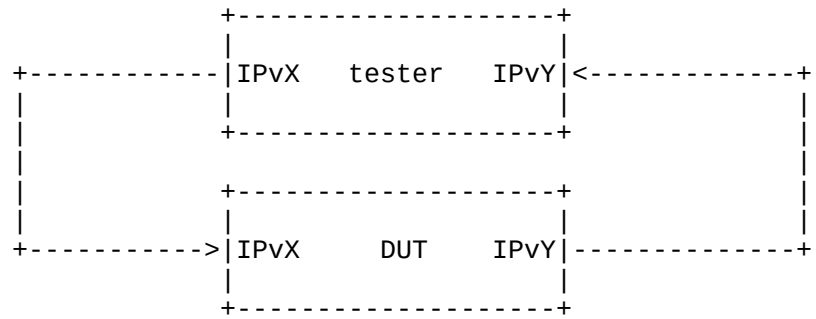


Figure 1. Test setup 1

4.2. Encapsulation/Double translation Transition Technologies

For evaluating the performance of Encapsulation and Double translation transition technologies, a dual DUT setup (see Figure 2) SHOULD be employed. The tester creates a network flow of IPvX packets. The first DUT is responsible for the encapsulation or translation of IPvX packets into IPvY packets. The IPvY packets are de-encapsulated/translated back to IPvX packets by the second DUT and forwarded to the tester.

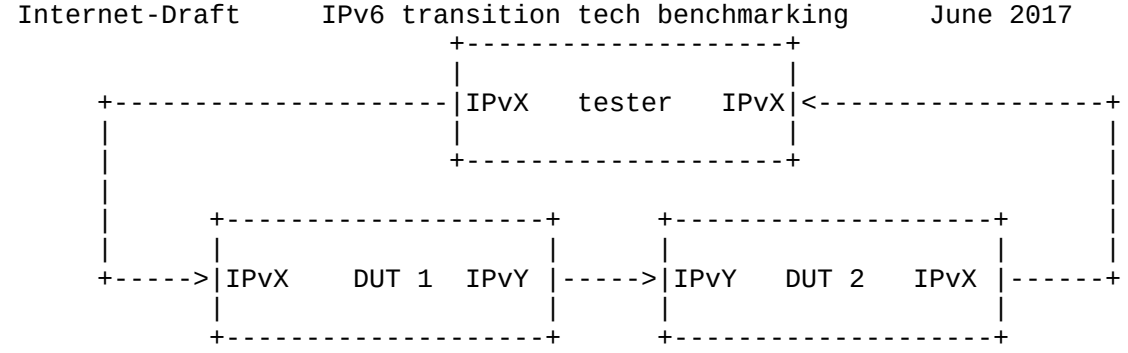


Figure 2. Test setup 2

One of the limitations of the dual DUT setup is the inability to reflect asymmetries in behavior between the DUTs. Considering this, additional performance tests SHOULD be performed using the single DUT setup.

Note: For encapsulation IPv6 transition technologies, in the single DUT setup, in order to test the de-encapsulation efficiency, the tester SHOULD be able to send IPvX packets encasulated as IPvY.

5. Test Traffic

The test traffic represents the experimental workload and SHOULD meet the requirements specified in this section. The requirements are dedicated to unicast IP traffic. Multicast IP traffic is outside of the scope of this document.

5.1. Frame Formats and Sizes

[RFC5180] describes the frame size requirements for two commonly used media types: Ethernet and SONET (Synchronous Optical Network). [RFC2544] covers also other media types, such as token ring and FDDI. The recommendations of the two documents can be used for the dual-stack transition technologies. For the rest of the transition technologies, the frame overhead introduced by translation or encapsulation MUST be considered.

The encapsulation/translation process generates different size frames on different segments of the test setup. For instance, the single translation transition technologies will create different frame sizes on the receiving segment of the test setup, as IPvX packets are translated to IPvY. This is not a problem if the bandwidth of the employed media is not exceeded. To prevent exceeding the limitations imposed by the media, the frame size overhead needs to be taken into account when calculating the maximum theoretical frame rates. The calculation method for the Ethernet, as

well as a calculation example, are detailed in Appendix A. The details of the media employed for the benchmarking tests MUST be noted in all test reports.

In the context of frame size overhead, MTU recommendations are needed in order to avoid frame loss due to MTU mismatch between the virtual encapsulation/translation interfaces and the physical network interface controllers (NICs). To avoid this situation, the larger MTU between the physical NICs and virtual encapsulation/translation interfaces SHOULD be set for all interfaces of the DUT and tester. To be more specific, the minimum IPv6 MTU size (1280 bytes) plus the encapsulation/translation overhead is the RECOMMENDED value for the physical interfaces as well as virtual ones.

5.1.1. Frame Sizes to Be Used over Ethernet

Based on the recommendations of [RFC5180], the following frame sizes SHOULD be used for benchmarking IPvX/IPvY traffic on Ethernet links: 64, 128, 256, 512, 768, 1024, 1280, 1518, 1522, 2048, 4096, 8192 and 9216.

For Ethernet frames exceeding 1500 bytes in size, the [IEEE802.1AC] standard can be consulted.

Note: for single translation transition technologies (e.g. NAT64) in the IPv6 -> IPv4 translation direction, 64 byte frames SHOULD be replaced by 84 byte frames. This would allow the frames to be transported over media such as the ones described by the IEEE 802.1Q standard. Moreover, this would also allow the implementation of a frame identifier in the UDP data.

The theoretical maximum frame rates considering an example of frame overhead are presented in Appendix A.

5.2. Protocol Addresses

The selected protocol addresses should follow the recommendations of [RFC5180](Section 5) for IPv6 and [RFC2544](Section 12) for IPv4.

Note: testing traffic with extension headers might not be possible for the transition technologies, which employ translation. Proposed IPvX/IPvY translation algorithms such as IP/ICMP translation [RFC7915] do not support the use of extension headers.

5.3. Traffic Setup

Following the recommendations of [RFC5180], all tests described SHOULD be performed with bi-directional traffic. Uni-directional

traffic tests MAY also be performed for a fine grained performance assessment.

Because of the simplicity of UDP, UDP measurements offer a more reliable basis for comparison than other transport layer protocols. Consequently, for the benchmarking tests described in Section 7 of this document UDP traffic SHOULD be employed.

Considering that a transition technology could process both native IPv6 traffic and translated/encapsulated traffic, the following traffic setups are recommended:

i) IPvX only traffic (where the IPvX traffic is to be translated/encapsulated by the DUT)
ii) 90% IPvX traffic and 10% IPvY native traffic
iii) 50% IPvX traffic and 50% IPvY native traffic
iv) 10% IPvX traffic and 90% IPvY native traffic

For the benchmarks dedicated to stateful IPv6 transition technologies, included in Section 8 of this memo (Concurrent TCP Connection Capacity and Maximum TCP Connection Establishment Rate), the traffic SHOULD follow the recommendations of [RFC3511], Sections 5.2.2.2 and 5.3.2.2.

6. Modifiers

The idea of testing under different operational conditions was first introduced in [RFC2544](Section 11) and represents an important aspect of benchmarking network elements, as it emulates, to some extent, the conditions of a production environment. Section 6 of [RFC5180] describes complementary testing conditions specific to IPv6. Their recommendations can also be followed for IPv6 transition technologies testing.

7. Benchmarking Tests

The following sub-sections contain the list of all recommended benchmarking tests.

7.1. Throughput

Use Section 26.1 of RFC2544 unmodified.

7.2. Latency

Objective: To determine the latency. Typical latency is based on the definitions of latency from [RFC1242]. However, this memo provides a new measurement procedure.

Procedure: Similar to [RFC2544], the throughput for DUT at each of the listed frame sizes SHOULD be determined. Send a stream of frames at a particular frame size through the DUT at the determined throughput rate to a specific destination. The stream SHOULD be at least 120 seconds in duration.

Identifying tags SHOULD be included in at least 500 frames after 60 seconds. For each tagged frame, the time at which the frame was fully transmitted (timestamp A) and the time at which the frame was received (timestamp B) MUST be recorded. The latency is timestamp B minus timestamp A as per the relevant definition from RFC 1242, namely latency as defined for store and forward devices or latency as defined for bit forwarding devices.

We recommend to encode the identifying tag in the payload of the frame. To be more exact, the identifier SHOULD be inserted after the UDP header.

From the resulted (at least 500) latencies, 2 quantities SHOULD be calculated. One is the typical latency, which SHOULD be calculated with the following formula:

TL=Median(Li)

Where: TL - the reported typical latency of the stream

Li -the latency for tagged frame i

The other measure is the worst case latency, which SHOULD be calculated with the following formula:

WCL=L99.9thPercentile

Where: WCL - The reported worst case latency

L99.9thPercentile - The 99.9th Percentile of the stream measured latencies

The test MUST be repeated at least 20 times with the reported value being the median of the recorded values for TL and WCL.

Reporting Format: The report MUST state which definition of latency (from RFC 1242) was used for this test. The summarized latency 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, the rate at which the latency test was run for that frame size, for the media types tested, and for the resultant typical latency and worst case latency values for each type of data stream tested. To account for the variation, the 1st and 99th percentiles of the 20 iterations MAY be reported in two separated columns. For a fine grained analysis, the histogram (as exemplified in [RFC5481] Section 4.4) of one of the iterations MAY be displayed .

7.3. Packet Delay Variation

Considering two of the metrics presented in [RFC5481], Packet Delay Variation (PDV) and Inter Packet Delay Variation (IPDV), it is RECOMMENDED to measure PDV. For a fine grained analysis of delay variation, IPDV measurements MAY be performed.

7.3.1. PDV

Objective: To determine the Packet Delay Variation as defined in [RFC5481].

Procedure: As described by [RFC2544], first determine the throughput for the DUT at each of the listed frame sizes. Send a stream of frames at a particular frame size through the DUT at the determined throughput rate to a specific destination. The stream SHOULD be at least 60 seconds in duration. Measure the One-way delay as described by [RFC3393] for all frames in the stream. Calculate the PDV of the stream using the formula:

PDV=D99.9thPercentile - Dmin

Where: D99.9thPercentile - the 99.9th Percentile (as it was described in [RFC5481]) of the One-way delay for the stream

Dmin - the minimum One-way delay in the stream

As recommended in [RFC2544], the test MUST be repeated at least 20 times with the reported value being the median of the recorded values. Moreover, the 1st and 99th percentiles SHOULD be calculated to account for the variation of the dataset.

Reporting Format: The PDV results SHOULD be reported in a table with a row for each of the tested frame sizes and columns for the frame size and the applied frame rate for the tested media types. Two columns for the 1st and 99th percentile values MAY be displayed. Following the recommendations of [RFC5481], the RECOMMENDED units of measurement are milliseconds.

7.3.2. IPDV

Objective: To determine the Inter Packet Delay Variation as defined in [RFC5481].

Procedure: As described by [RFC2544], first determine the throughput for the DUT at each of the listed frame sizes. Send a stream of frames at a particular frame size through the DUT at the determined throughput rate to a specific destination. The stream SHOULD be at least 60 seconds in duration. Measure the One-way delay as described by [RFC3393] for all frames in the stream. Calculate the IPDV for each of the frames using the formula:

IPDV(i)=D(i) - D(i-1)

Where: D(i) - the One-way delay of the i th frame in the stream

D(i-1) - the One-way delay of i-1 th frame in the stream

Given the nature of IPDV, reporting a single number might lead to over-summarization. In this context, the report for each measurement SHOULD include 3 values: Dmin, Dmed, and Dmax

Where: Dmin - the minimum IPDV in the stream

Dmed - the median IPDV of the stream

Dmax - the maximum IPDV in the stream

The test MUST be repeated at least 20 times. To summarize the 20 repetitions, for each of the 3 (Dmin, Dmed and Dmax) the median value SHOULD be reported.

Reporting format: The median for the 3 proposed values SHOULD be reported. The IPDV results SHOULD be reported in a table with a row for each of the tested frame sizes. The columns SHOULD include the frame size and associated frame rate for the tested media types and sub-columns for the three proposed reported values. Following the recommendations of [RFC5481], the RECOMMENDED units of measurement are milliseconds.

7.4. Frame Loss Rate

Use Section 26.3 of [RFC2544] unmodified.

7.5. Back-to-back Frames

Use Section 26.4 of [RFC2544] unmodified.

7.6. System Recovery

Use Section 26.5 of [RFC2544] unmodified.

7.7. Reset

Use Section 4 of [RFC6201] unmodified.

8. Additional Benchmarking Tests for Stateful IPv6 Transition Technologies

This section describes additional tests dedicated to the stateful IPv6 transition technologies. For the tests described in this section, the DUT devices SHOULD follow the test setup and test parameters recommendations presented in [RFC3511] (Sections 5.2 and 5.3)

The following additional tests SHOULD be performed.

8.1. Concurrent TCP Connection Capacity

Use Section 5.2 of [RFC3511] unmodified.

8.2. Maximum TCP Connection Establishment Rate

Use Section 5.3 of RFC3511 unmodified.

9. DNS Resolution Performance

This section describes benchmarking tests dedicated to DNS64 (see [RFC6147]), used as DNS support for single translation technologies such as NAT64.

9.1. Test and Traffic Setup

The test setup in Figure 3 follows the setup proposed for single translation IPv6 transition technologies in Figure 1.

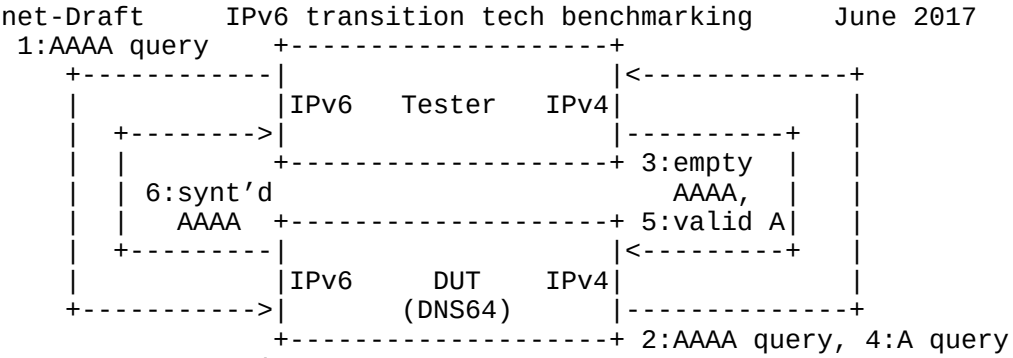


Figure 3. DNS64 test setup

The test traffic SHOULD follow the following steps.

1. Query for the AAAA record of a domain name (from client to DNS64 server)

2. Query for the AAAA record of the same domain name (from DNS64 server to authoritative DNS server)

3. Empty AAAA record answer (from authoritative DNS server to DNS64 server)

4. Query for the A record of the same domain name (from DNS64 server to authoritative DNS server)

5. Valid A record answer (from authoritative DNS server to DNS64 server)

6. Synthesized AAAA record answer (from DNS64 server to client)

The Tester plays the role of DNS client as well as authoritative DNS server. It MAY be realized as a single physical device, or alternatively, two physical devices MAY be used.

Please note that:

- If the DNS64 server implements caching and there is a cache hit, then step 1 is followed by step 6 (and steps 2 through 5 are omitted).
- If the domain name has an AAAA record, then it is returned in step 3 by the authoritative DNS server; steps 4 and 5 are omitted, and the DNS64 server does not synthesizes an AAAA record, but returns the received AAAA record to the client.

- As for the IP version used between the tester and the DUT, IPv6 MUST be used between the client and the DNS64 server (as a DNS64 server provides service for an IPv6-only client), but either IPv4 or IPv6 MAY be used between the DNS64 server and the authoritative DNS server.

9.2. Benchmarking DNS Resolution Performance

Objective: To determine DNS64 performance by means of the maximum number of successfully processed DNS requests per second.

Procedure: Send a specific number of DNS queries at a specific rate to the DUT and then count the replies received in time (within a predefined timeout period from the sending time of the corresponding query, having the default value 1 second) and valid (contains an AAAA record) from the DUT. If the count of sent queries is equal to the count of received replies, the rate of the queries is raised and the test is rerun. If fewer replies are received than queries were sent, the rate of the queries is reduced and the test is rerun. The duration of each trial SHOULD be at least 60 seconds. This will reduce the potential gain of a DNS64 server, which is able to exhibit higher performance by storing the requests and thus utilizing also the timeout time for answering them. For the same reason, no higher timeout time than 1 second SHOULD be used. For further considerations, see [Lencse1].

The maximum number of processed DNS queries per second is the fastest rate at which the count of DNS replies sent by the DUT is equal to the number of DNS queries sent to it by the test equipment.

The test SHOULD be repeated at least 20 times and the median and 1st /99th percentiles of the number of processed DNS queries per second SHOULD be calculated.

Details and parameters:

1. Caching
First, all the DNS queries MUST contain different domain names (or domain names MUST NOT be repeated before the cache of the DUT is exhausted). Then new tests MAY be executed with domain names, 20%, 40%, 60%, 80% and 100% of which are cached. We note that ensuring a record being cached requires repeating it both "late enough" after the first query to be already resolved and be present in the cache and "early enough" to be still present in the cache.

2. Existence of AAAA record
First, all the DNS queries MUST contain domain names which do not have an AAAA record and have exactly one A record.

Then new tests MAY be executed with domain names, 20%, 40%, 60%, 80% and 100% of which have an AAAA record.

Please note that the two conditions above are orthogonal, thus all their combinations are possible and MAY be tested. The testing with 0% cached domain names and with 0% existing AAAA record is REQUIRED and the other combinations are OPTIONAL. (When all the domain names are cached, then the results do not depend on what percentage of the domain names have AAAA records, thus these combinations are not worth testing one by one.)

Reporting format: The primary result of the DNS64 test is the median of the number of processed DNS queries per second measured with the above mentioned "0% + 0% combination". The median SHOULD be complemented with the 1st and 99th percentiles to show the stability of the result. If optional tests are done, the median and the 1st and 99th percentiles MAY be presented in a two dimensional table where the dimensions are the proportion of the repeated domain names and the proportion of the DNS names having AAAA records. The two table headings SHOULD contain these percentage values. Alternatively, the results MAY be presented as the corresponding two dimensional graph, too. In this case the graph SHOULD show the median values with the percentiles as error bars. From both the table and the graph, one dimensional excerpts MAY be made at any given fixed percentage value of the other dimension. In this case, the fixed value MUST be given together with a one dimensional table or graph.

9.2.1. Requirements for the Tester

Before a Tester can be used for testing a DUT at rate r queries per second with t seconds timeout, it MUST perform a self-test in order to exclude the possibility that the poor performance of the Tester itself influences the results. For performing a self-test, the tester is looped back (leaving out DUT) and its authoritative DNS server subsystem is configured to be able to answer all the AAAA record queries. For passing the self-test, the Tester SHOULD be able to answer AAAA record queries at 2*(r+delta) rate within 0.25*t timeout, where the value of delta is at least 0.1.

Explanation: When performing DNS64 testing, each AAAA record query may result in at most two queries sent by the DUT, the first one of them is for an AAAA record and the second one is for an A record (the are both sent when there is no cache hit and also no AAAA record exists). The parameters above guarantee that the authoritative DNS server subsystem of the DUT is able to answer the queries at the required frequency using up not more than the half of the timeout time.

Remark: a sample open-source test program, dns64perf++, is available from [Dns64perf] and it is documented in [Lencse2]. It implements only the client part of the Tester and it should be used together with an authoritative DNS server implementation, e.g. BIND, NSD or YADIFA. Its experimental extension for testing caching is available from [Lencse3] and it is documented in [Lencse4].

10. Overload Scalability

Scalability has been often discussed; however, in the context of network devices, a formal definition or a measurement method has not yet been proposed. In this context, we can define overload scalability as the ability of each transition technology to accommodate network growth. Poor scalability usually leads to poor performance. Considering this, overload scalability can be measured by quantifying the network performance degradation associated with an increased number of network flows.

The following subsections describe how the test setups can be modified to create network growth and how the associated performance degradation can be quantified.

10.1. Test Setup

The test setups defined in Section 3 have to be modified to create network growth.

10.1.1. Single Translation Transition Technologies

In the case of single translation transition technologies the network growth can be generated by increasing the number of network flows generated by the tester machine (see Figure 4).

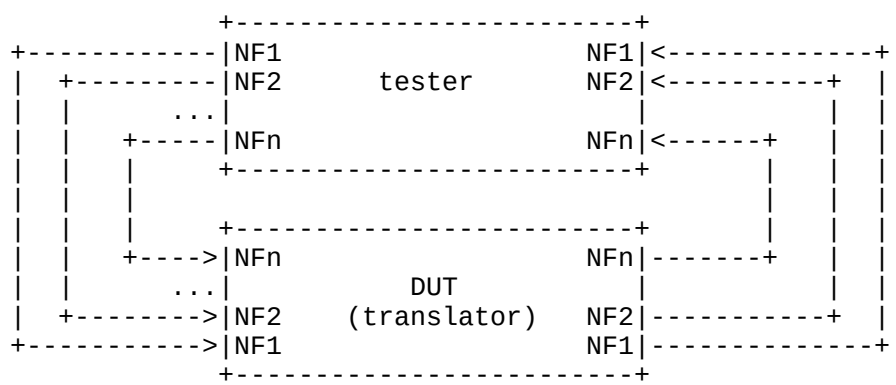


Figure 4. Test setup 3

10.1.2. Encapsulation/Double Translation Transition Technologies

Similarly, for the encapsulation/double translation technologies a multi-flow setup is recommended. Considering a multipoint-to-point scenario, for most transition technologies, one of the edge nodes is designed to support more than one connecting devices. Hence, the recommended test setup is a n:1 design, where n is the number of client DUTs connected to the same server DUT (See Figure 5).

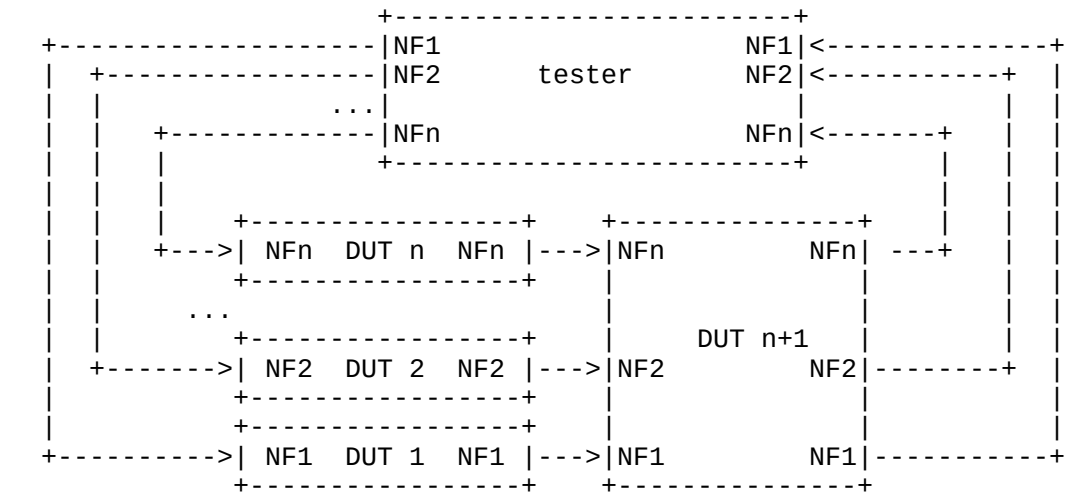


Figure 5. Test setup 4

This test setup can help to quantify the scalability of the server device. However, for testing the overload scalability of the client DUTs additional recommendations are needed.
For encapsulation transition technologies, a m:n setup can be created, where m is the number of flows applied to the same client device and n the number of client devices connected to the same server device.
For the translation based transition technologies, the client devices can be separately tested with n network flows using the test setup presented in Figure 4.

10.2. Benchmarking Performance Degradation

10.2.1. Network performance degradation with simultaneous load

Objective: To quantify the performance degradation introduced by n parallel and simultaneous network flows.

Procedure: First, the benchmarking tests presented in Section 7 have to be performed for one network flow.

The same tests have to be repeated for n network flows, where the network flows are started simultaneously. The performance degradation of the X benchmarking dimension SHOULD be calculated as relative performance change between the 1-flow (single flow) results and the n-flow results, using the following formula:

```
          Xn - X1
Xpd= ----------- * 100, where: X1 - result for 1-flow
           X1                  Xn - result for n-flows
```

This formula SHOULD be applied only for lower is better benchmarks (e.g. latency).
For higher is better benchmarks (e.g. throughput), the following formula is RECOMMENDED.

```
          X1 - Xn
Xpd= ----------- * 100, where: X1 - result for 1-flow
           X1                  Xn - result for n-flows
```

As a guideline for the maximum number of flows n, the value can be deduced by measuring the Concurrent TCP Connection Capacity as described by [RFC3511], following the test setups specified by Section 4.

Reporting Format: The performance degradation SHOULD be expressed as a percentage. The number of tested parallel flows n MUST be clearly specified. For each of the performed benchmarking tests, there SHOULD be a table containing a column for each frame size. The table SHOULD also state the applied frame rate. In the case of benchmarks for which more than one value is reported (e.g. IPDV Section 7.3.2), a column for each of the values SHOULD be included.

10.2.2. Network performance degradation with incremental load

Objective: To quantify the performance degradation introduced by n parallel and incrementally started network flows.

Procedure: First, the benchmarking tests presented in Section 7 have to be performed for one network flow.

The same tests have to be repeated for n network flows, where the network flows are started incrementally in succession, each after time t. In other words, if flow i is started at time x, flow i+1 will be started at time x+t. Considering the time t, the time duration of each iteration must be extended with the time necessary to start all the flows, namely (n-1)xt. The measurement for the first flow SHOULD be at least 60 seconds in duration.

The performance degradation of the x benchmarking dimension SHOULD be calculated as relative performance change between the 1-flow results and the n-flow results, using the formula presented in Section 10.2.1. Intermediary degradation points for 1/4*n, 1/2*n and 3/4*n MAY also be performed.

Reporting Format: The performance degradation SHOULD be expressed as a percentage. The number of tested parallel flows n MUST be clearly specified. For each of the performed benchmarking tests, there SHOULD be a table containing a column for each frame size. The table SHOULD also state the applied frame rate and time duration T, used as increment step between the network flows. The units of measurement for T SHOULD be seconds. A column for the intermediary degradation points MAY also be displayed. In the case of benchmarks for which more than one value is reported (e.g. IPDV Section 7.3.2), a column for each of the values SHOULD be included.

11. NAT44 and NAT66

Although these technologies are not the primary scope of this document, the benchmarking methodology associated with single translation technologies as defined in Section 4.1 can be employed to benchmark NAT44 (as defined by [RFC2663] with the behavior described by [RFC7857]) implementations and NAT66 (as defined by [RFC6296]) implementations.

12. Summarizing function and variation

To ensure the stability of the benchmarking scores obtained using the tests presented in Sections 7 through 9, multiple test iterations are RECOMMENDED. Using a summarizing function (or measure of central tendency) can be a simple and effective way to compare the results obtained across different iterations. However, over-summarization is an unwanted effect of reporting a single number.

Measuring the variation (dispersion index) can be used to counter the over-summarization effect. Empirical data obtained following the proposed methodology can also offer insights on which summarizing function would fit better.

To that end, data presented in [ietf95pres] indicate the median as suitable summarizing function and the 1st and 99th percentiles as variation measures for DNS Resolution Performance and PDV. The median and percentile calculation functions SHOULD follow the recommendations of [RFC2330] Section 11.3.

For a fine grained analysis of the frequency distribution of the data, histograms or cumulative distribution function plots can be employed.

13. 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 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.

14. IANA Considerations

The IANA has allocated the prefix 2001:2::/48 [RFC5180] for IPv6 benchmarking. For IPv4 benchmarking, the 198.18.0.0/15 prefix was reserved, as described in [RFC6890]. The two ranges are sufficient for benchmarking IPv6 transition technologies. Thus, no action is requested.

15. References

15.1. 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>.

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP performance metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <http://www.rfc-editor.org/info/rfc2330>.

[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>.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <http://www.rfc-editor.org/info/rfc3393>.

[RFC3511] Hickman, B., Newman, D., Tadjudin, S. and T. Martin, "Benchmarking Methodology for Firewall Performance", RFC 3511, DOI 10.17487/RFC3511, April 2003, <http://www.rfc-editor.org/info/rfc3511>.

[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, <http://www.rfc-editor.org/info/rfc5180>.

[RFC5481] Morton, A., and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <http://www.rfc-editor.org/info/rfc5481>.

[RFC6201] Asati, R., Pignataro, C., Calabria, F. and C. Olvera, "Device Reset Characterization ", RFC 6201, DOI 10.17487/RFC6201, March 2011, <http://www.rfc-editor.org/info/rfc6201>.

15.2. Informative References

[RFC2663] Srisuresh, P., and M. Holdrege. "IP Network Address Translator (NAT) Terminology and Considerations", RFC2663, DOI 10.17487/RFC2663, August 1999, <http://www.rfc-editor.org/info/rfc2663>.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, DOI 10.17487/RFC4213, October 2005, <http://www.rfc-editor.org/info/rfc4213>.

[RFC4659] De Clercq, J., Ooms, D., Carugi, M., and F. Le Faucheur, "BGP-MPLS IP Virtual Private Network (VPN) Extension for IPv6 VPN", RFC 4659, September 2006, <http://www.rfc-editor.org/info/4659>.

[RFC4798] De Clercq, J., Ooms, D., Prevost, S., and F. Le Faucheur, "Connecting IPv6 Islands over IPv4 MPLS Using IPv6 Provider Edge Routers (6PE)", RFC 4798, February 2007, <http://www.rfc-editor.org/info/rfc4798>

[RFC5569] Despres, R., "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)", RFC 5569, DOI 10.17487/RFC5569, January 2010, <http://www.rfc-editor.org/info/rfc5569>.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, DOI 10.17487/RFC6144, April 2011, <http://www.rfc-editor.org/info/rfc6144>.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <http://www.rfc-editor.org/info/rfc6146>.

[RFC6147] Bagnulo, M., Sullivan, A., Matthews, P., and I. van Beijnum, "DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers", RFC 6147, DOI 10.17487/RFC6147, April 2011, <http://www.rfc-editor.org/info/rfc6147>.

[RFC6219] Li, X., Bao, C., Chen, M., Zhang, H., and J. Wu, "The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition", RFC 6219, DOI 10.17487/RFC6219, May 2011, <http://www.rfc-editor.org/info/rfc6219>.

[RFC6296] Wasserman, M., and F. Baker. "IPv6-to-IPv6 network prefix translation." RFC6296, DOI 10.17487/RFC6296, June 2011.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, DOI 10.17487/RFC6333, August 2011, <http://www.rfc-editor.org/info/rfc6333>.

[RFC6877] Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", RFC 6877, DOI 10.17487/RFC6877, April 2013, <http://www.rfc-editor.org/info/rfc6877>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC6890, DOI 10.17487/RFC6890, April 2013, <http://www.rfc-editor.org/info/rfc6890>.

[RFC7596] Cui, Y., Sun, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the Dual-Stack Lite Architecture", RFC 7596, DOI 10.17487/RFC7596, July 2015, <http://www.rfc-editor.org/info/rfc7596>.

[RFC7597] Troan, O., Ed., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, Ed., "Mapping of Address and Port with Encapsulation (MAP-E)", RFC 7597, DOI 10.17487/RFC7597, July 2015, <http://www.rfc-editor.org/info/rfc7597>.

[RFC7599] Li, X., Bao, C., Dec, W., Ed., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", RFC 7599, DOI 10.17487/RFC7599, July 2015, <http://www.rfc-editor.org/info/rfc7599>.

[RFC7857] Penno, R., Perreault, S., Boucadair, M., Sivakumar, S., and K. Naito "Updates to Network Address Translation (NAT) Behavioral Requirements" RFC 7857, DOI 10.17487/RFC7857, April 2016, <http://www.rfc-editor.org/info/rfc7857>.

[RFC7915] LBao, C., Li, X., Baker, F., Anderson, T., and F. Gont, "IP/ICMP Translation Algorithm", RFC 7915, DOI 10.17487/RFC7915, June 2016, <http://www.rfc-editor.org/info/rfc7915>.

[Dns64perf] Bakai, D., "A C++11 DNS64 performance tester", available: https://github.com/bakaid/dns64perfpp

[ietf95pres] Georgescu, M., "Benchmarking Methodology for IPv6 Transition Technologies", IETF 95, Buenos Aires, Argentina, April 2016, available: https://www.ietf.org/proceedings/95/slides/slides-95-bmwg-2.pdf

[Lencse1] Lencse, G., Georgescu, M. and Y. Kadobayashi, "Benchmarking Methodology for DNS64 Servers", unpublished, revised version is available: http://www.hit.bme.hu/~lencse/publications/ECC-2017-B-M-DNS64-revised.pdf

[Lencse2] Lencse, G., Bakai, D, "Design and Implementation of a Test Program for Benchmarking DNS64 Servers", IEICE Transactions on Communications, vol. E100-B, no. 6. pp. 948-954, (June 2017), freely available from: http://doi.org/10.1587/transcom.2016EBN0007

[Lencse3] http://www.hit.bme.hu/~lencse/dns64perfppc/

[Lencse4] Lencse, G., "Enabling Dns64perf++ for Benchmarking the Caching Performance of DNS64 Servers", unpublished, review version is available: http://www.hit.bme.hu/~lencse/publications/IEICE-2016-dns64perfppc-for-review.pdf

[IEEE802.1AC-2016] IEEE Standard, "802.1AC-2016 - IEEE Standard for Local and metropolitan area networks -- Media Access Control (MAC) Service Definition", 2016, available: https://standards.ieee.org/findstds/standard/802.1AC-2016.html

16. Acknowledgements

The authors would like to thank Youki Kadobayashi and Hiroaki Hazeyama for their constant feedback and support. The thanks should be extended to the NECOMA project members for their continuous support. The thank you list should also include Emanuel Popa, Ionut Spirlea and the RCS&RDS IP/MPLS Backbone Team for their support and insights. We would also like to thank Scott Bradner for the useful suggestions. We also note that portions of text from Scott’s documents were used in this memo (e.g. Latency section). A big thank you to Al Morton and Fred Baker for their detailed review of the draft and very helpful suggestions. Other helpful comments and suggestions were offered by Bhuvaneswaran Vengainathan, Andrew McGregor, Nalini Elkins, Kaname Nishizuka, Yasuhiro Ohara, Masataka Mawatari, Kostas Pentikousis, Bela Almasi, Tim Chown, Paul Emmerich and Stenio Fernandes. A special thank you to the RFC Editor Team for their thorough editorial review and helpful suggestions. This document was prepared using 2-Word-v2.0.template.dot.

Appendix A. Theoretical Maximum Frame Rates

This appendix describes the recommended calculation formulas for the theoretical maximum frame rates to be employed over Ethernet as example media. The formula takes into account the frame size overhead created by the encapsulation or the translation process. For example, the 6in4 encapsulation described in [RFC4213] adds 20 bytes of overhead to each frame.

Considering X to be the frame size and O to be the frame size overhead created by the encapsulation on translation process, the maximum theoretical frame rate for Ethernet can be calculated using the following formula:

```
        Line Rate (bps)
------------------------------
(8bits/byte)*(X+O+20)bytes/frame
```

The calculation is based on the formula recommended by RFC5180 in Appendix A1. As an example, the frame rate recommended for testing a 6in4 implementation over 10Mb/s Ethernet with 64 bytes frames is:

```
        10,000,000(bps)
------------------------------      = 12,019 fps
(8bits/byte)*(64+20+20)bytes/frame
```

The complete list of recommended frame rates for 6in4 encapsulation can be found in the following table:

Frame size (bytes)	10 Mb/s (fps)	100 Mb/s (fps)	1000 Mb/s (fps)	10000 Mb/s (fps)
64	12,019	120,192	1,201,923	12,019,231
128	7,440	74,405	744,048	7,440,476
256	4,223	42,230	422,297	4,222,973
512	2,264	22,645	226,449	2,264,493
678	1,740	17,409	174,094	1,740,947
1024	1,175	11,748	117,481	1,174,812
1280	947	9,470	94,697	946,970
1518	802	8,023	80,231	802,311
1522	800	8,003	80,026	800,256
2048	599	5,987	59,866	598,659
4096	302	3,022	30,222	302,224
8192	152	1,518	15,185	151,846
9216	135	1,350	13,505	135,048

Authors' Addresses

Marius Georgescu
RCS&RDS
Strada Dr. Nicolae D. Staicovici 71-75
Bucharest 030167
Romania

Phone: +40 31 005 0979
Email: marius.georgescu@rcs-rds.ro

Liviu Pislaru
RCS&RDS
Strada Dr. Nicolae D. Staicovici 71-75
Bucharest 030167
Romania

Phone: +40 31 005 0979
Email: liviu.pislaru@rcs-rds.ro

Gabor Lencse
Szechenyi Istvan University
Egyetem ter 1.
Gyor
Hungary

Phone: +36 20 775 8267
Email: lencse@sze.hu

Internet-Draft
Network Working Group
Intended Status: Informational
Expires: January 8, 2017

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Mark Tassinari
Hewlett-Packard
Vishwas Manral
Nano Sec
Sarah Banks
VSS Monitoring
July 8, 2016

Benchmarking Methodology for SDN Controller Performance
draft-ietf-bmwg-sdn-controller-benchmark-meth-02

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.

3. 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.

3.1. Test setup - Controller working in Standalone Mode

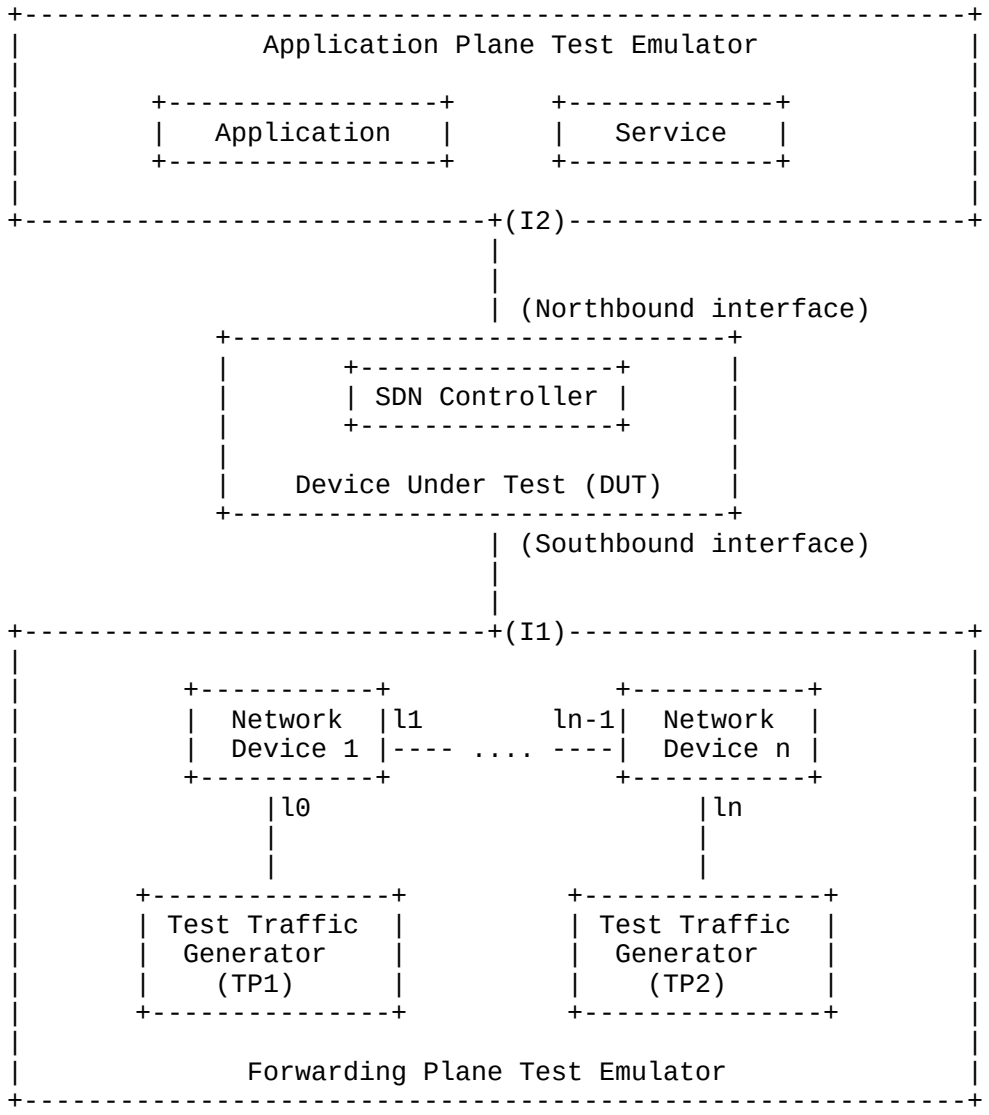


Figure 1

3.2. Test setup - Controller working in Cluster Mode

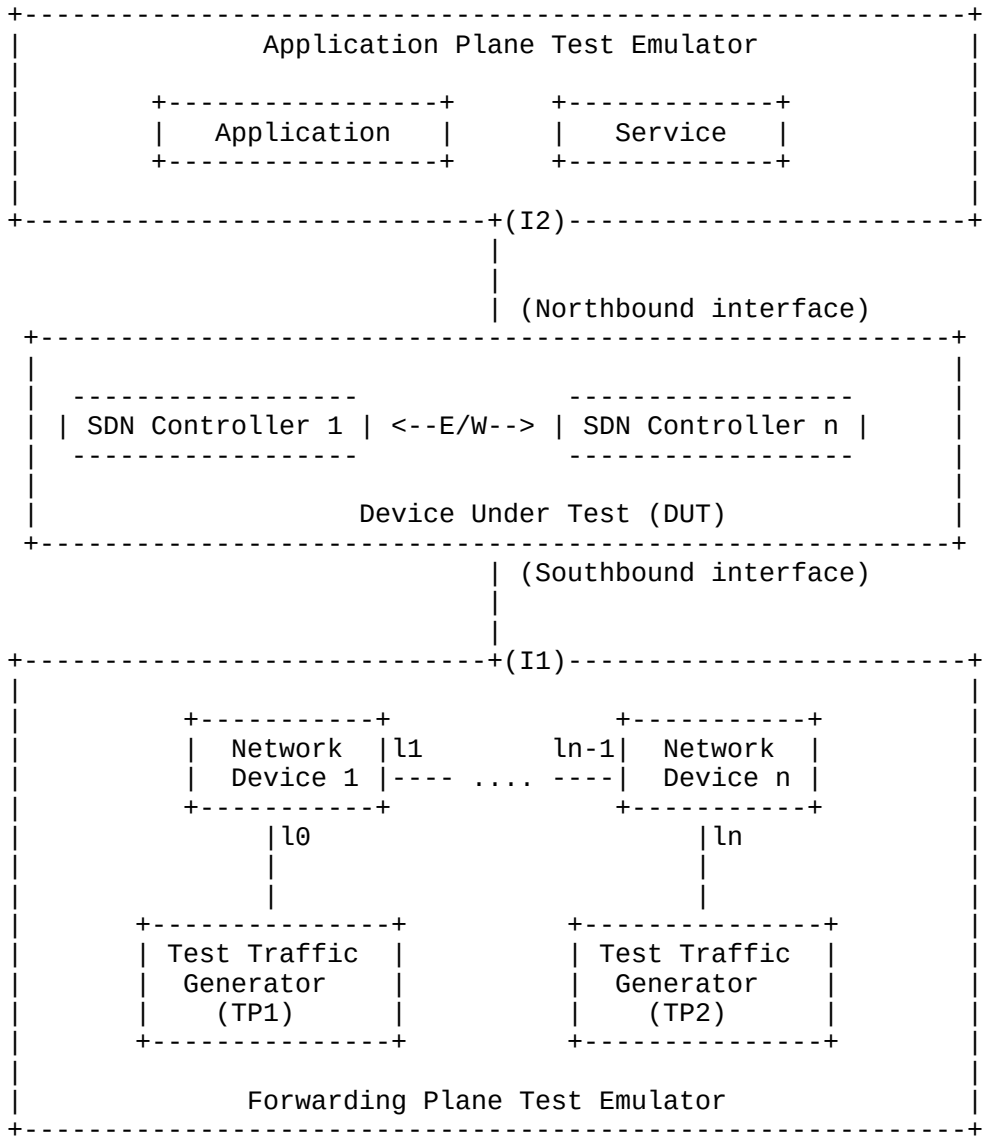


Figure 2

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 a few 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 arrival of new flows. The test cases SHOULD use multiple frame sizes as recommended in RFC2544 for benchmarking.

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 is recommended that 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.

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.

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.Test Traffic Type
11.Controller System Configuration (e.g., CPU, Memory, Operating System, Interface Speed etc.,)
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.,)

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 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 test 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 test completed successfully. (e.g., the topology matches).

Measurement:

Topology Discovery Time Tr1 = Tmn-Tm1.

$$\text{Average Topology Discovery Time} = \frac{Tr1 + Tr2 + Tr3 \ldots Trn}{\text{Total Test Iterations}}$$

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.

Prerequisite:

1. The controller MUST have 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 test 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

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

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:

The maximum number of asynchronous messages (session aliveness check message, new flow arrival notification message etc.) that the controller(s) can process, defined as the number of asynchronous messages the controller(s) can process at its Southbound interface between the start of the test and the expiry of given test 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 have completed the network topology discovery for the connected Network Devices.

Procedure:

1. Generate asynchronous messages continuously at the maximum possible rate on the established connections from all the connected Network Devices in the forwarding plane test emulator for the Test Duration (Td).
2. Record the total number of responses received from the controller (Nrx) as well as the number of messages sent(Ntx) to the controller within the test duration(Td) at the forwarding plane test emulator interface (I1).

Measurement:

$$\text{Asynchronous Message Processing Rate Tr1} = \frac{Nrx}{Td}$$

$$\text{Average Asynchronous Message Processing Rate} = \frac{Tr1 + Tr2 + Tr3..Trn}{\text{Total Test Iterations}}$$

Loss Ratio = (Ntx-Nrx)/100.

Reporting Format:

The Asynchronous Message Processing 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 Asynchronous Message Processing Rate.

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

- Offered rate (Ntx)

- Loss Ratio

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 coordinate SHOULD be the Number of nodes (N), the Y coordinate SHOULD be the average Asynchronous Message Processing Rate.

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 Asynchronous Message Processing Rate.

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), 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.

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 test 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.

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

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.

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 test 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.

$$\text{Average Proactive Path Provisioning Time} = \frac{Tr1 + Tr2 + Tr3 \; .. \; Trn}{\text{Total Test Iterations}}$$

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 test 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 test duration (Td).

Measurement:

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

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

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 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 test and the expiry of given test 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 ’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 test duration (Td).

Measurement:

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

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

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.

Prerequisite:

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

2. The periodic network discovery operation should be configured to twice the Test 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 test when the controller sends the first topology re-discovery message to the Network Device or the expiry of test interval (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.

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

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. 6.2 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 test when the controller starts dropping the control connection.
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.

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. 3a. Increase the number of nodes by 1 when the comparison is successful and repeat the test.
4. 3b. Decrease the number of nodes by 1 when the comparison fails and repeat the test.
5. Continue the test until the comparison of step 3b is successful.
6. Record the number of nodes for the last iteration (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. 6.2.3 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 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 test 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. 6.3 Security

5.3.1. 6.3.1 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 test 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.

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 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 test 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 test 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).

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

[RFC2544] S. Bradner, J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices",RFC 2544, March 1999.

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

May 1998.

[RFC6241] R. Enns, M. Bjorklund, J. Schoenwaelder, A. Bierman, "Network Configuration Protocol (NETCONF)",RFC 6241, July 2011.

[RFC6020] M. Bjorklund, "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010

[RFC5440] JP. Vasseur, JL. Le Roux, "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, March 2009.

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

[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-02 (Work in progress), July 8, 2016

6.2. Informative References

[I-D.i2rs-architecture] A. Atlas, J. Halpern, S. Hares, D. Ward, T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-09 (Work in progress), March 6, 2015

[OpenContrail] Ankur Singla, Bruno Rijsman, "OpenContrail Architecture Documentation", http://opencontrail.org/opencontrail-architecture-documentation

[OpenDaylight] OpenDaylight Controller:Architectural Framework, https://wiki.opendaylight.org/view/OpenDaylight_Controller

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.

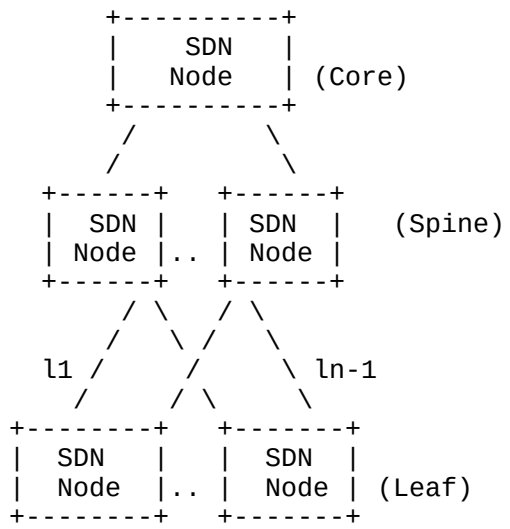
9. Acknowledgments

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)

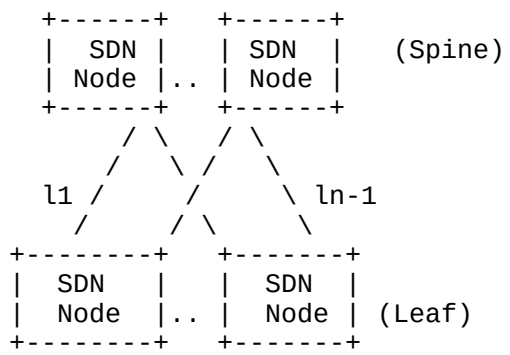
This document was prepared using 2-Word-v2.0.template.dot.

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), 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).

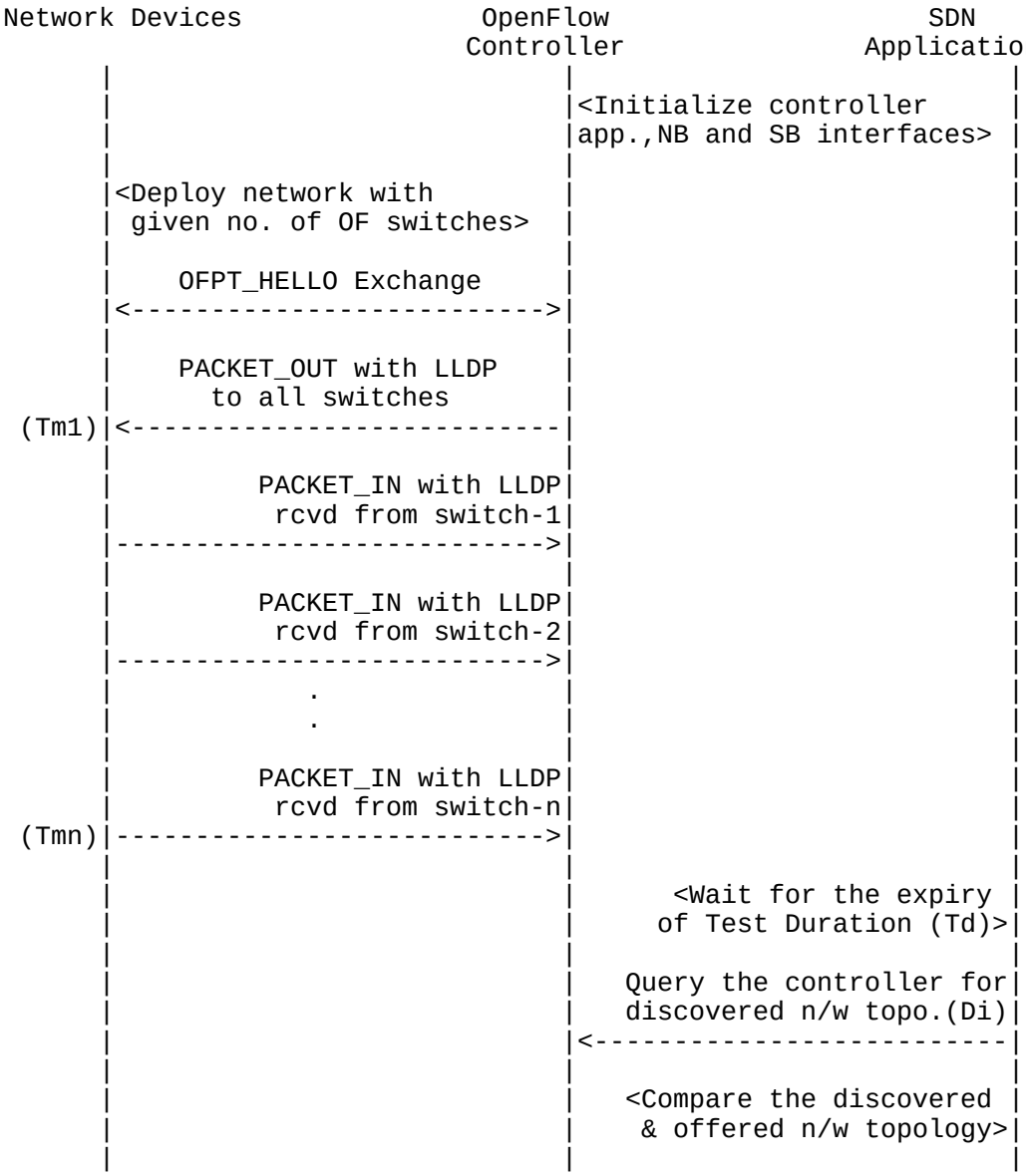
B.3. Connection Overview

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:

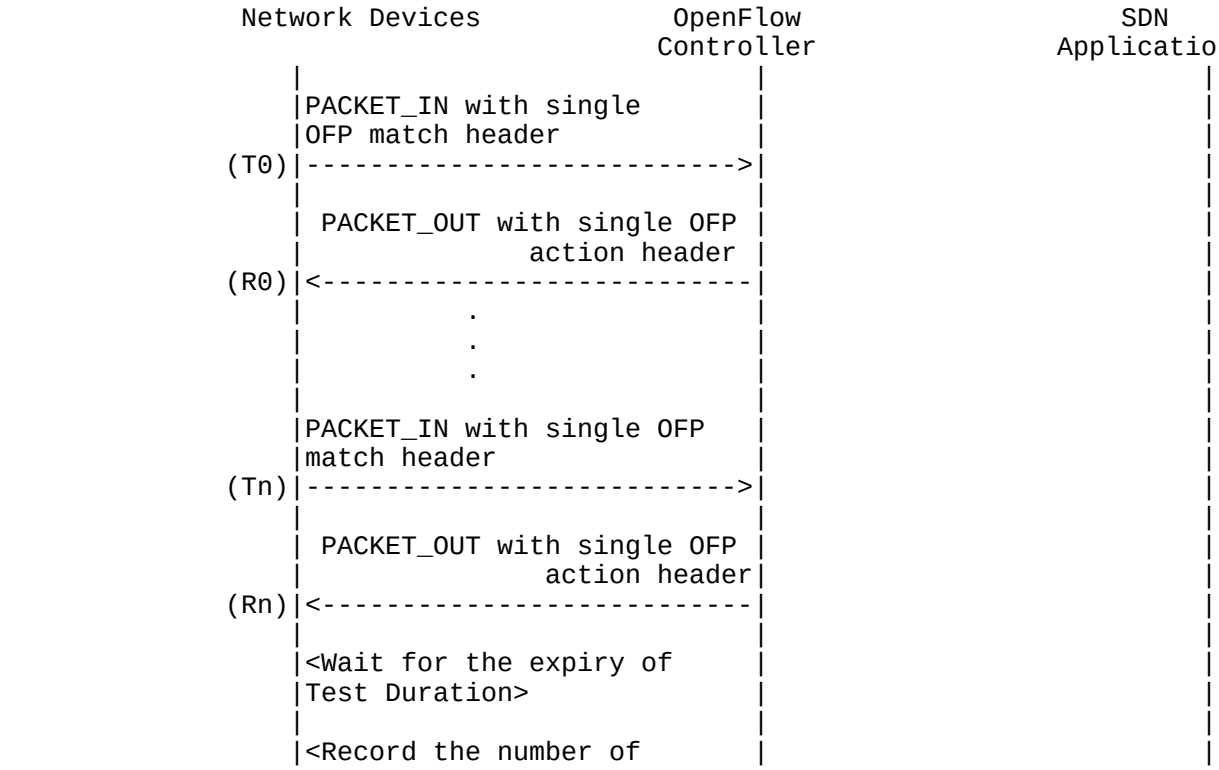
NB: Northbound
SB: Southbound
OF: OpenFlow
Tm1: Time of reception of first LLDP message from controller
Tmn: Time of last LLDP message sent to controller

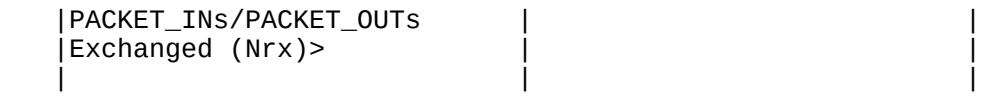
Discussion:

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:

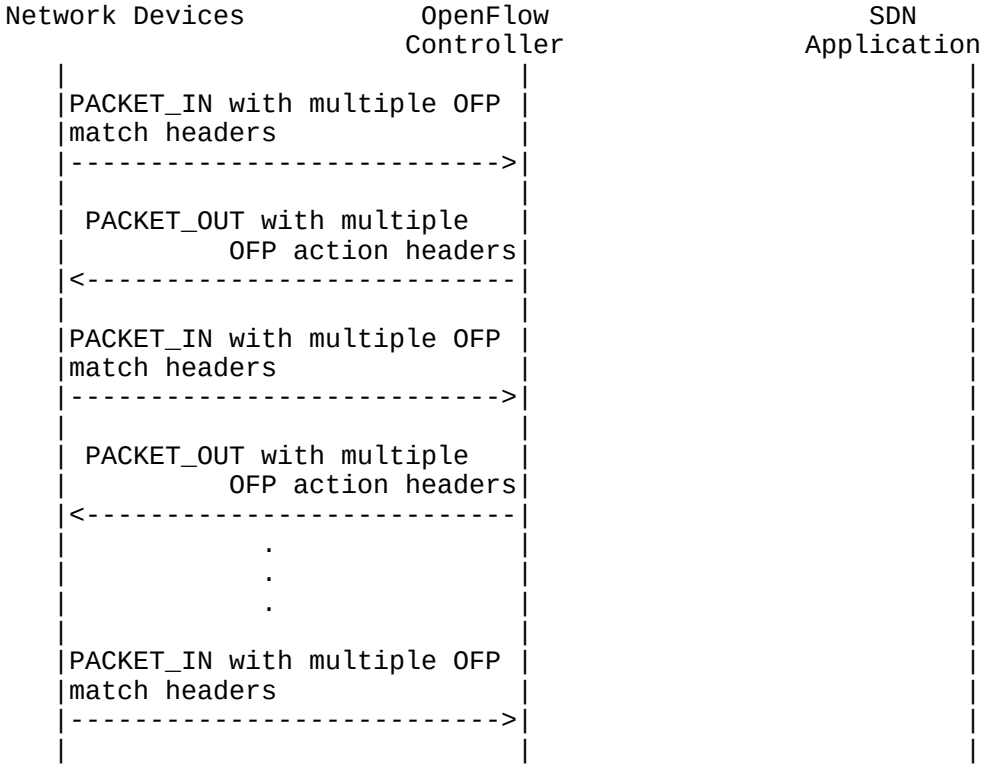
T0,T1, ..Tn are PACKET_IN messages transmit timestamps.
R0,R1, ..Rn are PACKET_OUT messages receive timestamps.
Nrx : Number of successful PACKET_IN/PACKET_OUT message exchanges

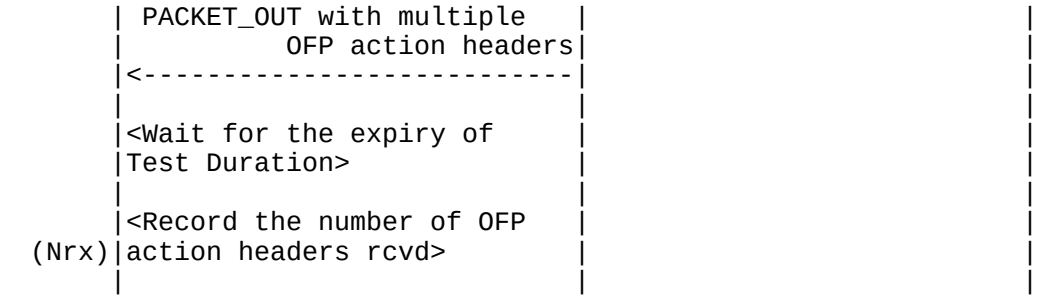
Discussion:

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:



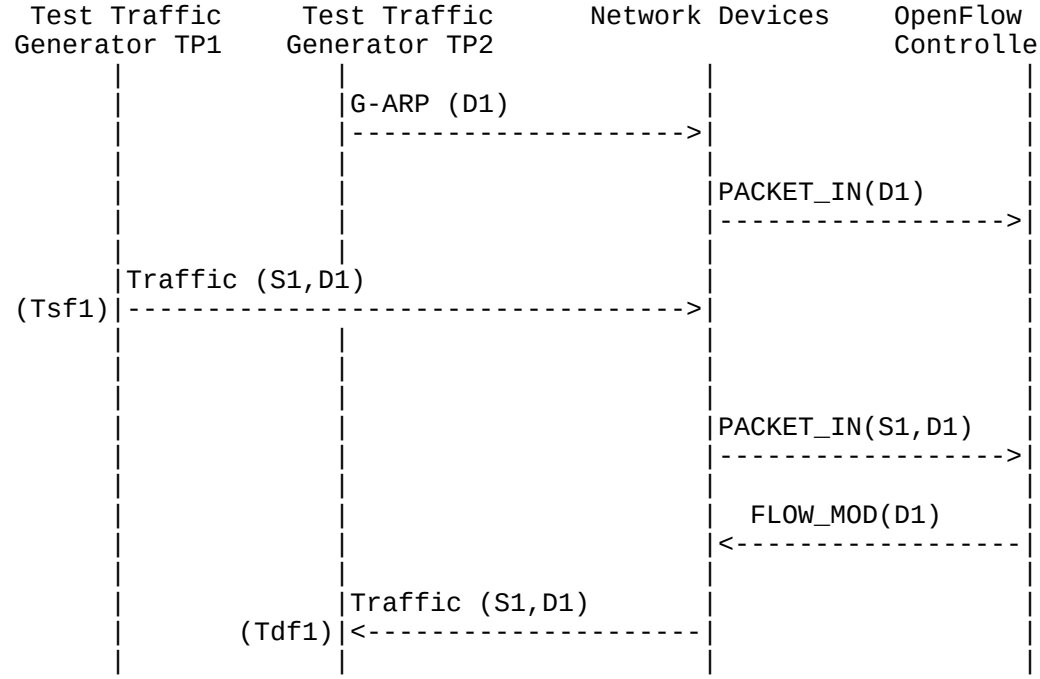


Discussion:

The Asynchronous Message Processing Rate will be obtained by calculating the number of OFP action headers received in all PACKET_OUT messages during the test duration.

B.4.4. Reactive Path Provisioning Time

Procedure:



Legend:

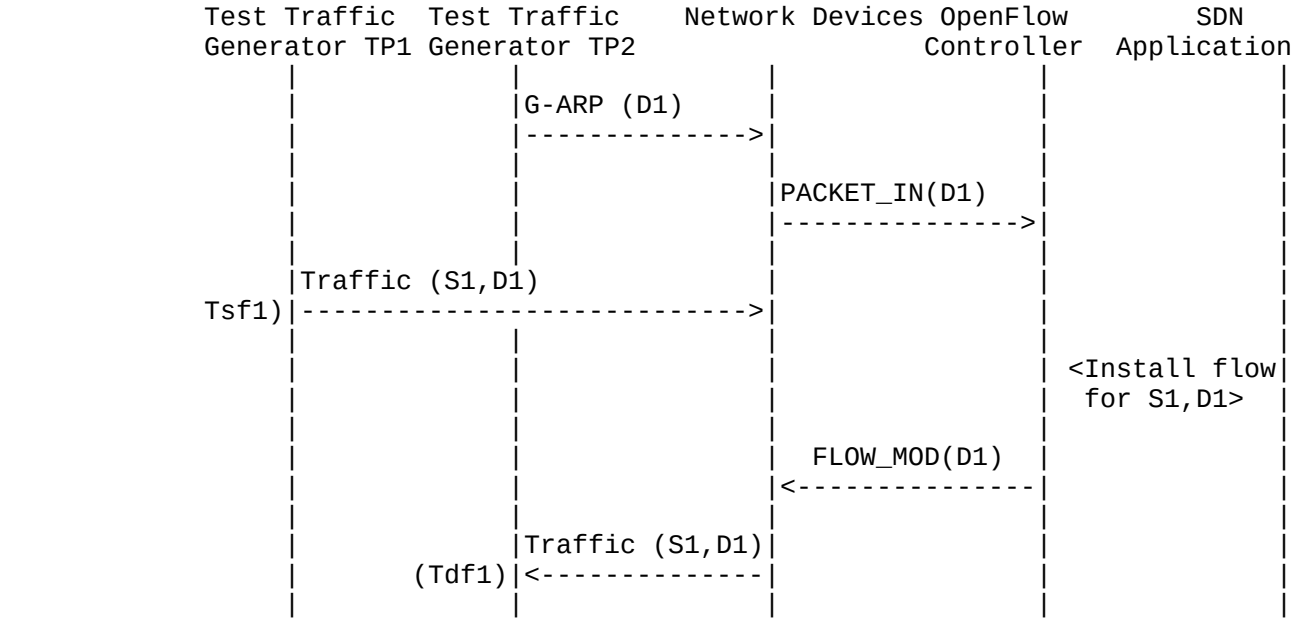
G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

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:

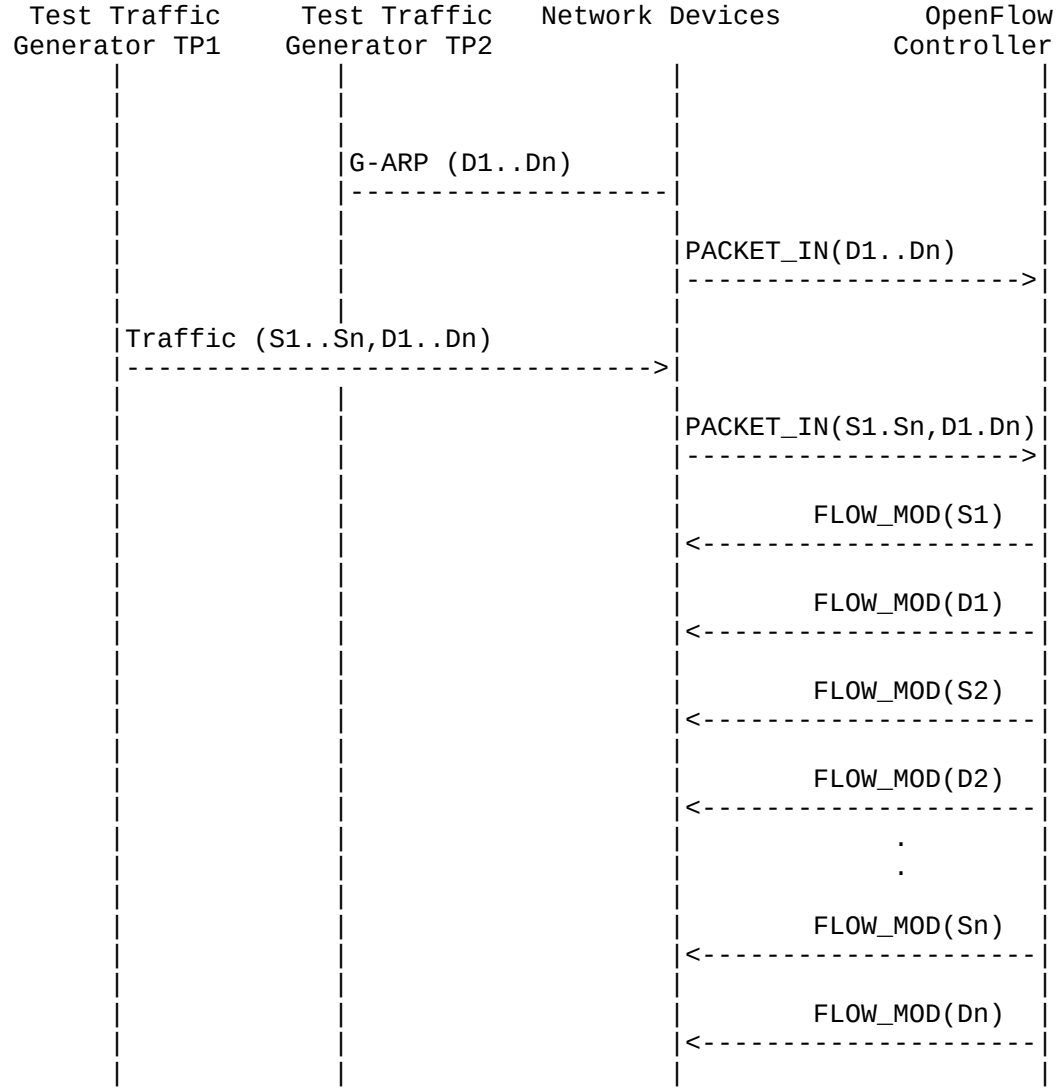
G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

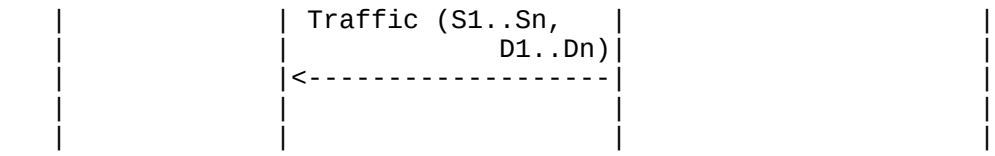
Discussion:

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:

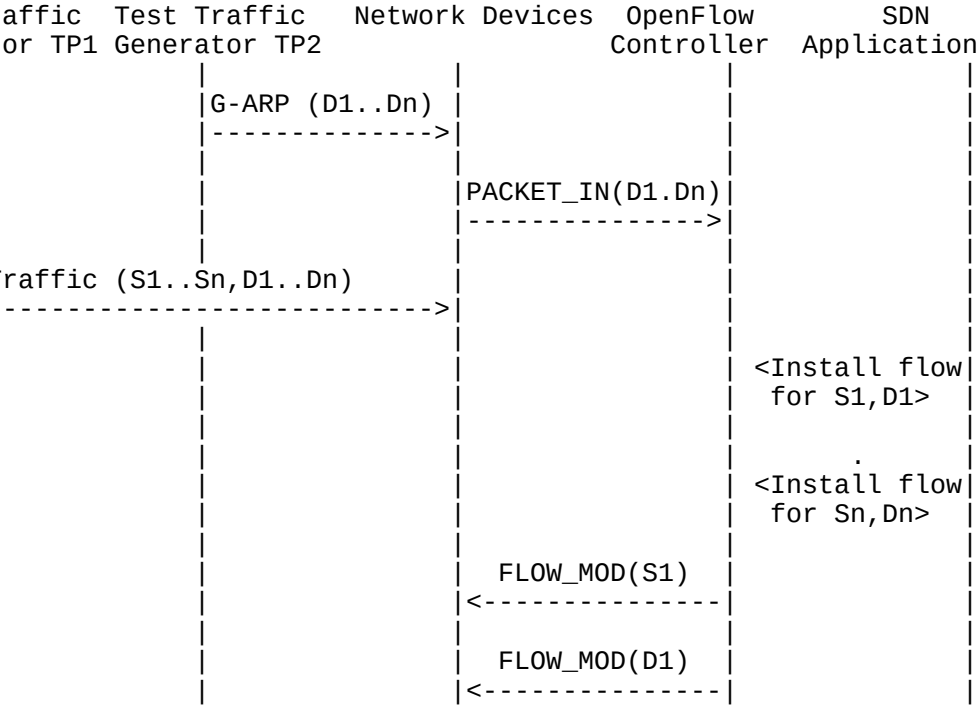
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 .... Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

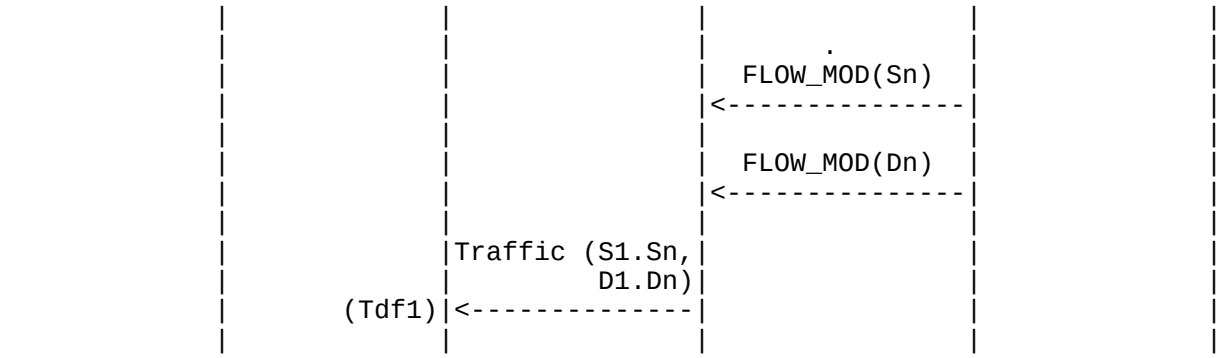
Discussion:

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

B.4.7. Proactive Path Provisioning Rate

Procedure:





Legend:

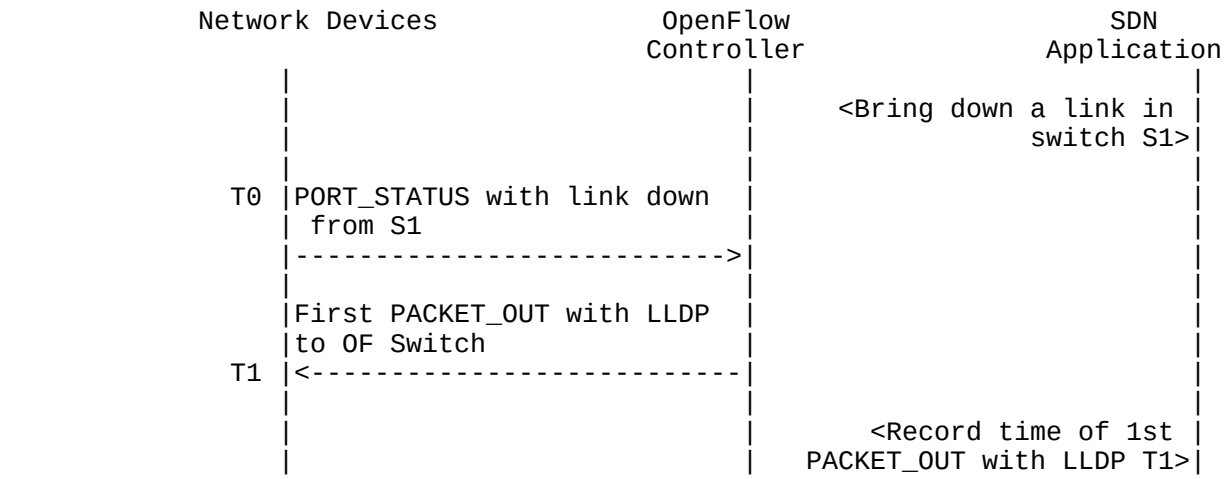
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 .... Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

Discussion:

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

B.4.8. Network Topology Change Detection Time

Procedure:



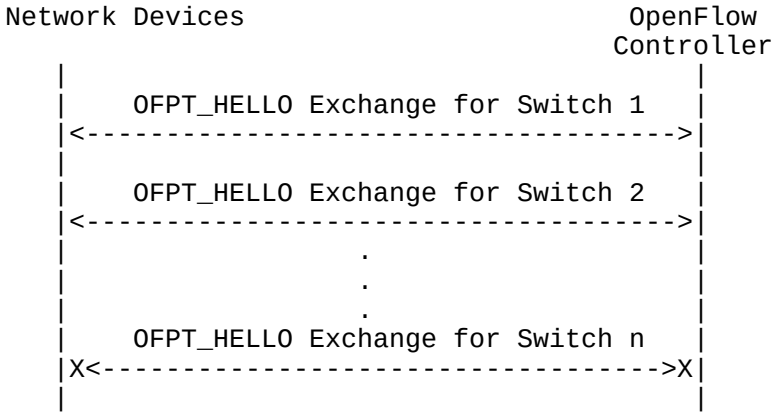
Discussion:

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:

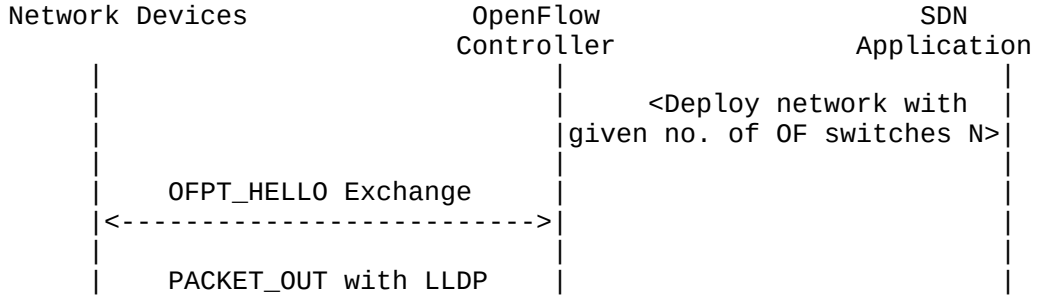


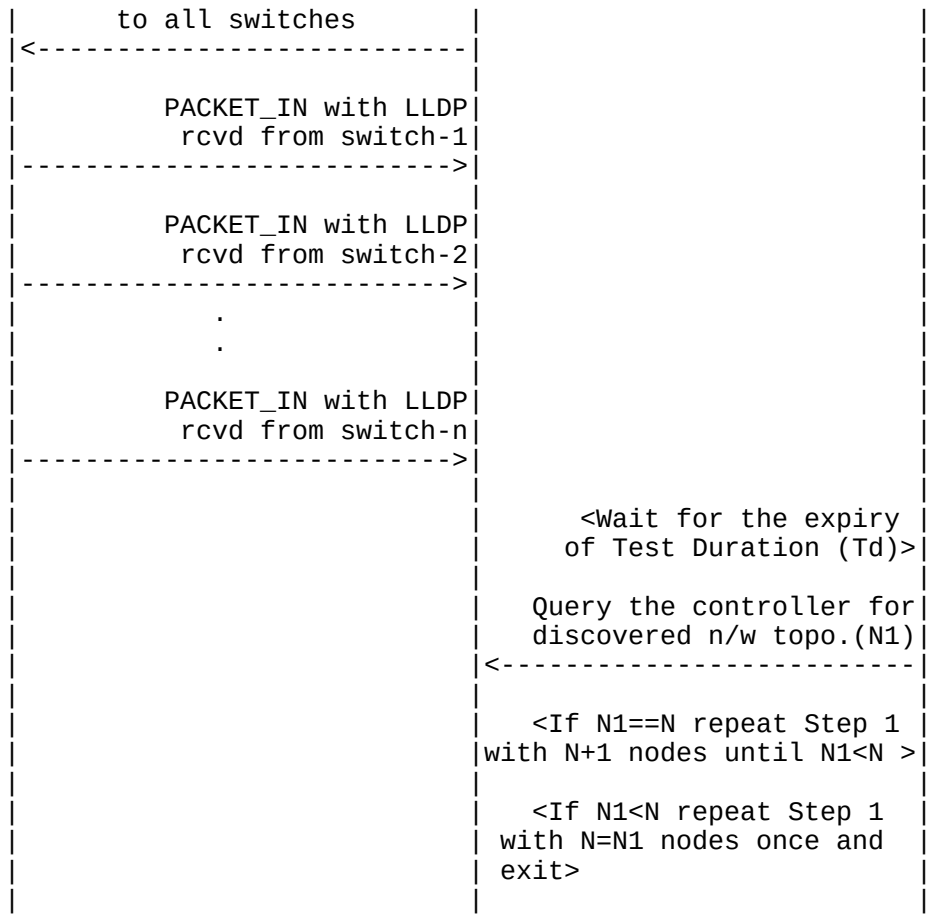
Discussion:

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

B.5.2. Network Discovery Size

Procedure:





Legend:

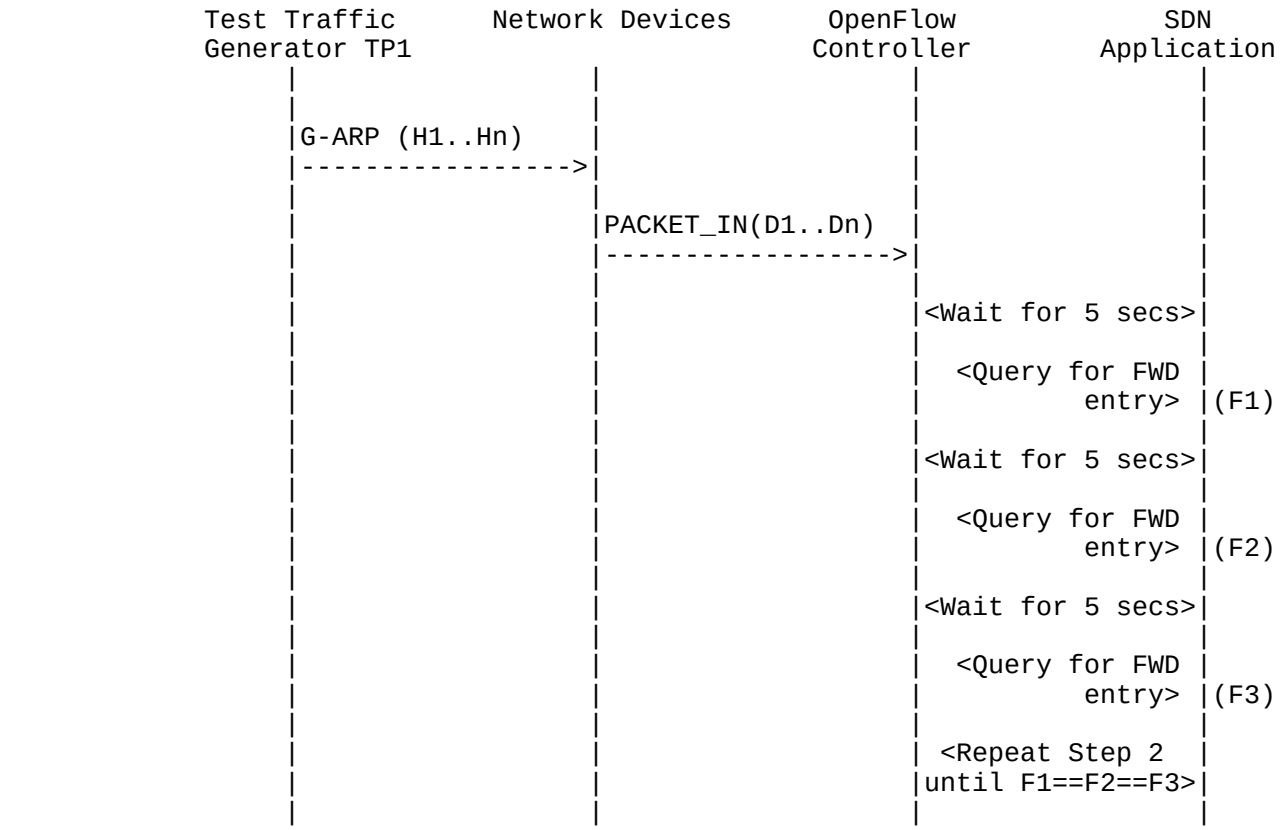
n/w topo: Network Topology
OF: OpenFlow

Discussion:

The value of N1 provides the Network Discovery Size value. The test 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

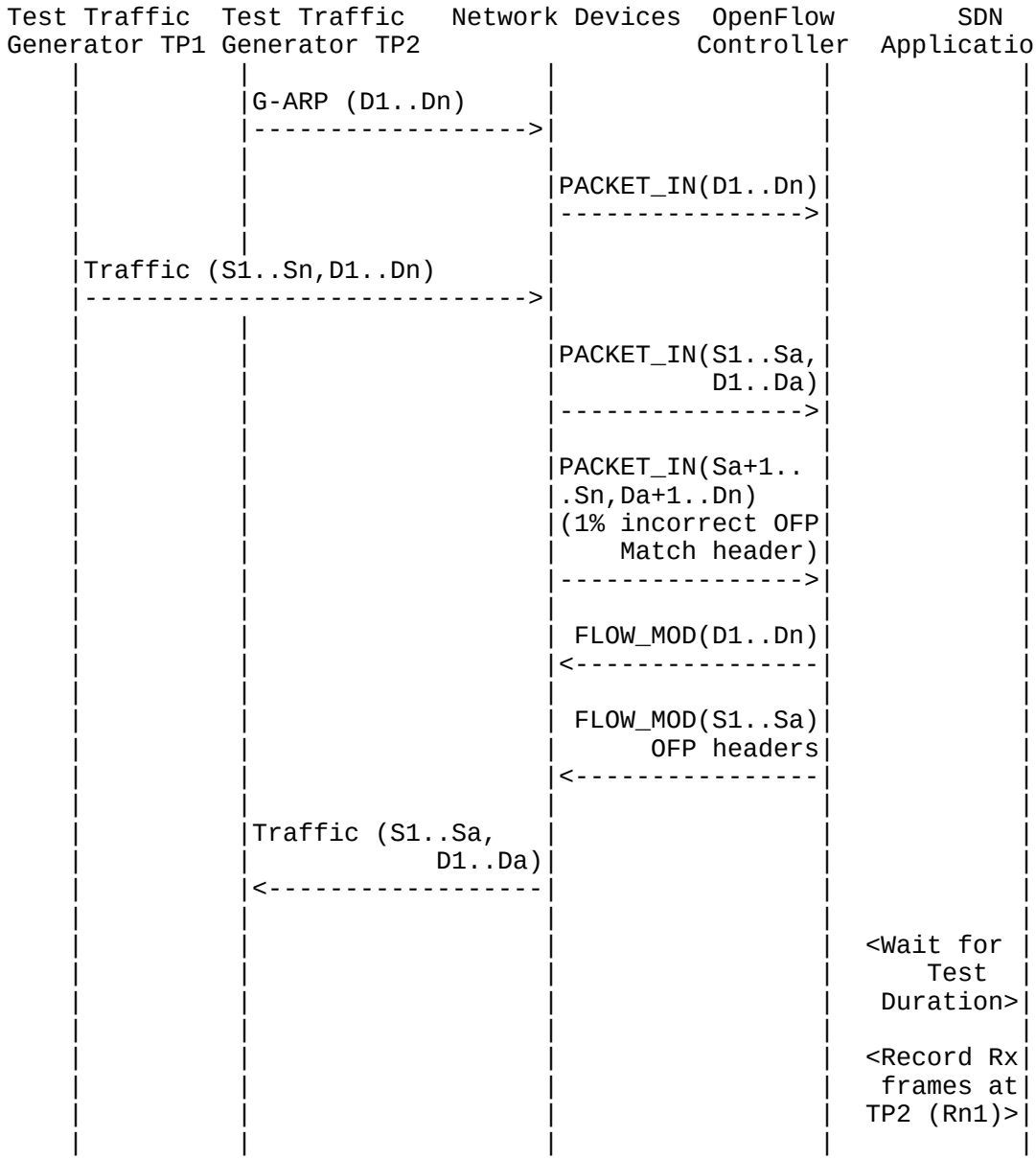
Discussion:

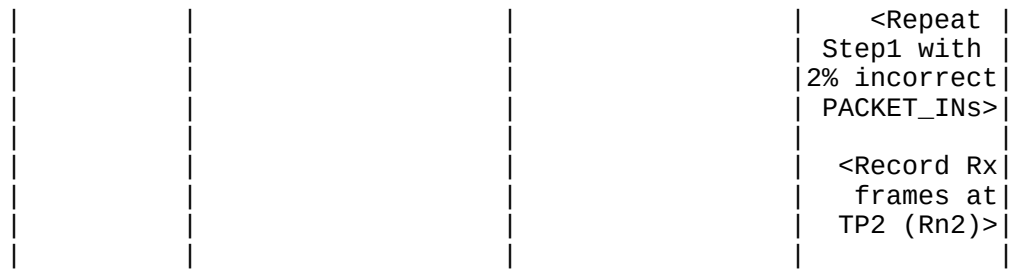
Query the controller forwarding table entries for multiple times until the three consecutive queries return the same value. The last value retrieved from the controller will provide the Forwarding Table Capacity value. The query interval is user configurable. The 5 seconds shown in this example is for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

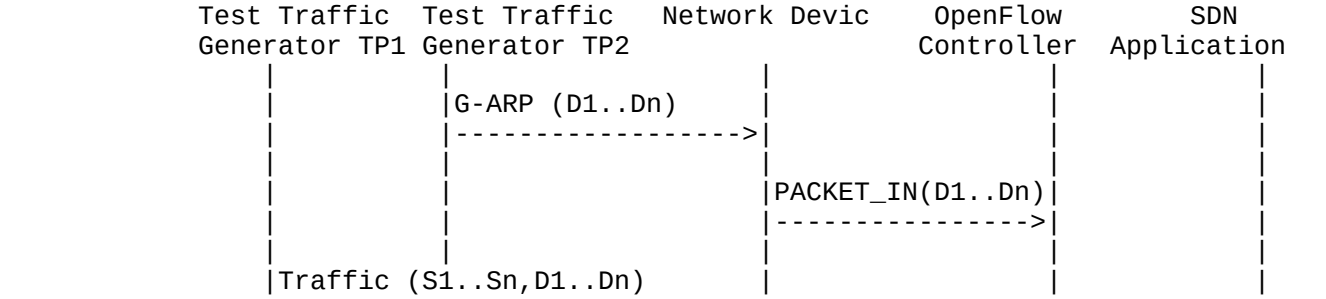
Discussion:

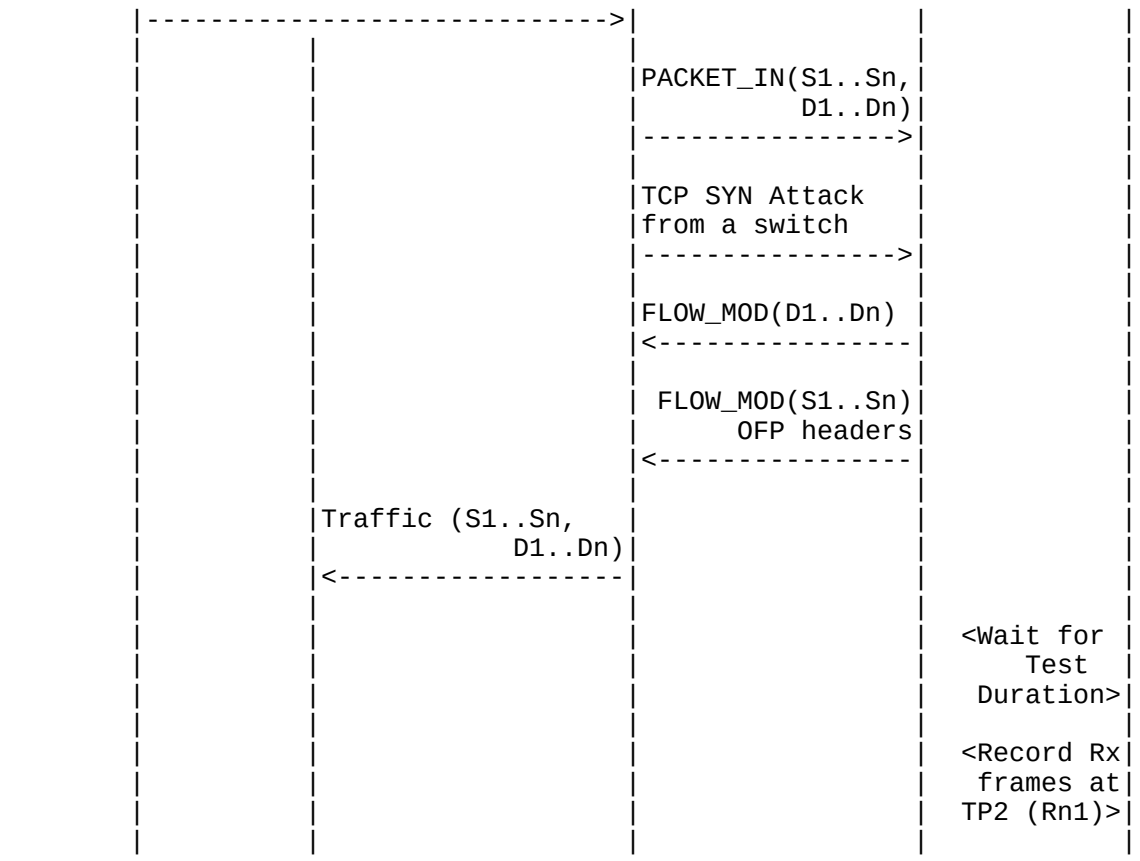
The traffic rate sent towards OpenFlow switch from Test Port 1 should be 1% higher than the Path Programming Rate. Rn1 will provide the Path Provisioning Rate of controller at 1% of incorrect frames handling and Rn2 will provide the Path Provisioning Rate of controller at 2% of incorrect frames handling.

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:

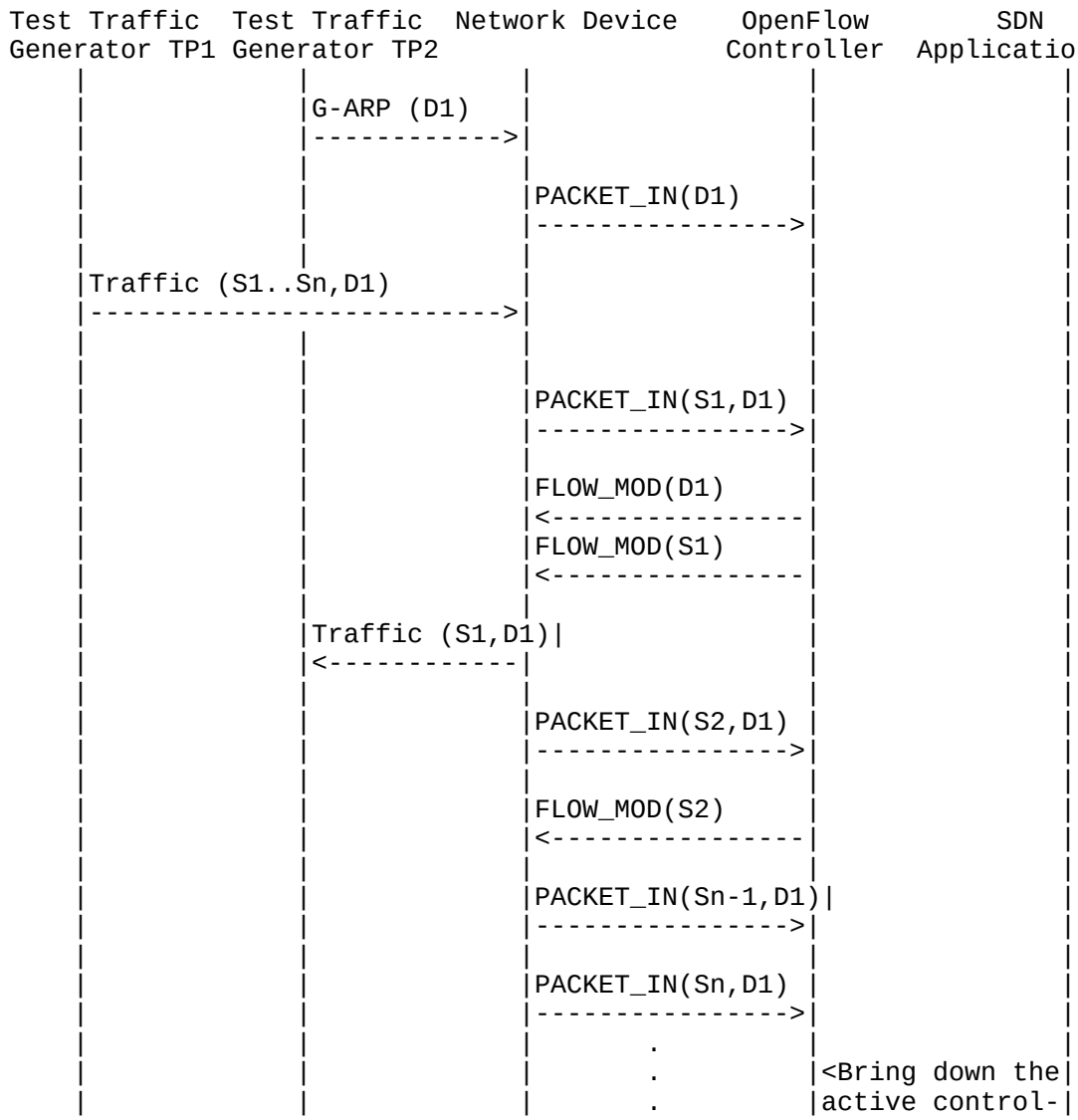
TCP SYN attack should be launched from one of the emulated/simulated OpenFlow Switch. Rn1 provides the Path Programming Rate of controller uponhandling denial of service attack.

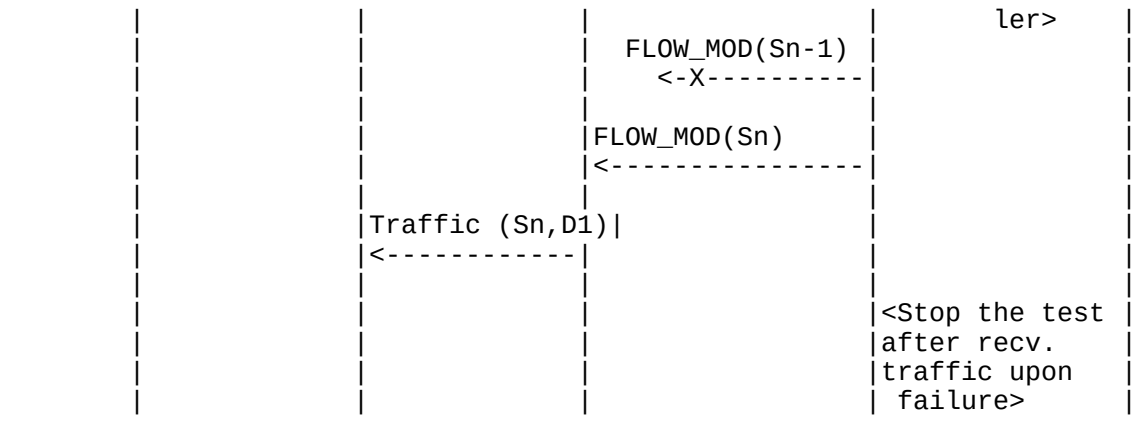
The procedure defined above provides test steps to determine the effect of handling denial of service on Path Programming Rate. Same procedure can be adopted to determine the effects on other performance tests listed in this benchmarking tests.

B.7. Reliability

B.7.1. Controller Failover Time

Procedure:





Legend:

G-ARP: Gratuitous ARP.

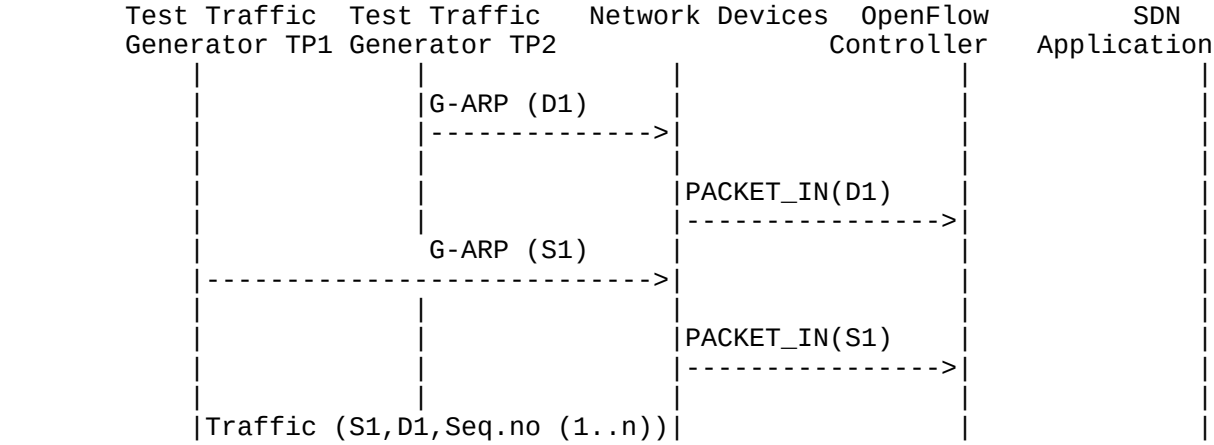
Discussion:

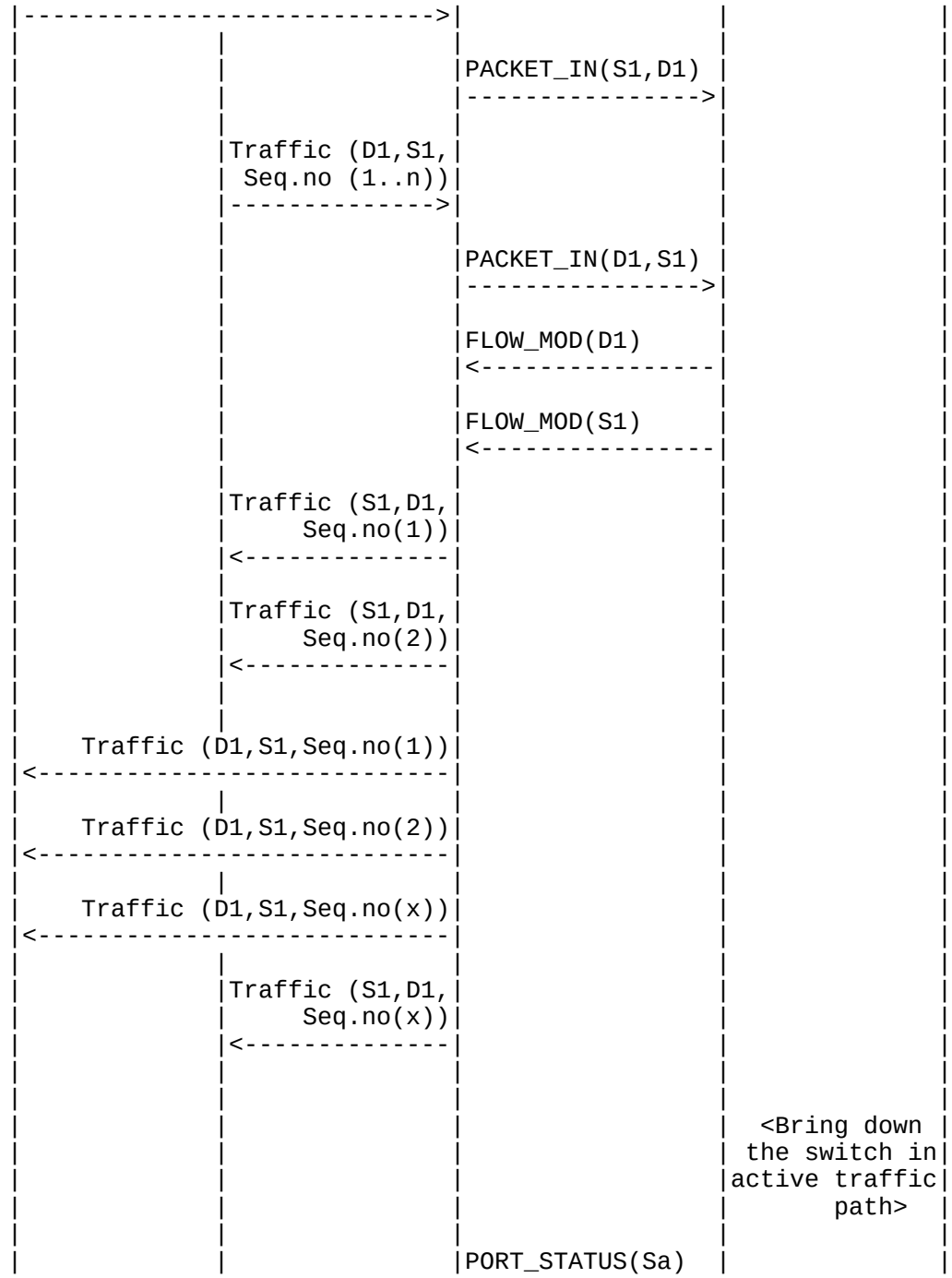
The time difference between the last valid frame received before the traffic loss and the first frame received after the traffic loss will provide the controller failover time.

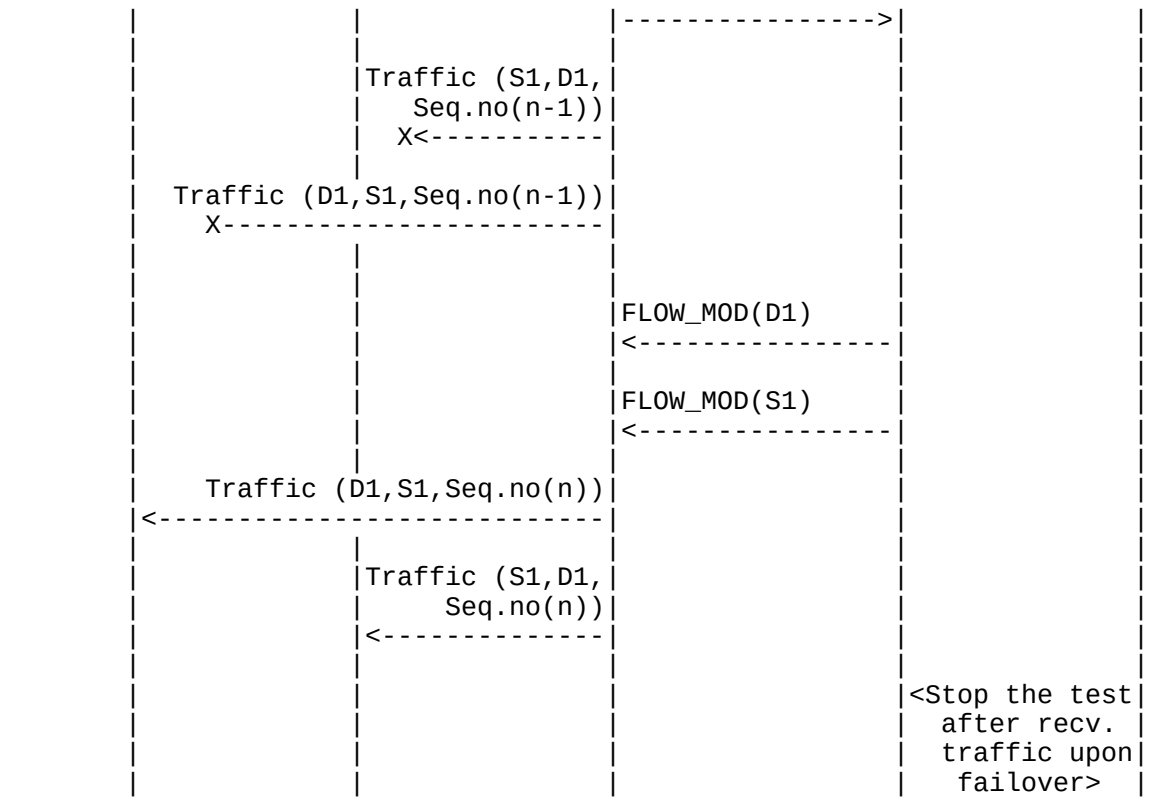
If there is no frame loss during controller failover time, the controller failover time can be deemed negligible.

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: Neighbour 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 test 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

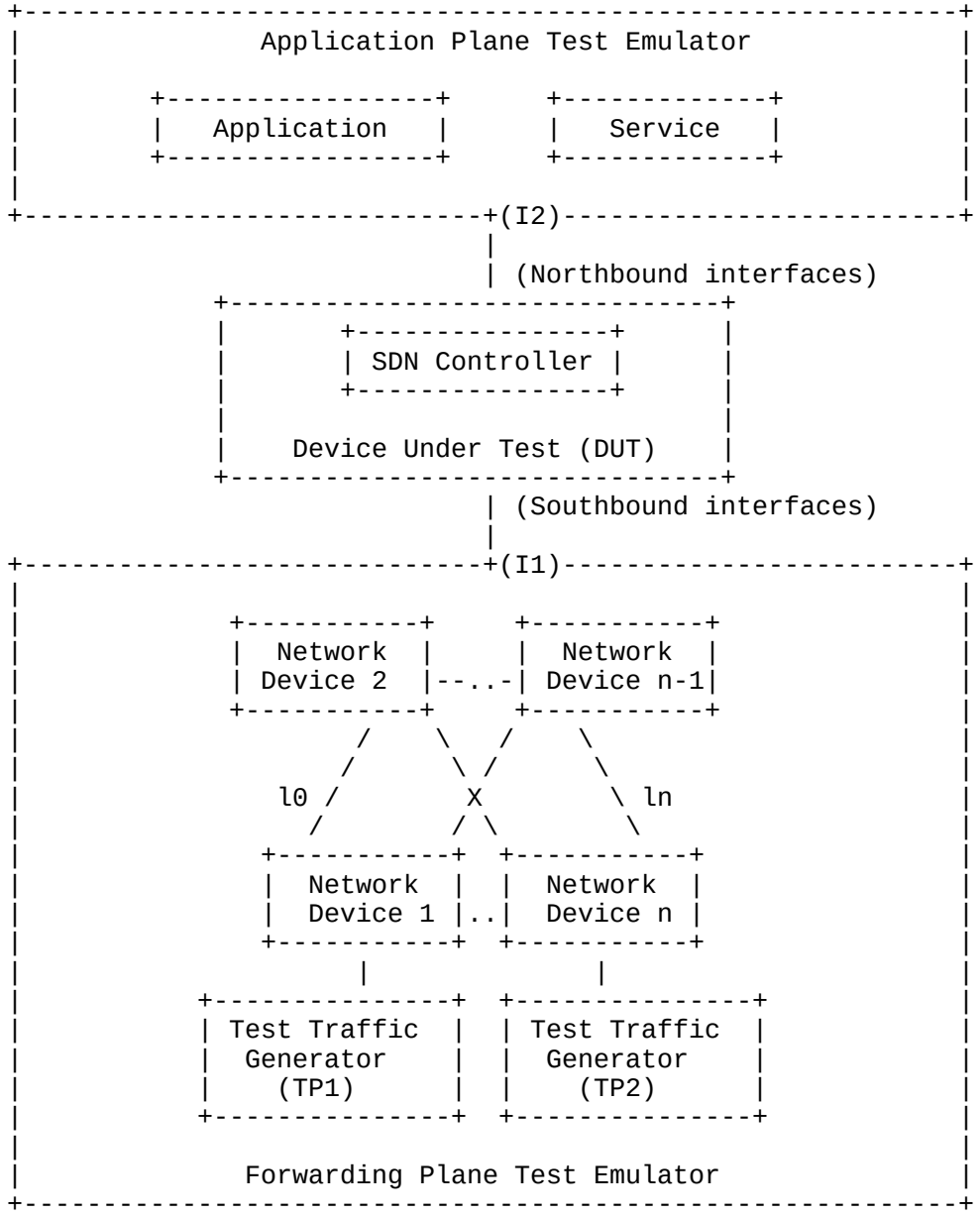


Figure 1

3.2. Test setup - Controller working in Cluster Mode

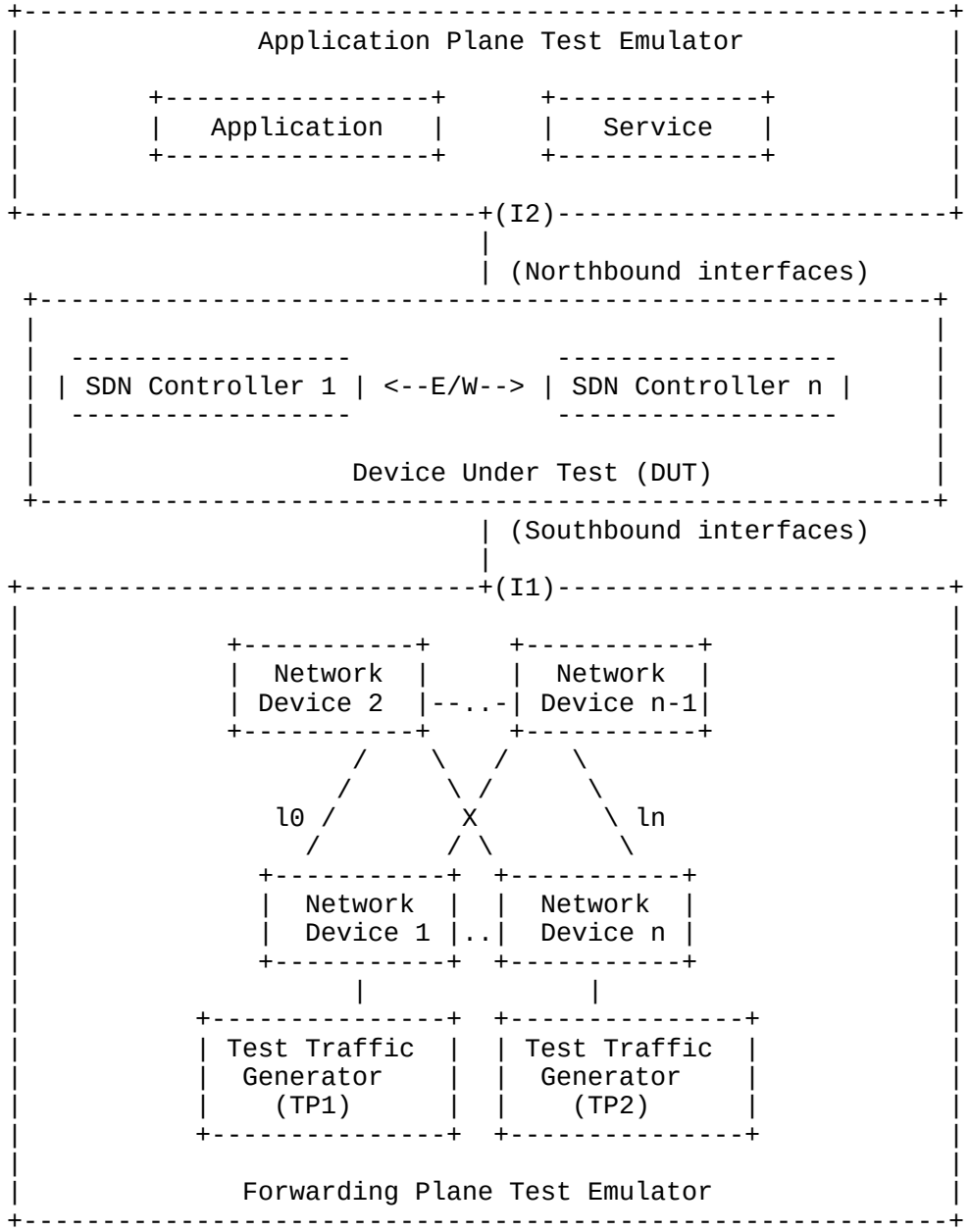


Figure 2

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).

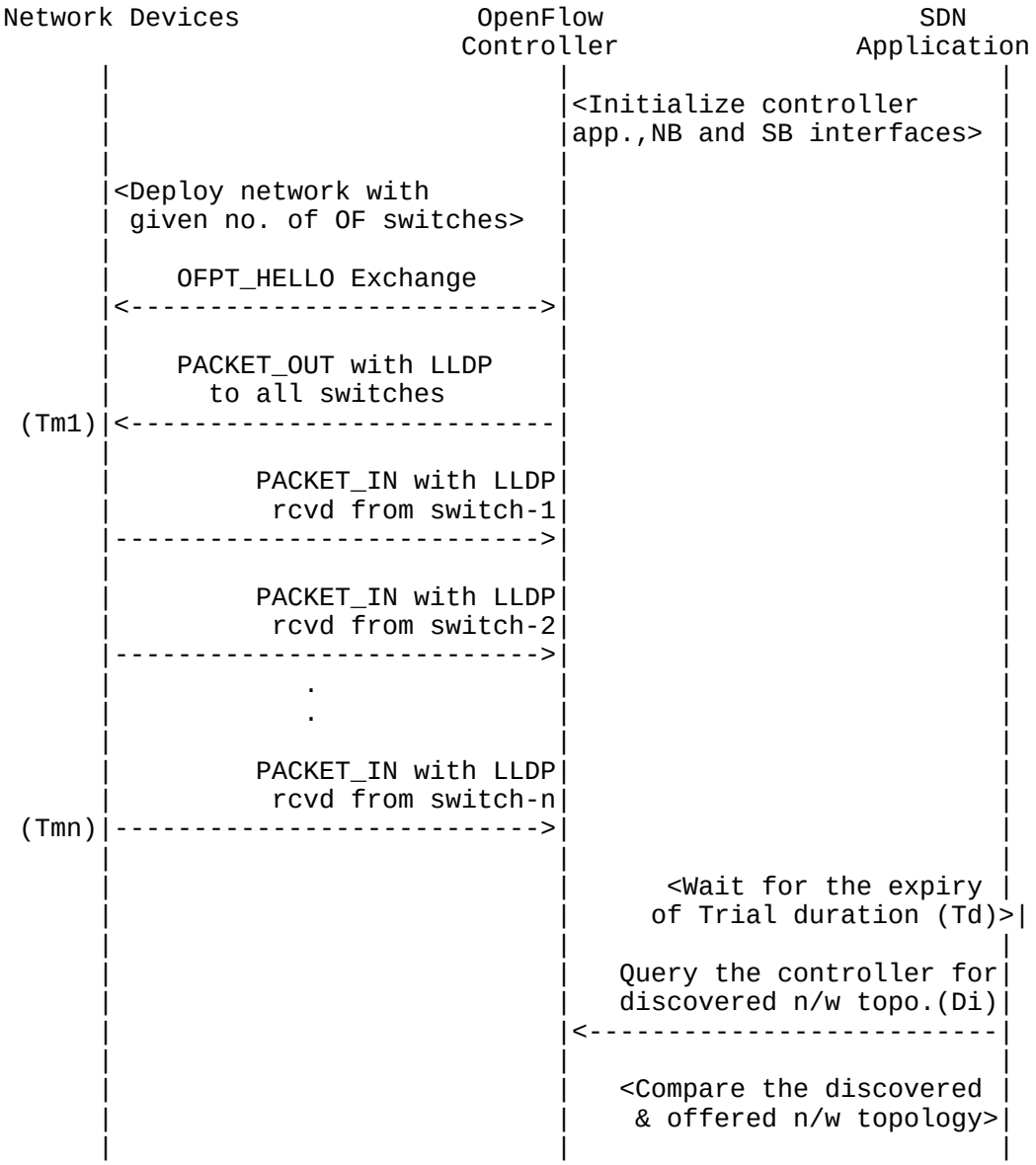
A.3. Connection Overview

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:

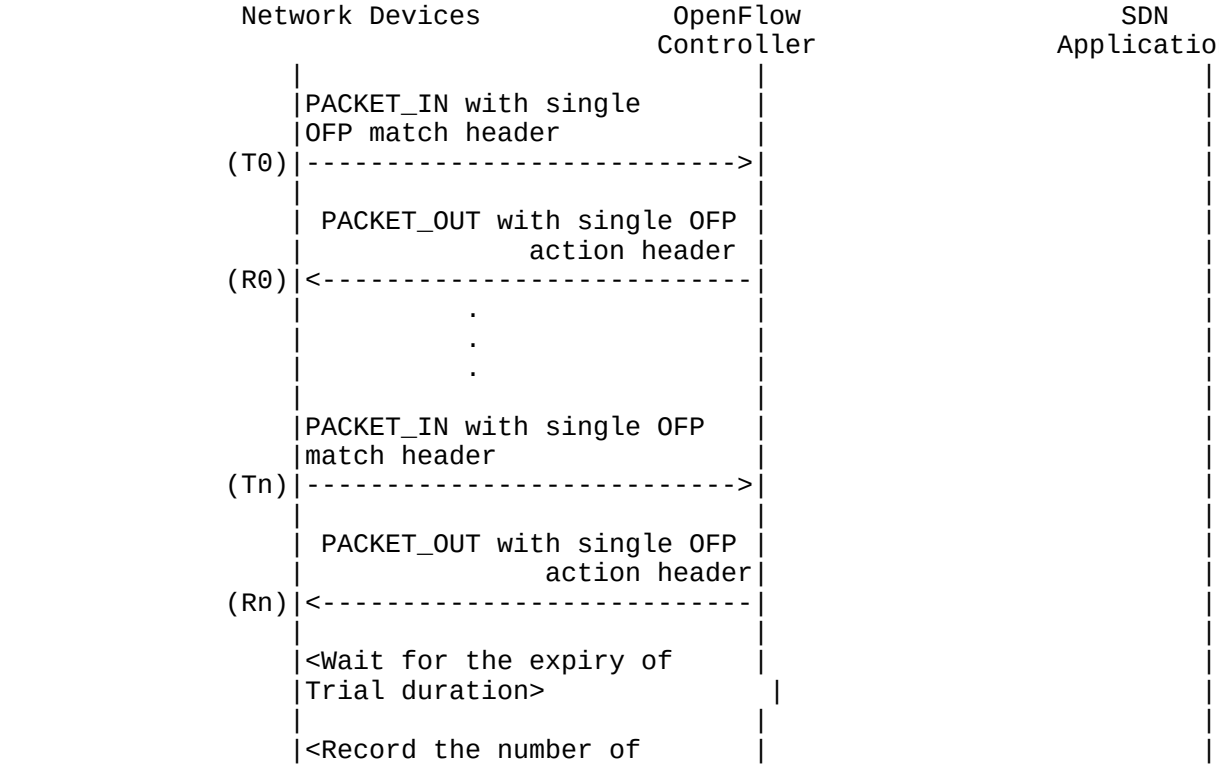
NB: Northbound
SB: Southbound
OF: OpenFlow
Tm1: Time of reception of first LLDP message from controller
Tmn: Time of last LLDP message sent to controller

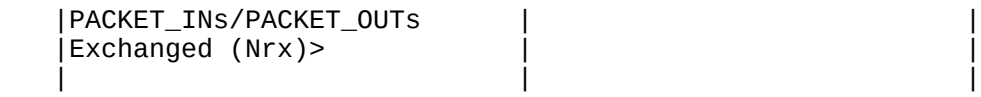
Discussion:

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:

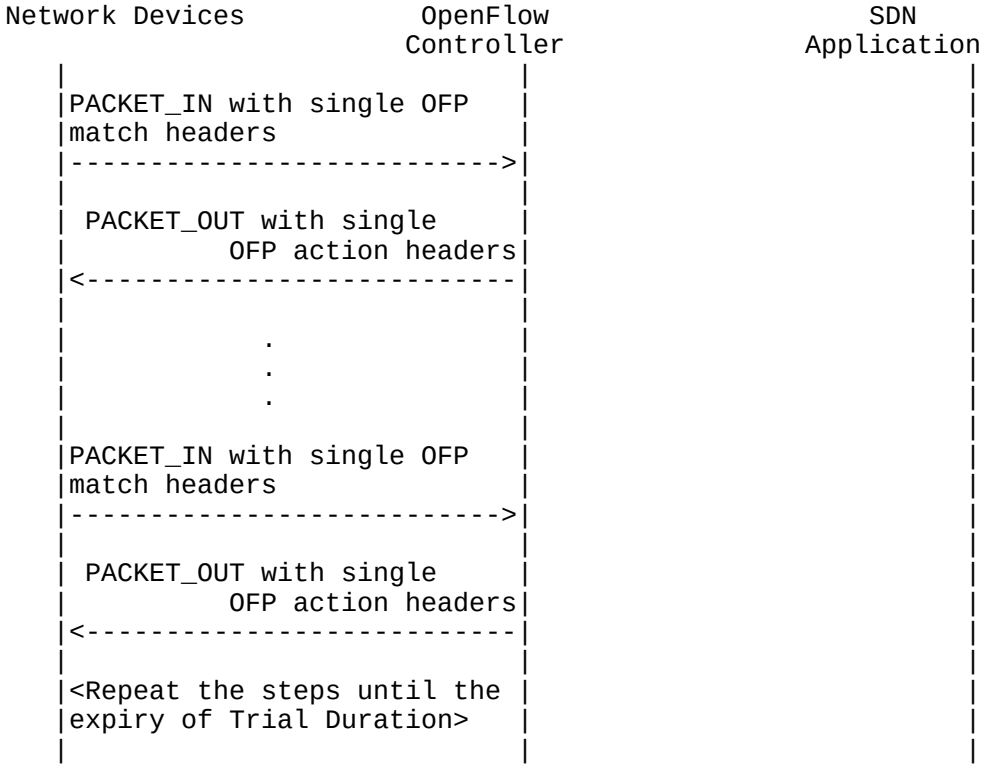
T0,T1, ..Tn are PACKET_IN messages transmit timestamps.
R0,R1, ..Rn are PACKET_OUT messages receive timestamps.
Nrx : Number of successful PACKET_IN/PACKET_OUT message exchanges

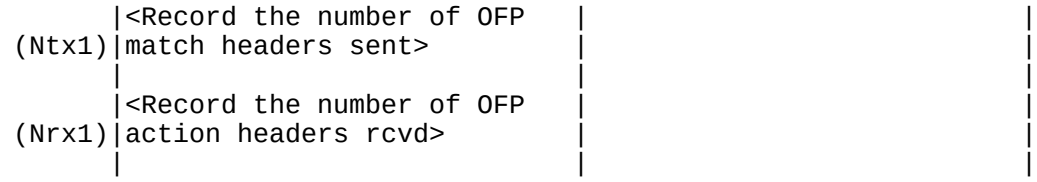
Discussion:

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:





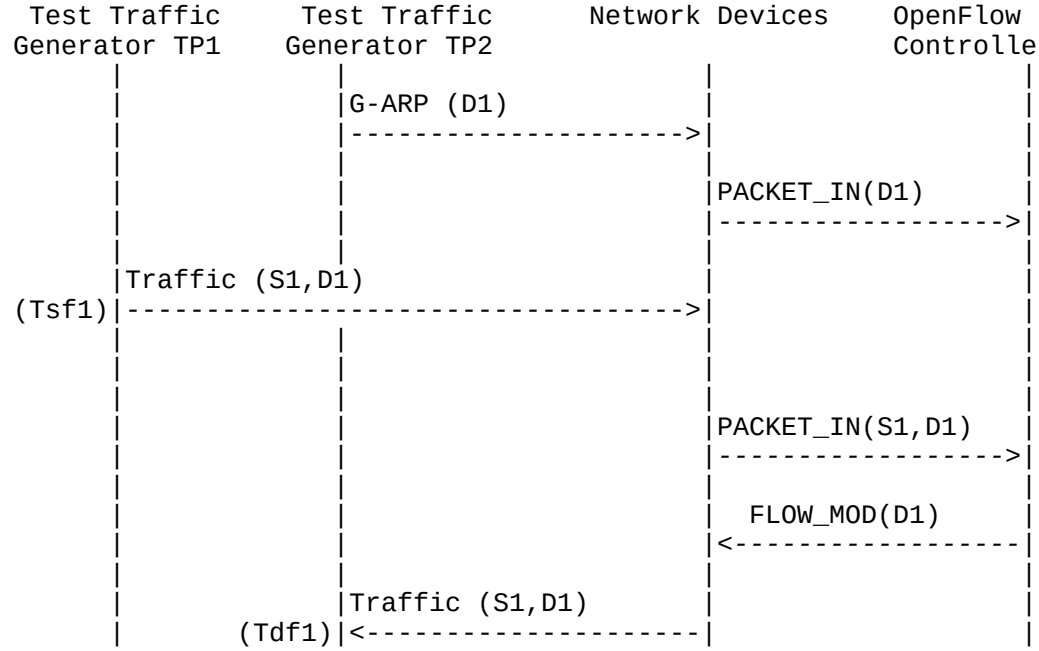
Note: The Ntx1 on initial trials should be greater than Nrx1 and repeat the trials until the Nrxn for two consecutive trials equeal to (+/-P%).

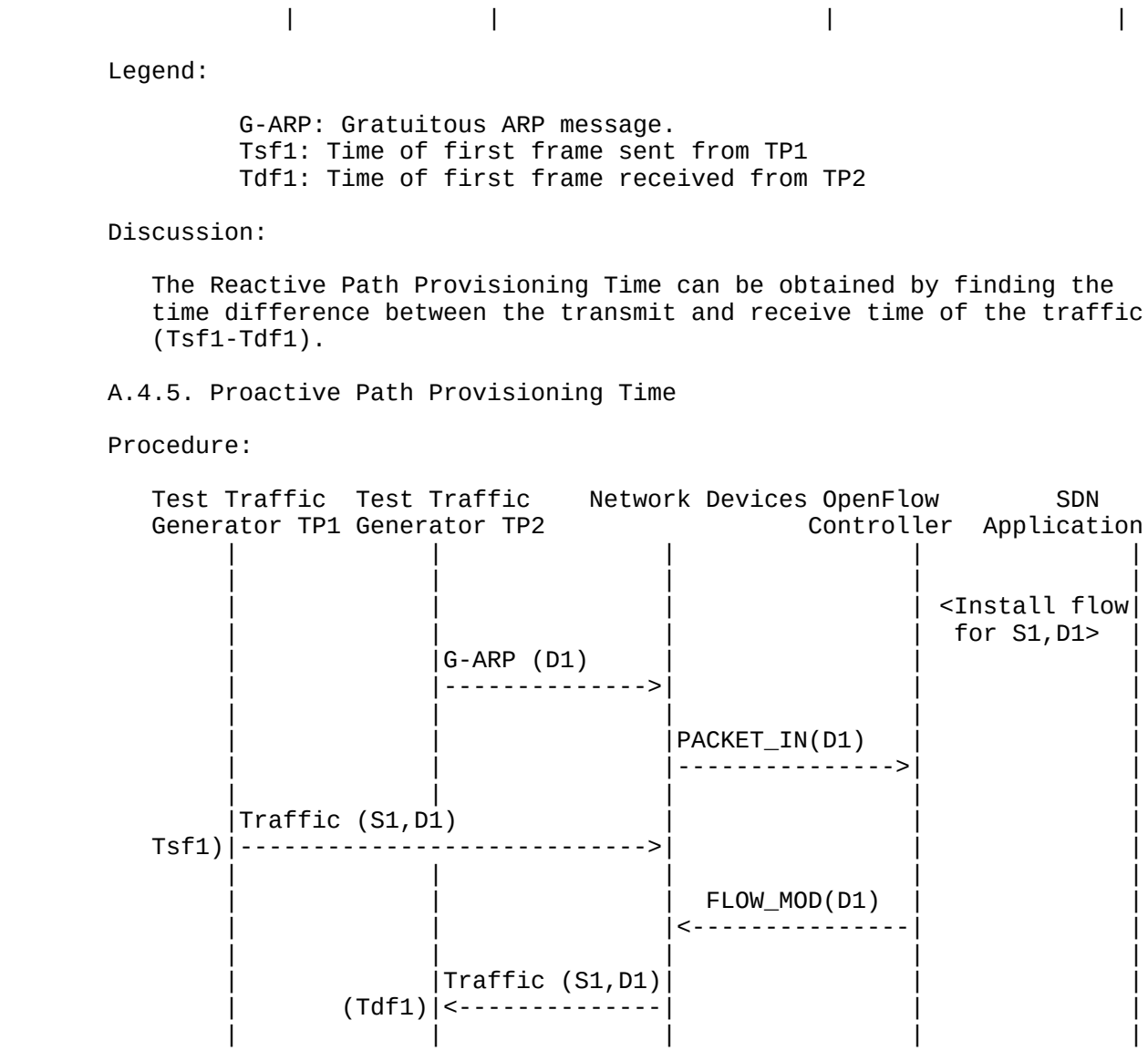
Discussion:

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:

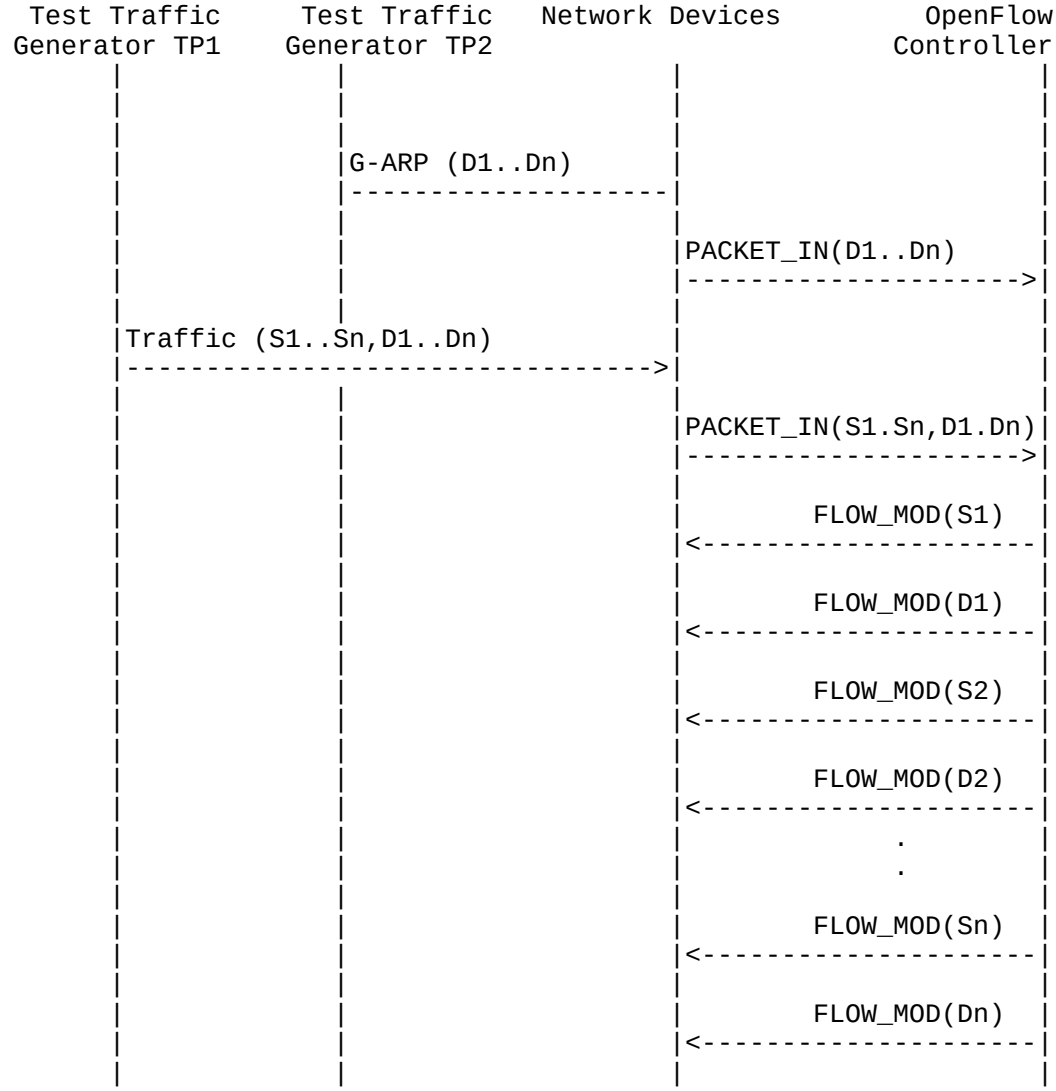
G-ARP: Gratuitous ARP message.
Tsf1: Time of first frame sent from TP1
Tdf1: Time of first frame received from TP2

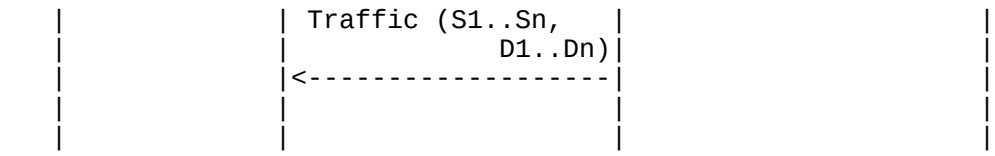
Discussion:

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:

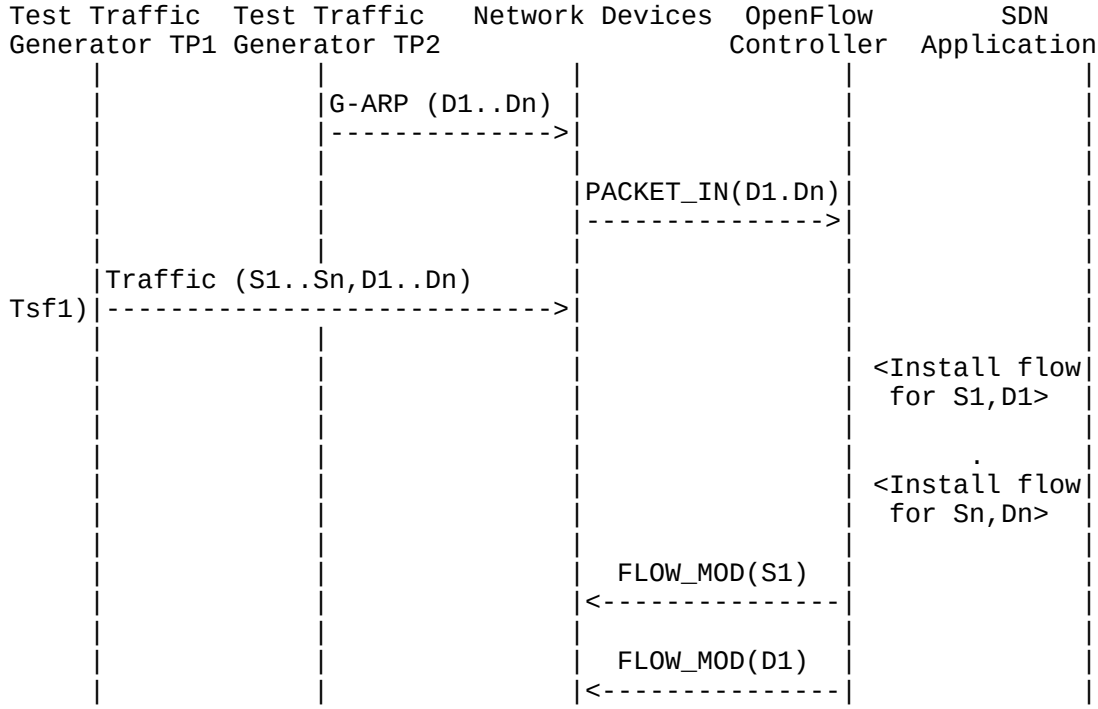
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ....
Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

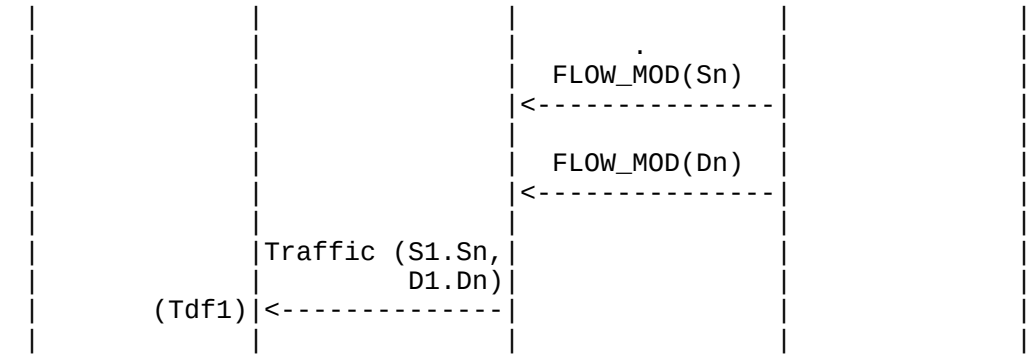
Discussion:

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:

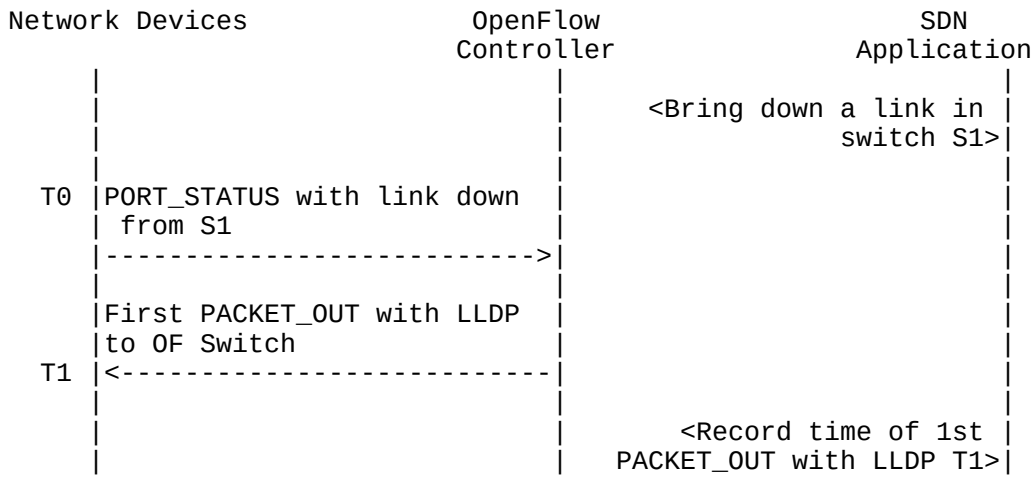
G-ARP: Gratuitous ARP
D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ....
Destination Endpoint n
S1..Sn: Source Endpoint 1, Source Endpoint 2 .., Source Endpoint n

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:



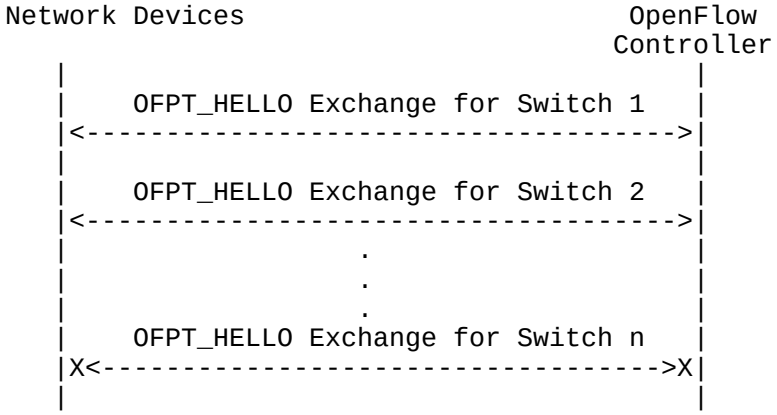
Discussion:

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:

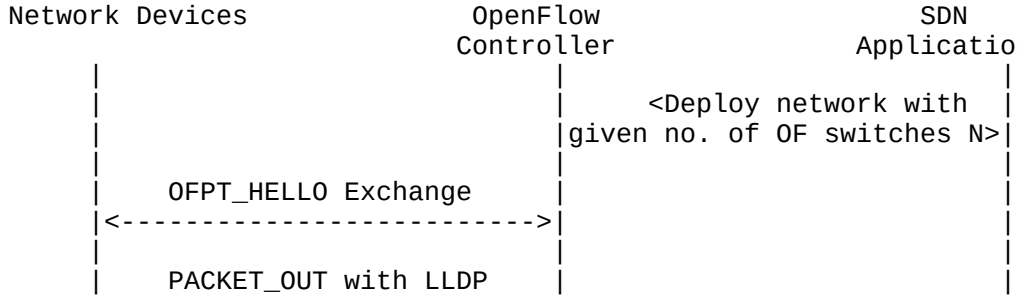


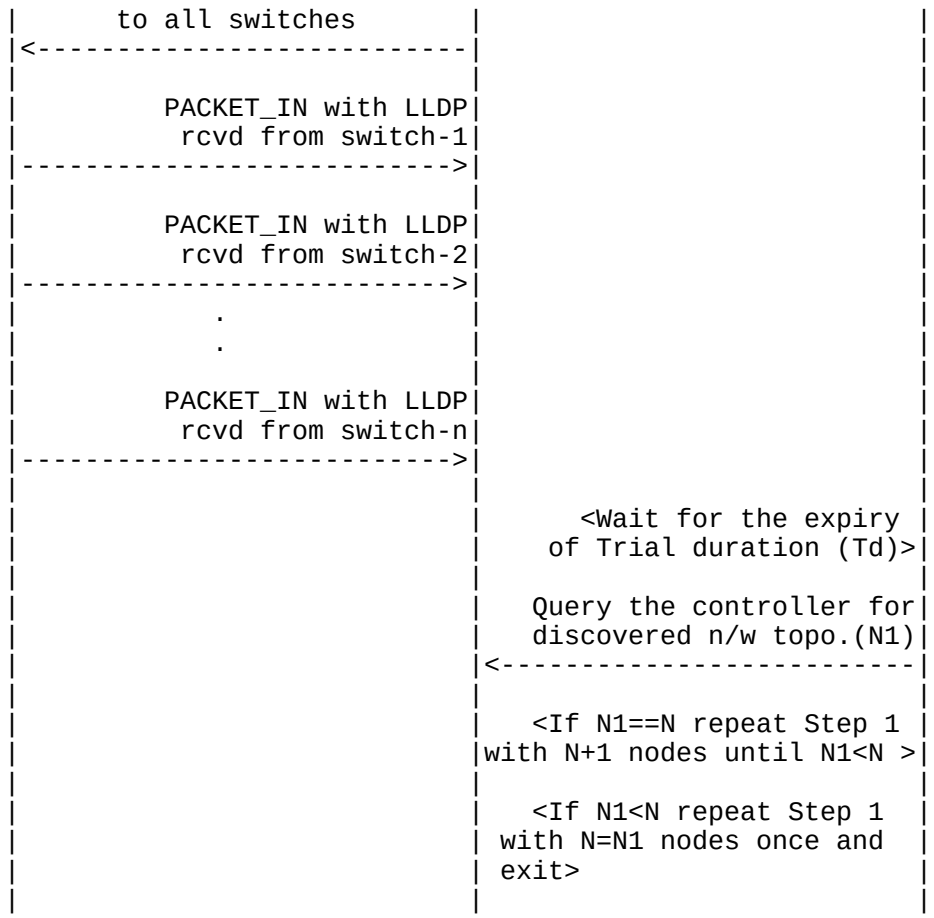
Discussion:

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

A.5.2. Network Discovery Size

Procedure:





Legend:

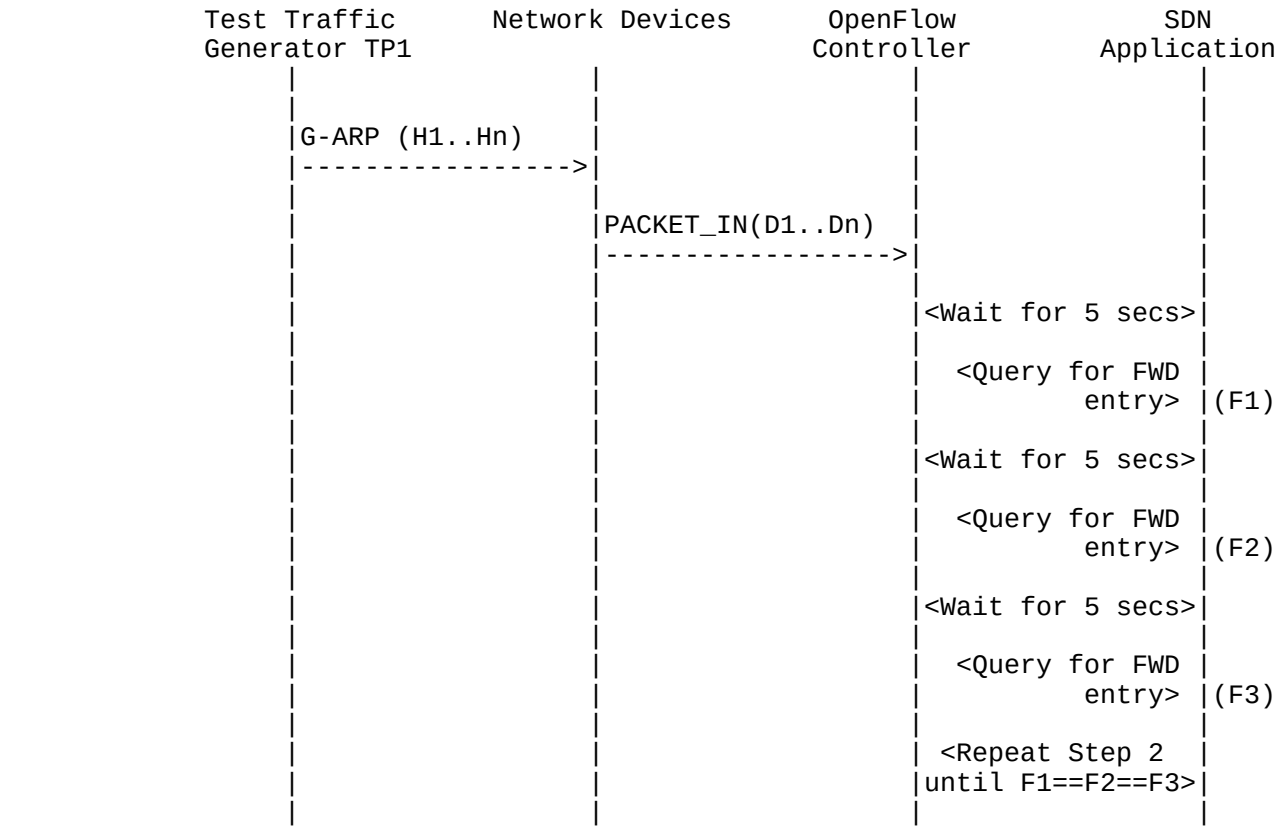
n/w topo: Network Topology
OF: OpenFlow

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

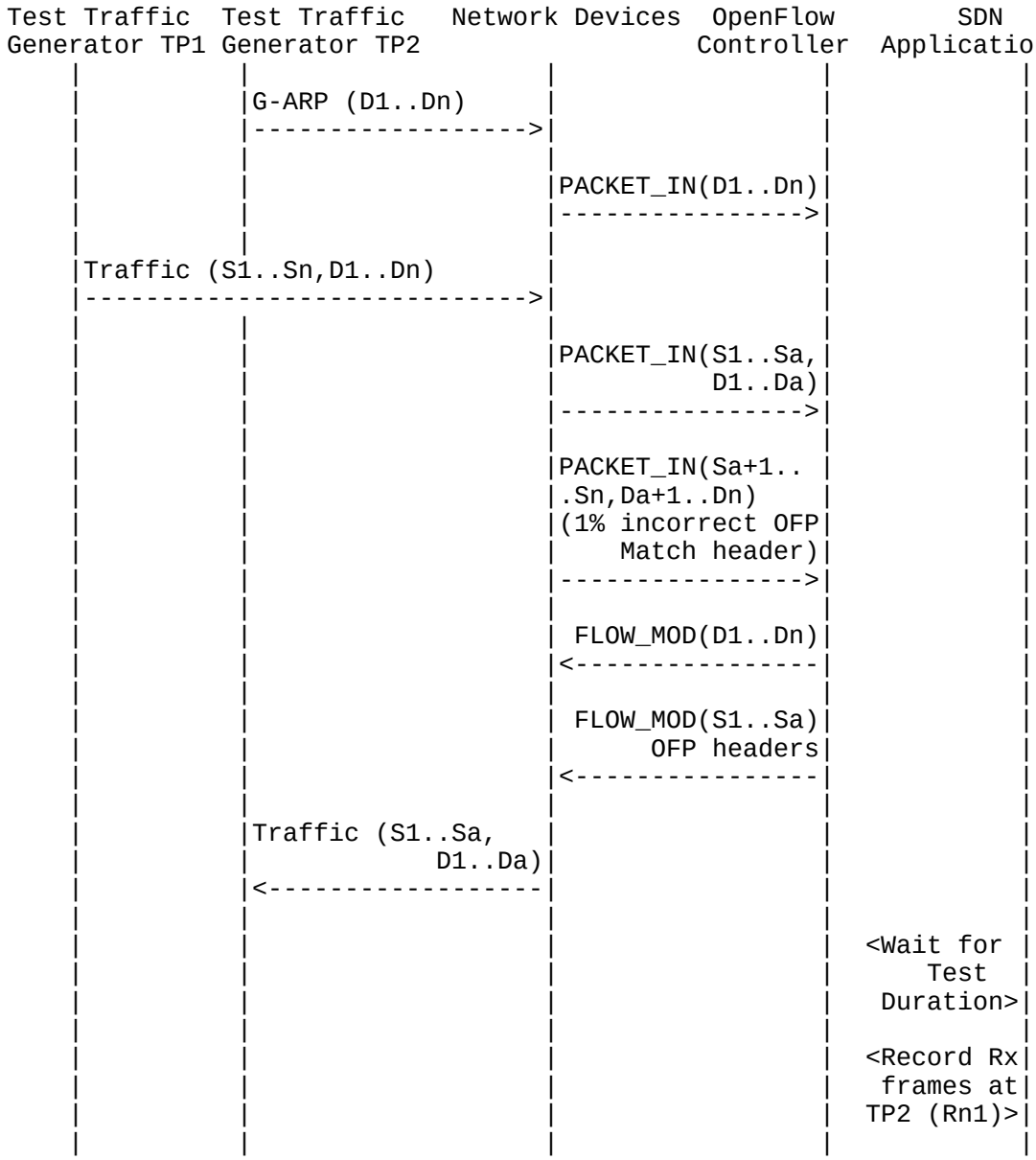
Discussion:

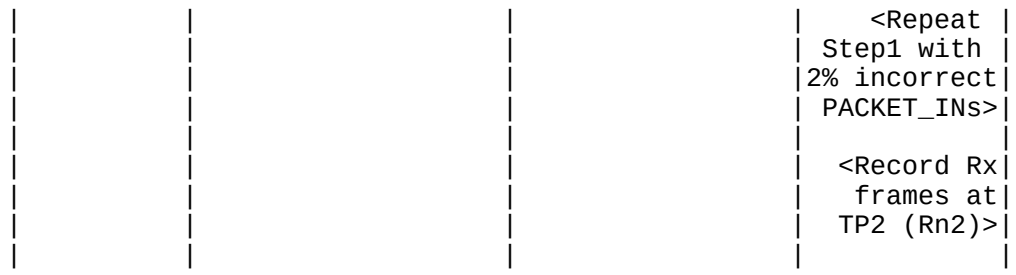
Query the controller forwarding table entries for multiple times until the three consecutive queries return the same value. The last value retrieved from the controller will provide the Forwarding Table Capacity value. The query interval is user configurable. The 5 seconds shown in this example is for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

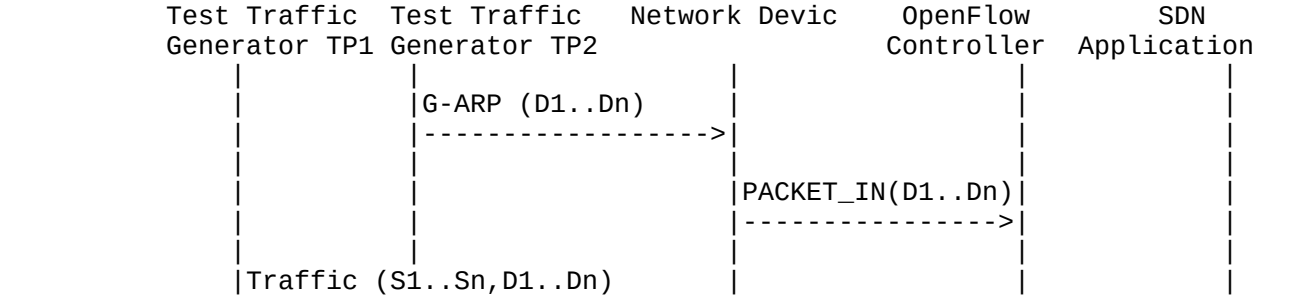
Discussion:

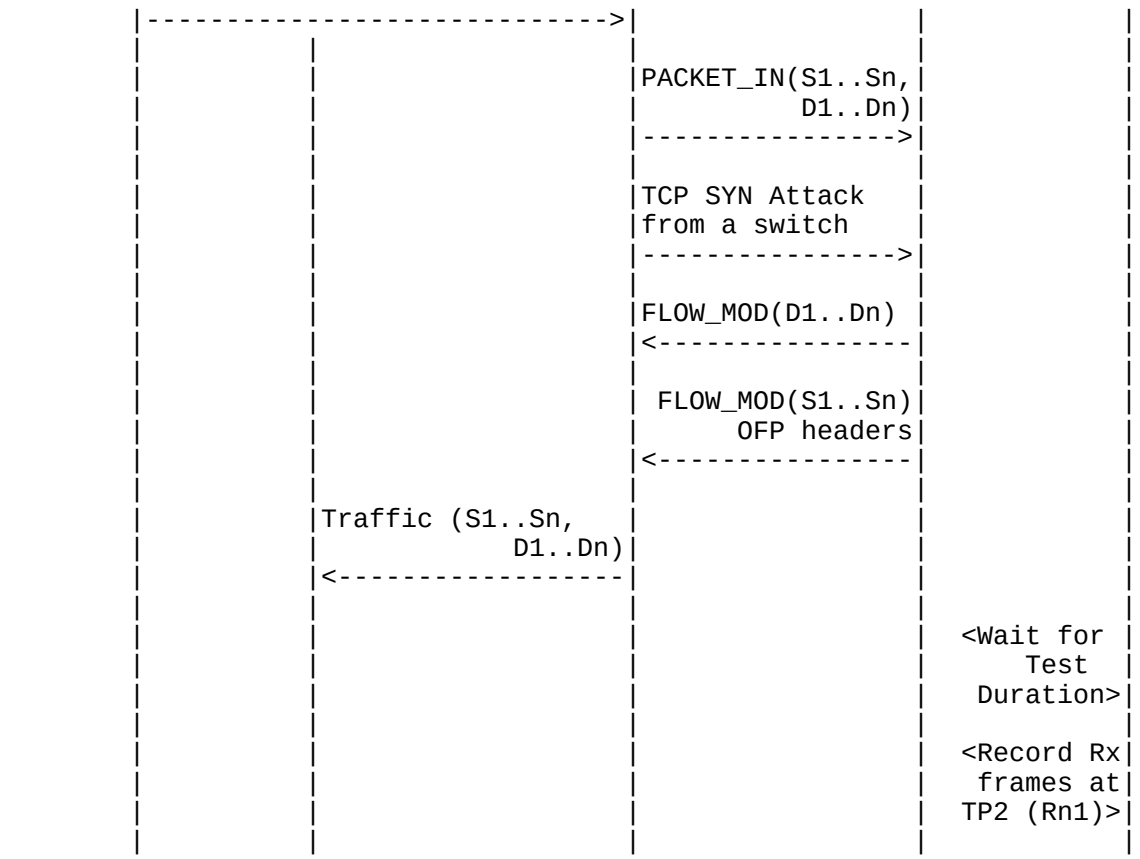
The traffic rate sent towards OpenFlow switch from Test Port 1 should be 1% higher than the Path Programming Rate. Rn1 will provide the Path Provisioning Rate of controller at 1% of incorrect frames handling and Rn2 will provide the Path Provisioning Rate of controller at 2% of incorrect frames handling.

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:

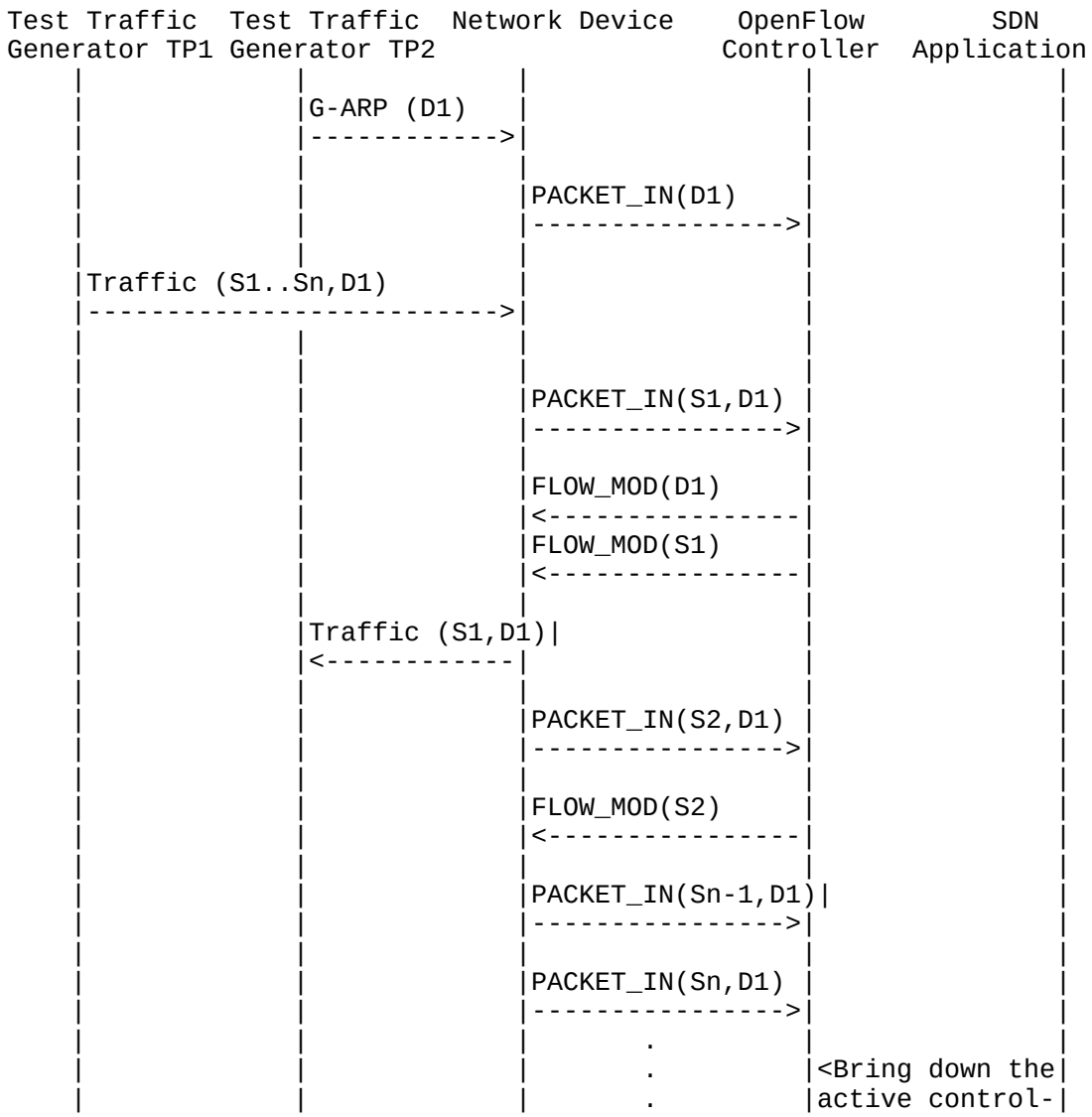
TCP SYN attack should be launched from one of the emulated/simulated OpenFlow Switch. Rn1 provides the Path Programming Rate of controller uponhandling denial of service attack.

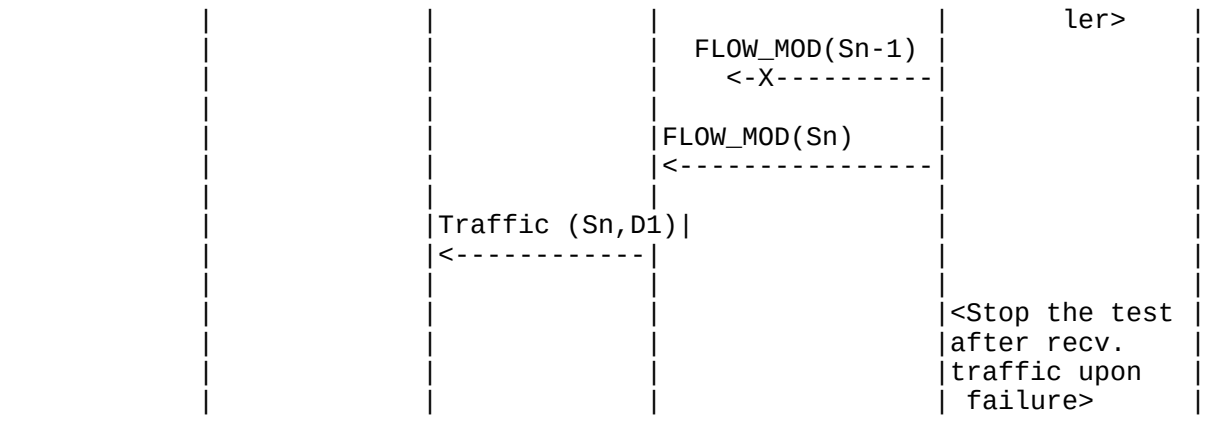
The procedure defined above provides test steps to determine the effect of handling denial of service on Path Programming Rate. Same procedure can be adopted to determine the effects on other performance tests listed in this benchmarking tests.

A.7. Reliability

A.7.1. Controller Failover Time

Procedure:





Legend:

G-ARP: Gratuitous ARP.

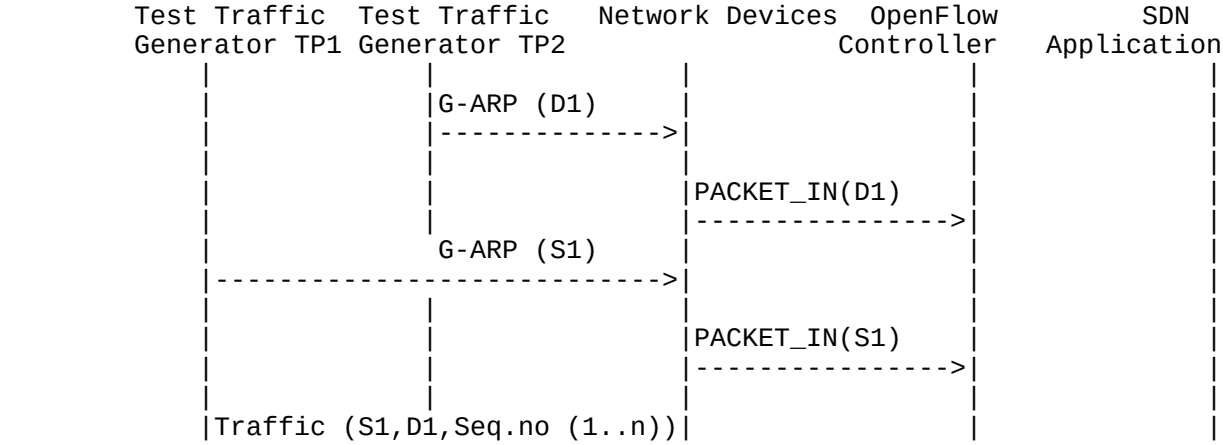
Discussion:

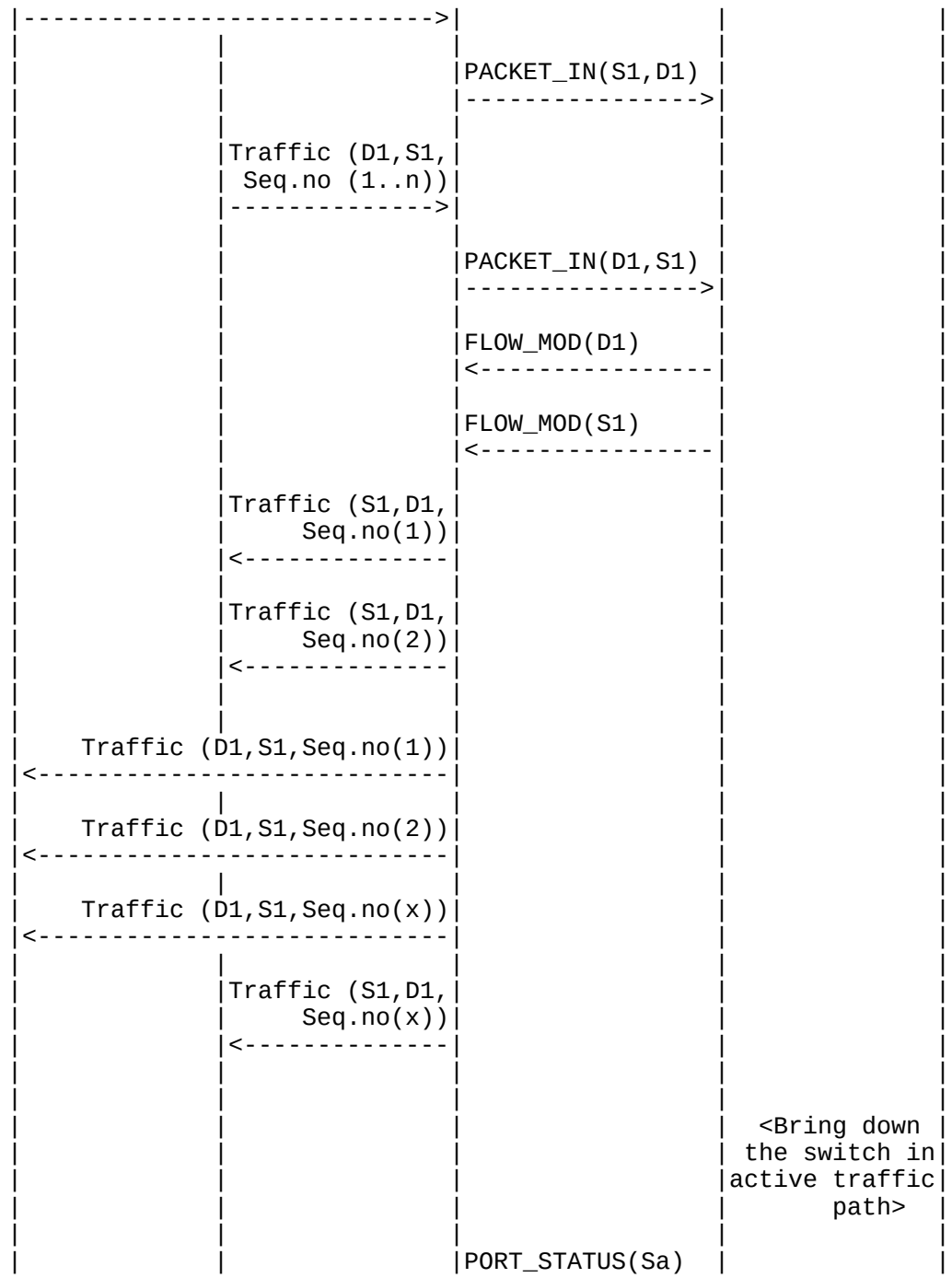
The time difference between the last valid frame received before the traffic loss and the first frame received after the traffic loss will provide the controller failover time.

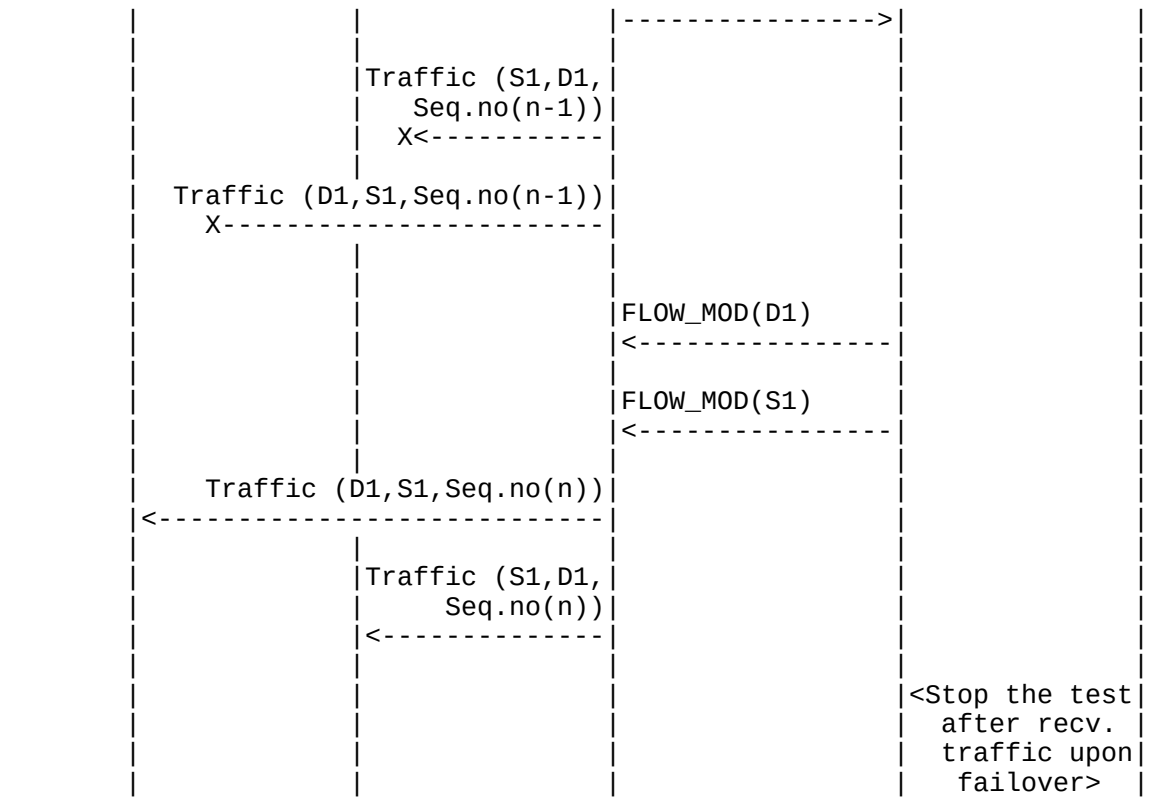
If there is no frame loss during controller failover time, the controller failover time can be deemed negligible.

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: January 8, 2017

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Mark Tassinari
Hewlett-Packard
Vishwas Manral
Nano Sec
Sarah Banks
VSS Monitoring
July 8, 2016

Terminology for Benchmarking SDN Controller Performance
draft-ietf-bmwg-sdn-controller-benchmark-term-02

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 January 8, 2017.

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 RFC 7426 ''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 RFC 4689 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 RFC 7426.

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 RFC 2330.

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. Test Iterations

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. Test Duration

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

Discussion:
Test duration forms the basis for stop criteria for benchmarking tests. Test 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.

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 test 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 maximum number of asynchronous messages that the controller(s) can process, defined as the number of asynchronous messages the controller(s) can process at its Southbound interface between the start of the test and the expiry of given test duration..

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 full connection capacity for the given test duration. This test assumes that the controller will respond to all the received asynchronous messages.

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 test and the expiry of given test 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 test and the expiry of given test 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

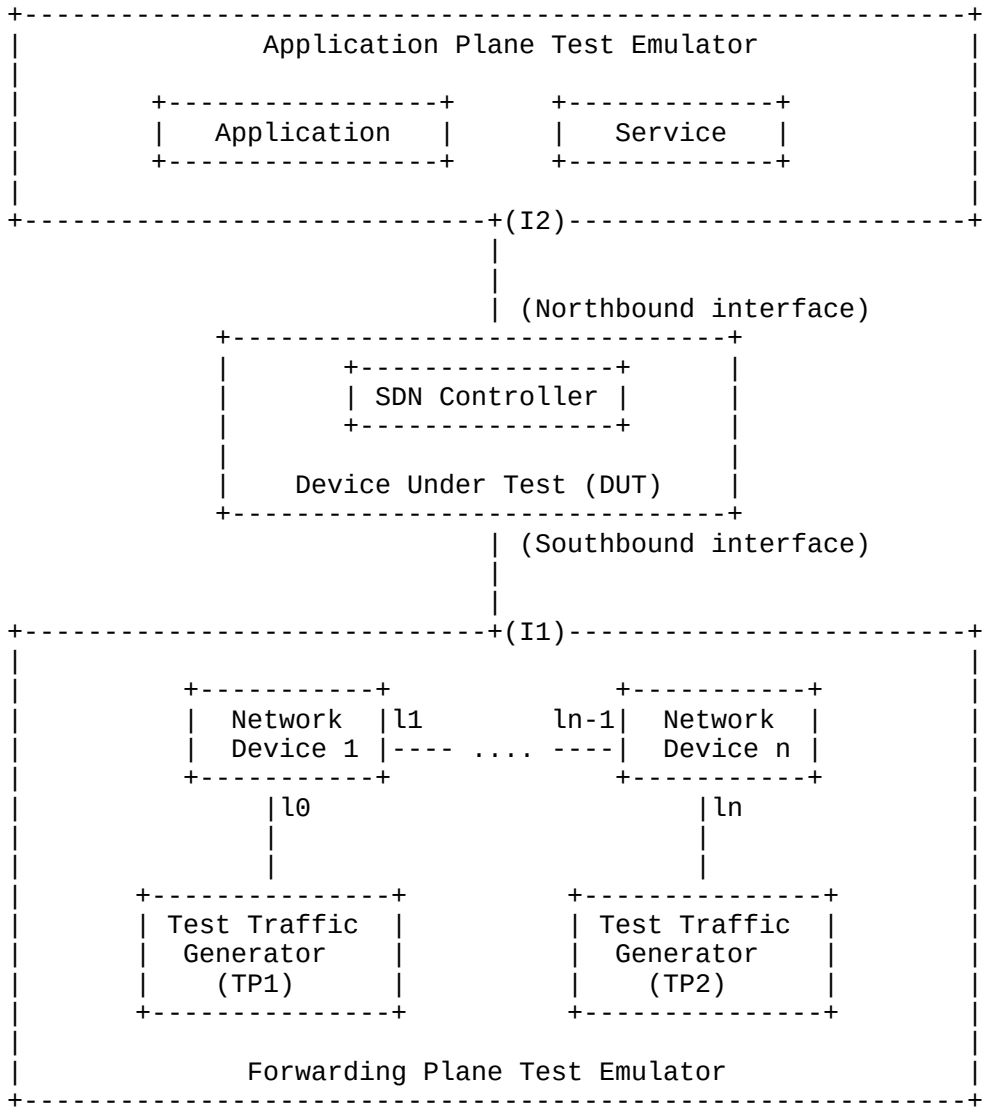


Figure 1

3.2. Test setup - Controller working in Cluster Mode

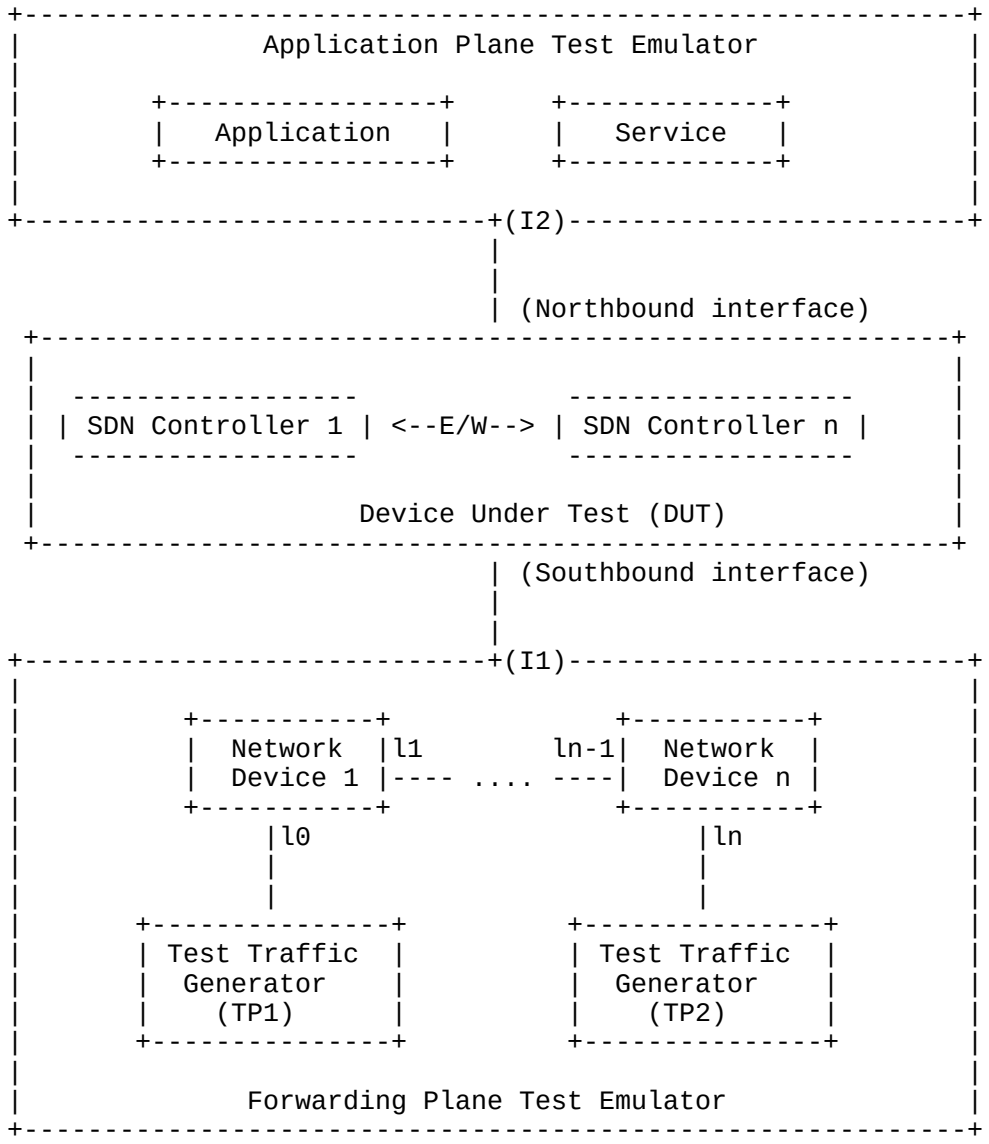


Figure 2

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 Provisioning Rate	1. Network Discovery Size	
Operational	1. Asynchronous Message Processing Rate 2. 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.

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

[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-02 (Work in progress), July 8, 2016

5.2. Informative References

[OpenContrail] Ankur Singla, Bruno Rijsman, "OpenContrail Architecture Documentation", http://opencontrail.org/opencontrail-architecture-documentation

[OpenDaylight] OpenDaylight Controller:Architectural Framework, https://wiki.opendaylight.org/view/OpenDaylight_Controller

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

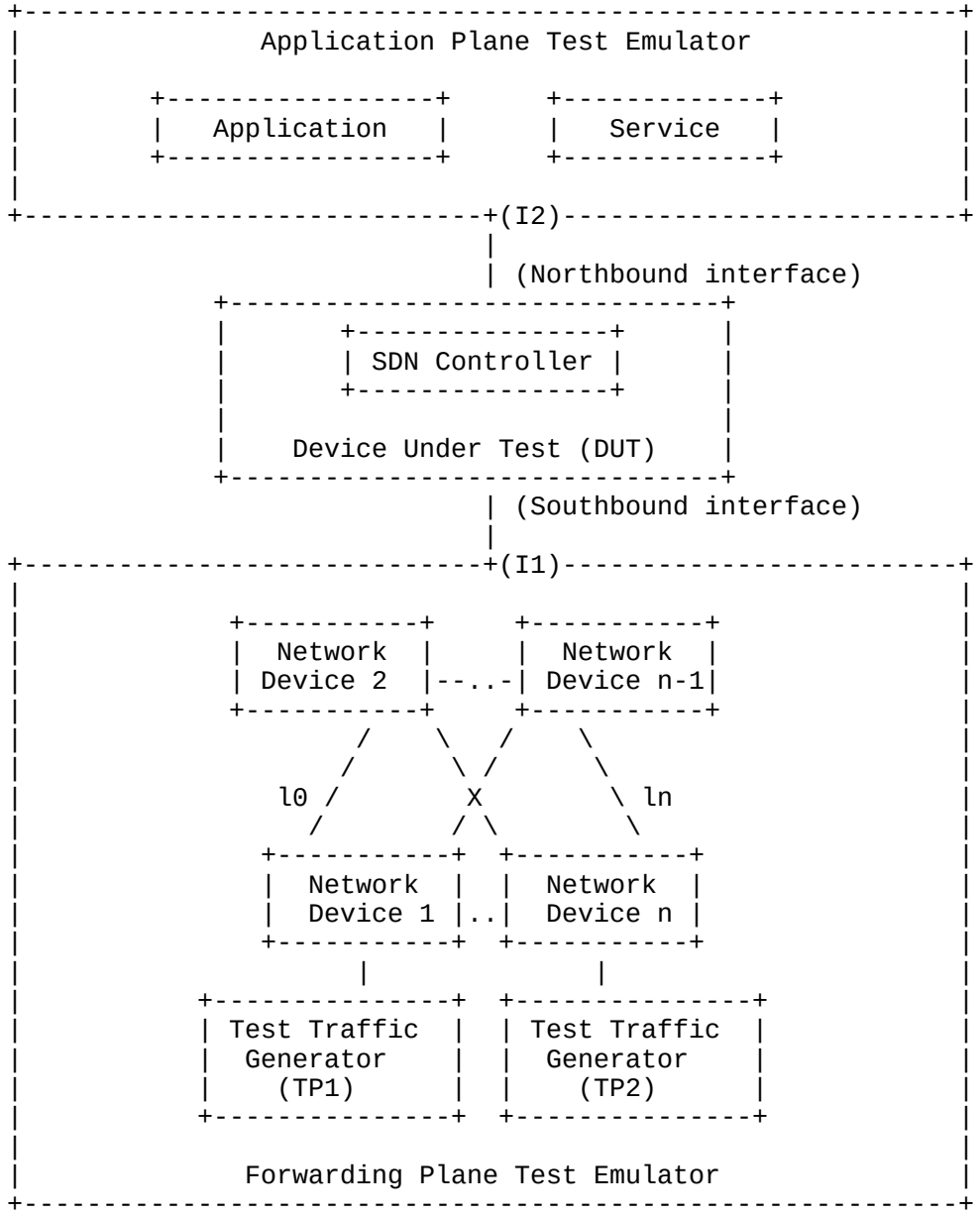


Figure 1

3.2. Test setup - Controller working in Cluster Mode

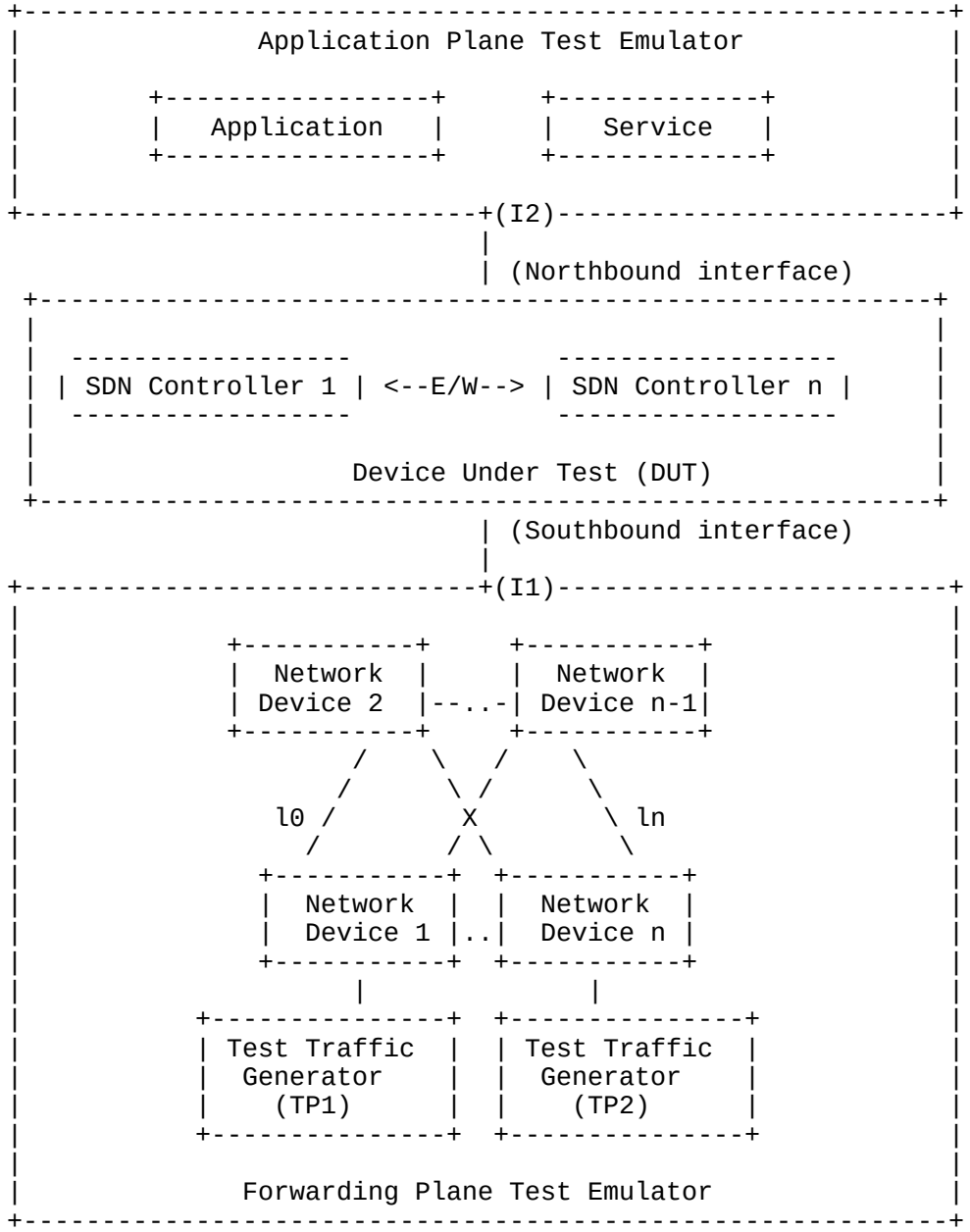


Figure 2

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: September 17, 2017

A. Morton
AT&T Labs

March 16, 2017

Considerations for Benchmarking Virtual Network Functions and Their Infrastructure
draft-ietf-bmwg-virtual-net-05

Abstract

The Benchmarking Methodology Working Group has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions. This memo investigates additional considerations when network functions are virtualized and performed in general purpose hardware.

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 September 17, 2017.

Copyright Notice

Table of Contents

1. Introduction

The Benchmarking Methodology Working Group (BMWG) has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions (or physical network functions, PNFs). The Black-box Benchmarks of Throughput, Latency, Forwarding Rates and others have served our industry for many years. [RFC1242] and [RFC2544] are the cornerstones of the work.

An emerging set of service provider and vendor development goals is to reduce costs while increasing flexibility of network devices, and drastically accelerate their deployment. Network Function Virtualization (NFV) has the promise to achieve these goals, and therefore has garnered much attention. It now seems certain that some network functions will be virtualized following the success of cloud computing and virtual desktops supported by sufficient network

path capacity, performance, and widespread deployment; many of the same techniques will help achieve NFV.

In the context of Virtualized Network Functions (VNF), the supporting Infrastructure requires general-purpose computing systems, storage systems, networking systems, virtualization support systems (such as hypervisors), and management systems for the virtual and physical resources. There will be many potential suppliers of Infrastructure systems and significant flexibility in configuring the systems for best performance. There are also many potential suppliers of VNFs, adding to the combinations possible in this environment. The separation of hardware and software suppliers has a profound implication on benchmarking activities: much more of the internal configuration of the black-box device under test (DUT) must now be specified and reported with the results, to foster both repeatability and comparison testing at a later time.

Consider the following User Story as further background and motivation:

"I'm designing and building my NFV Infrastructure platform. The first steps were easy because I had a small number of categories of VNFs to support and the VNF vendor gave HW recommendations that I followed. Now I need to deploy more VNFs from new vendors, and there are different hardware recommendations. How well will the new VNFs perform on my existing hardware? Which among several new VNFs in a given category are most efficient in terms of capacity they deliver? And, when I operate multiple categories of VNFs (and PNFs) *concurrently* on a hardware platform such that they share resources, what are the new performance limits, and what are the software design choices I can make to optimize my chosen hardware platform? Conversely, what hardware platform upgrades should I pursue to increase the capacity of these concurrently operating VNFs?"

See http://www.etsi.org/technologies-clusters/technologies/nfv for more background, for example, the white papers there may be a useful starting place. The Performance and Portability Best Practices [NFV.PER001] are particularly relevant to BMWG. There are documents available in the Open Area http://docbox.etsi.org/ISG/NFV/Open/Latest_Drafts/ including drafts describing Infrastructure aspects and service quality.

2. Scope

At the time of this writing, BMWG is considering the new topic of Virtual Network Functions and related Infrastructure to ensure that common issues are recognized from the start, using background materials from industry and SDOs (e.g., IETF, ETSI NFV).

This memo investigates additional methodological considerations necessary when benchmarking VNFs instantiated and hosted in general-purpose hardware, using bare metal hypervisors [BareMetal] or other isolation environments such as Linux containers. An essential consideration is benchmarking physical and virtual network functions in the same way when possible, thereby allowing direct comparison. Benchmarking combinations of physical and virtual devices and functions in a System Under Test is another topic of keen interest.

A clearly related goal: the benchmarks for the capacity of a general-purpose platform to host a plurality of VNF instances should be investigated. Existing networking technology benchmarks will also be considered for adaptation to NFV and closely associated technologies.

A non-goal is any overlap with traditional computer benchmark development and their specific metrics (SPECmark suites such as SPECCPU).

A continued non-goal is any form of architecture development related to NFV and associated technologies in BMWG, consistent with all chartered work since BMWG began in 1989.

3. Considerations for Hardware and Testing

This section lists the new considerations which must be addressed to benchmark VNF(s) and their supporting infrastructure. The System Under Test (SUT) is composed of the hardware platform components, the VNFs installed, and many other supporting systems. It is critical to document all aspects of the SUT to foster repeatability.

3.1. Hardware Components

New Hardware components will become part of the test set-up.

1. High volume server platforms (general-purpose, possibly with virtual technology enhancements).

2. Storage systems with large capacity, high speed, and high reliability.

3. Network Interface ports specially designed for efficient service of many virtual NICs.

4. High capacity Ethernet Switches.

The components above are subjects for development of specialized benchmarks which are focused on the special demands of network function deployment.

Labs conducting comparisons of different VNFs may be able to use the same hardware platform over many studies, until the steady march of innovations overtakes their capabilities (as happens with the lab's traffic generation and testing devices today).

3.2. Configuration Parameters

It will be necessary to configure and document the settings for the entire general-purpose platform to ensure repeatability and foster future comparisons, including but clearly not limited-to the following:

- number of server blades (shelf occupation)
- CPUs
- caches
- memory
- storage system
- I/O

as well as configurations that support the devices which host the VNF itself:

- Hypervisor (or other forms of virtual function hosting)
- Virtual Machine (VM)
- Infrastructure Virtual Network (which interconnects Virtual Machines with physical network interfaces, or with each other through virtual switches, for example)

and finally, the VNF itself, with items such as:

- specific function being implemented in VNF
- reserved resources for each function (e.g., CPU pinning and Non-Uniform Memory Access, NUMA node assignment)
- number of VNFs (or sub-VNF components, each with its own VM) in the service function chain (see section 1.1 of [RFC7498] for a definition of service function chain)
- number of physical interfaces and links transited in the service function chain

In the physical device benchmarking context, most of the corresponding infrastructure configuration choices were determined by the vendor. Although the platform itself is now one of the configuration variables, it is important to maintain emphasis on the networking benchmarks and capture the platform variables as input factors.

3.3. Testing Strategies

The concept of characterizing performance at capacity limits may change. For example:

1. It may be more representative of system capacity to characterize the case where Virtual Machines (VM, hosting the VNF) are operating at 50% Utilization, and therefore sharing the "real" processing power across many VMs.

2. Another important case stems from the need for partitioning functions. A noisy neighbor (VM hosting a VNF in an infinite loop) would ideally be isolated and the performance of other VMs would continue according to their specifications.

3. System errors will likely occur as transients, implying a distribution of performance characteristics with a long tail (like latency), leading to the need for longer-term tests of each set of configuration and test parameters.

4. The desire for elasticity and flexibility among network functions will include tests where there is constant flux in the number of VM instances, the resources the VMs require, and the set-up/tear-down of network paths that support VM connectivity. Requests for and instantiation of new VMs, along with Releases for VMs hosting VNFs that are no longer needed would be an normal operational condition. In other words, benchmarking should include scenarios with production life cycle management of VMs and their VNFs and network connectivity in-progress, including VNF scaling up/down operations, as well as static configurations.

5. All physical things can fail, and benchmarking efforts can also examine recovery aided by the virtual architecture with different approaches to resiliency.

6. The sheer number of test conditionas and configuration combinations encourage increased efficiency, including automated testing arrangements, combination sub-sampling through an understanding of inter-relationships, and machine-readable test results.

3.4. Attention to Shared Resources

Since many components of the new NFV Infrastructure are virtual, test set-up design must have prior knowledge of inter-actions/dependencies within the various resource domains in the System Under Test (SUT). For example, a virtual machine performing the role of a traditional tester function such as generating and/or receiving traffic should avoid sharing any SUT resources with the Device Under Test DUT. Otherwise, the results will have unexpected dependencies not encountered in physical device benchmarking.

Note: The term "tester" has traditionally referred to devices dedicated to testing in BMWG literature. In this new context, "tester" additionally refers to functions dedicated to testing, which may be either virtual or physical. "Tester" has never referred to the individuals performing the tests.

The shared-resource aspect of test design remains one of the critical challenges to overcome in a way to produce useful results. Benchmarking set-ups may designate isolated resources for the DUT and other critical support components (such as the host/kernel) as the first baseline step, and add other loading processes. The added complexity of each set-up leads to shared-resource testing scenarios, where the characteristics of the competing load (in terms of memory, storage, and CPU utilization) will directly affect the benchmarking results (and variability of the results), but the results should reconcile with the baseline.

The physical test device remains a solid foundation to compare with results using combinations of physical and virtual test functions, or results using only virtual testers when necessary to assess virtual interfaces and other virtual functions.

4. Benchmarking Considerations

This section discusses considerations related to Benchmarks applicable to VNFs and their associated technologies.

4.1. Comparison with Physical Network Functions

In order to compare the performance of VNFs and system implementations with their physical counterparts, identical benchmarks must be used. Since BMWG has already developed specifications for many network functions, there will be re-use of existing benchmarks through references, while allowing for the possibility of benchmark curation during development of new methodologies. Consideration should be given to quantifying the number of parallel VNFs required to achieve comparable scale/capacity

with a given physical device, or whether some limit of scale was reached before the VNFs could achieve the comparable level. Again, implementation based-on different hypervisors or other virtual function hosting remain as critical factors in performance assessment.

4.2. Continued Emphasis on Black-Box Benchmarks

When the network functions under test are based on Open Source code, there may be a tendency to rely on internal measurements to some extent, especially when the externally-observable phenomena only support an inference of internal events (such as routing protocol convergence observed in the dataplane). Examples include CPU/Core utilization, Network utilization, Storage utilization, and Memory Comitted/used. These "white-box" metrics provide one view of the resource footprint of a VNF. Note: The resource utilization metrics do not easily match the 3x4 Matrix, described in Section 4.4 below.

However, external observations remain essential as the basis for Benchmarks. Internal observations with fixed specification and interpretation may be provided in parallel (as auxilliary metrics), to assist the development of operations procedures when the technology is deployed, for example. Internal metrics and measurements from Open Source implementations may be the only direct source of performance results in a desired dimension, but corroborating external observations are still required to assure the integrity of measurement discipline was maintained for all reported results.

A related aspect of benchmark development is where the scope includes multiple approaches to a common function under the same benchmark. For example, there are many ways to arrange for activation of a network path between interface points and the activation times can be compared if the start-to-stop activation interval has a generic and unambiguous definition. Thus, generic benchmark definitions are preferred over technology/protocol specific definitions where possible.

4.3. New Benchmarks and Related Metrics

There will be new classes of benchmarks needed for network design and assistance when developing operational practices (possibly automated management and orchestration of deployment scale). Examples follow in the paragraphs below, many of which are prompted by the goals of increased elasticity and flexibility of the network functions, along with accelerated deployment times.

- Time to deploy VNFs: In cases where the general-purpose hardware is already deployed and ready for service, it is valuable to know the response time when a management system is tasked with "standing-up" 100's of virtual machines and the VNFs they will host.

- Time to migrate VNFs: In cases where a rack or shelf of hardware must be removed from active service, it is valuable to know the response time when a management system is tasked with "migrating" some number of virtual machines and the VNFs they currently host to alternate hardware that will remain in-service.

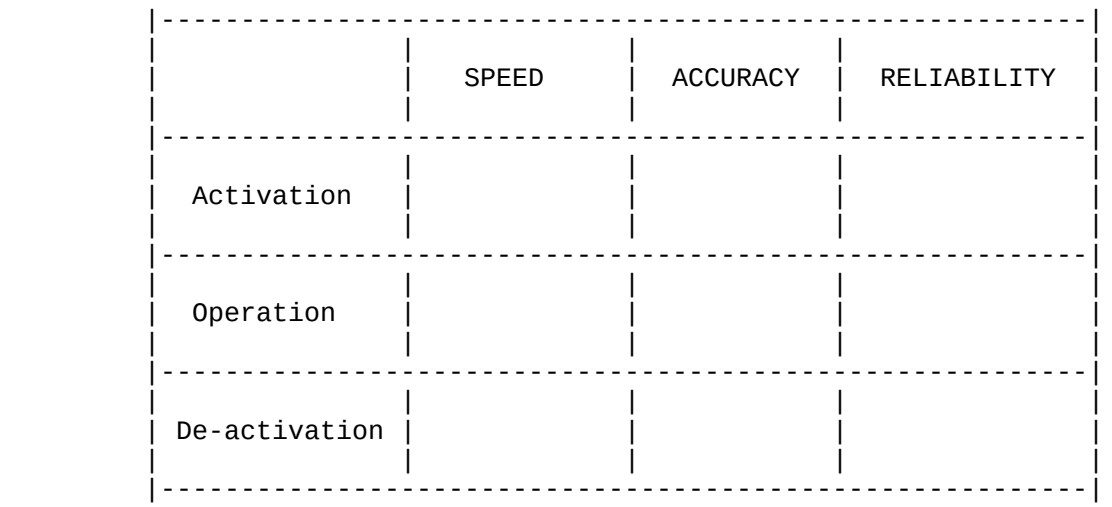
- Time to create a virtual network in the general-purpose infrastructure: This is a somewhat simplified version of existing benchmarks for convergence time, in that the process is initiated by a request from (centralized or distributed) control, rather than inferred from network events (link failure). The successful response time would remain dependent on dataplane observations to confirm that the network is ready to perform.

- Effect of verification measurements on performance: A complete VNF, or something as simple as a new policy to implement in a VNF, is implemented. The action to verify instantiation of the VNF or policy could affect performance during normal operation.

Also, it appears to be valuable to measure traditional packet transfer performance metrics during the assessment of traditional and new benchmarks, including metrics that may be used to support service engineering such as the Spatial Composition metrics found in [RFC6049]. Examples include Mean one-way delay in section 4.1 of [RFC6049], Packet Delay Variation (PDV) in [RFC5481], and Packet Reordering [RFC4737] [RFC4689].

4.4. Assessment of Benchmark Coverage

It can be useful to organize benchmarks according to their applicable life cycle stage and the performance criteria they were designed to assess. The table below (derived from [X3.102]) provides a way to organize benchmarks such that there is a clear indication of coverage for the intersection of life cycle stages and performance criteria.



For example, the "Time to deploy VNFs" benchmark described above would be placed in the intersection of Activation and Speed, making it clear that there are other potential performance criteria to benchmark, such as the "percentage of unsuccessful VM/VNF stand-ups" in a set of 100 attempts. This example emphasizes that the Activation and De-activation life cycle stages are key areas for NFV and related infrastructure, and encourage expansion beyond traditional benchmarks for normal operation. Thus, reviewing the benchmark coverage using this table (sometimes called the 3x3 matrix) can be a worthwhile exercise in BMWG.

In one of the first applications of the 3x3 matrix in BMWG [I-D.ietf-bmwg-sdn-controller-benchmark-meth], we discovered that metrics on measured size, capacity, or scale do not easily match one of the three columns above. Following discussion, this was resolved in two ways:

- Add a column, Scale, for use when categorizing and assessing the coverage of benchmarks (without measured results). Examples of this use are found in [I-D.ietf-bmwg-sdn-controller-benchmark-meth] and [I-D.vsperf-bmwg-vswitch-opnfv]. This is the 3x4 Matrix.

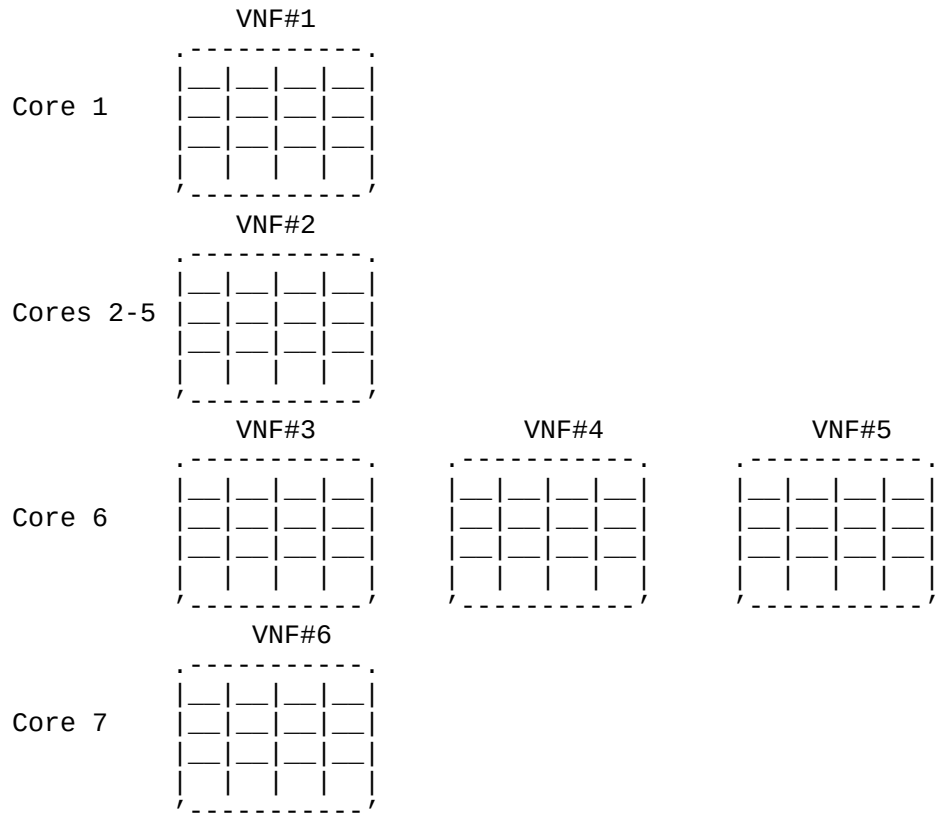
- If using the matrix to report results in an organized way, keep size, capacity, and scale metrics separate from the 3x3 matrix and incorporate them in the report with other qualifications of the results.

Note: The resource utilization (e.g., CPU) metrics do not fit in the Matrix. They are not benchmarks, and omitting them confirms their

status as auxilliary metrics. Resource assignments are configuration parameters, and these are reported seperately.

This approach encourages use of the 3x3 matrix to organize reports of results, where the capacity at which the various metrics were measured could be included in the title of the matrix (and results for multiple capacities would result in separate 3x3 matrices, if there were sufficient measurements/results to organize in that way).

For example, results for each VM and VNF could appear in the 3x3 matrix, organized to illustrate resource occupation (CPU Cores) in a particular physical computing system, as shown below.



The combination of tables above could be built incrementally, beginning with VNF#1 and one Core, then adding VNFs according to their supporting core assignments. X-Y plots of critical benchmarks would also provide insight to the effect of increased HW utilization. All VNFs might be of the same type, or to match a production environment there could be VNFs of multiple types and categories. In

this figure, VNFs #3-#5 are assumed to require small CPU resources, while VNF#2 requires 4 cores to perform its function.

4.5. Power Consumption

Although there is incomplete work to benchmark physical network function power consumption in a meaningful way, the desire to measure the physical infrastructure supporting the virtual functions only adds to the need. Both maximum power consumption and dynamic power consumption (with varying load) would be useful. The IPMI standard [IPMI2.0] has been implemented by many manufacturers, and supports measurement of instantaneous energy consumption.

To assess the instantaneous energy consumption of virtual resources, it may be possible to estimate the value using an overall metric based on utilization readings, according to [I-D.krishnan-nfvrg-policy-based-rm-nfviaas].

5. 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.

6. IANA Considerations

No IANA Action is requested at this time.

7. Acknowledgements

The author acknowledges an encouraging conversation on this topic with Mukhtiar Shaikh and Ramki Krishnan in November 2013. Bhavani Parise and Ilya Varlashkin have provided useful suggestions to expand

these considerations. Bhuvaneswaran Vengainathan has already tried the 3x3 matrix with SDN controller draft, and contributed to many discussions. Scott Bradner quickly pointed out shared resource dependencies in an early vSwitch measurement proposal, and the topic was included here as a key consideration. Further development was encouraged by Barry Constantine's comments following the IETF-92 BMWG session: the session itself was an affirmation for this memo. There have been many interesting contributions from Maryam Tahhan, Marius Georgescu, Jacob Rapp, Saurabh Chattopadhyay, and others.

8. Version history

(This section should be removed by the RFC Editor.)

version 05: Address IESG & Last Call Comments (editorial)

Version 03 & 04: address mininal comments and few WGLC comments

Version 02:

New version history section.

Added Memory in section 3.2, configuration.

Updated ACKs and References.

Version 01:

Addressed Ramki Krishnan's comments on section 4.5, power, see that section (7/27 message to the list). Addressed Saurabh Chattopadhyay's 7/24 comments on VNF resources and other resource conditions and their effect on benchmarking, see section 3.4. Addressed Marius Georgescu's 7/17 comments on the list (sections 4.3 and 4.4).

AND, comments from the extended discussion during IETF-93 BMWG session:

Section 4.2: VNF footprint and auxilliary metrics (Maryam Tahhan),
Section 4.3: Verification affect metrics (Ramki Krishnan);
Section 4.4: Auxilliary metrics in the Matrix (Maryam Tahhan, Scott Bradner, others)

9. References

9.1. Normative References

[NFV.PER001]
"Network Function Virtualization: Performance and Portability Best Practices", Group Specification ETSI GS NFV-PER 001 V1.1.1 (2014-06), June 2014.

[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>.

[RFC4689] Poretsky, S., Perser, J., Erramilli, S., and S. Khurana, "Terminology for Benchmarking Network-layer Traffic Control Mechanisms", RFC 4689, DOI 10.17487/RFC4689, October 2006, <http://www.rfc-editor.org/info/rfc4689>.

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, DOI 10.17487/RFC4737, November 2006, <http://www.rfc-editor.org/info/rfc4737>.

[RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <http://www.rfc-editor.org/info/rfc7498>.

9.2. Informative References

[BareMetal]
Popek, Gerald J.; Goldberg, Robert P. , , "Formal requirements for virtualizable third generation architectures". Communications of the ACM. 17 (7): 412-421. doi:10.1145/361011.361073.", 1974.

[I-D.ietf-bmwg-sdn-controller-benchmark-meth]
Vengainathan, B., Basil, A., Tassinari, M., Manral, V., and S. Banks, "Benchmarking Methodology for SDN Controller Performance", draft-ietf-bmwg-sdn-controller-benchmark-meth-03 (work in progress), January 2017.

[I-D.krishnan-nfvrg-policy-based-rm-nfviaas]
Krishnan, R., Figueira, N., Krishnaswamy, D., Lopez, D., Wright, S., Hinrichs, T., Krishnaswamy, R., and A. Yerra, "NFVIaaS Architectural Framework for Policy Based Resource Placement and Scheduling", draft-krishnan-nfvrg-policy-based-rm-nfviaas-06 (work in progress), March 2016.

[I-D.vsperf-bmwg-vswitch-opnfv]
Tahhan, M., O'Mahony, B., and A. Morton, "Benchmarking Virtual Switches in OPNFV", draft-vsperf-bmwg-vswitch-opnfv-02 (work in progress), March 2016.

[IPMI2.0] "Intelligent Platform Management Interface, v2.0 with latest Errata", http://www.intel.com/content/www/us/en/servers/ipmi/ipmi-intelligent-platform-mgt-interface-spec-2nd-gen-v2-0-spec-update.html, April 2015.

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

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <http://www.rfc-editor.org/info/rfc5481>.

[RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, DOI 10.17487/RFC6049, January 2011, <http://www.rfc-editor.org/info/rfc6049>.

[X3.102] ANSI X3.102, , "ANSI Standard on Data Communications, User-Oriented Data Communications Framework", 1983.

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
URI: http://home.comcast.net/~acmacm/

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 17, 2017

M. Tahhan
B. O'Mahony
Intel
A. Morton
AT&T Labs

October 14, 2016

Benchmarking Virtual Switches in OPNFV
draft-ietf-bmwg-vswitch-opnfv-01

Abstract

This memo describes the progress of the Open Platform for NFV (OPNFV) project on virtual switch performance "VSWITCHPERF". This project intends to build on the current and completed work of the Benchmarking Methodology Working Group in IETF, by referencing existing literature. The Benchmarking Methodology Working Group has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions. Therefore, this memo begins to describe the additional considerations when virtual switches are implemented in general-purpose hardware. The expanded tests and benchmarks are also influenced by the OPNFV mission to support virtualization of the "telco" infrastructure.

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 April 17, 2017.

Table of Contents

1. Introduction

Benchmarking Methodology Working Group (BMWG) has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions. The Black-box Benchmarks of Throughput, Latency, Forwarding Rates and others have served our industry for many years. Now, Network Function Virtualization (NFV) has the goal to transform how internetwork functions are implemented, and therefore has garnered much attention.

This memo summarizes the progress of the Open Platform for NFV (OPNFV) project on virtual switch performance characterization, "VSWITCHPERF", through the Brahmaputra (second) release [BrahRel]. This project intends to build on the current and completed work of the Benchmarking Methodology Working Group in IETF, by referencing existing literature. For example, currently the most often referenced RFC is [RFC2544] (which depends on [RFC1242]) and foundation of the benchmarking work in OPNFV is common and strong.

See https://wiki.opnfv.org/characterize_vswitch_performance_for_telco_nfv_use_cases for more background, and the OPNFV website for general information: https://www.opnfv.org/

The authors note that OPNFV distinguishes itself from other open source compute and networking projects through its emphasis on existing "telco" services as opposed to cloud-computing. There are many ways in which telco requirements have different emphasis on performance dimensions when compared to cloud computing: support for and transfer of isochronous media streams is one example.

Note also that the move to NFV Infrastructure has resulted in many new benchmarking initiatives across the industry. The authors are currently doing their best to maintain alignment with many other projects, and this Internet Draft is one part of the efforts. We acknowledge the early work in [I-D.huang-bmwg-virtual-network-performance], and useful discussion with the authors.

2. Scope

The primary purpose and scope of the memo is to inform the industry of work-in-progress that builds on the body of extensive BMWG literature and experience, and describe the extensions needed for benchmarking virtual switches. Inital feedback indicates that many of these extensions may be applicable beyond the current scope (to hardware switches in the NFV Infrastructure and to virtual routers, for example). Additionally, this memo serves as a vehicle to include

more detail and commentary from BMWG and other Open Source communities, under BMWG's chartered work to characterize the NFV Infrastructure (a virtual switch is an important aspect of that infrastructure).

The benchmarking covered in this memo should be applicable to many types of vswitches, and remain vswitch-agnostic to great degree. There has been no attempt to track and test all features of any specific vswitch implementation.

3. Benchmarking Considerations

This section highlights some specific considerations (from [I-D.ietf-bmwg-virtual-net])related to Benchmarks for virtual switches. The OPNFV project is sharing its present view on these areas, as they develop their specifications in the Level Test Design (LTD) document.

3.1. Comparison with Physical Network Functions

To compare the performance of virtual designs and implementations with their physical counterparts, identical benchmarks are needed. BMWG has developed specifications for many network functions this memo re-uses existing benchmarks through references, and expands them during development of new methods. A key configuration aspect is the number of parallel cores required to achieve comparable performance with a given physical device, or whether some limit of scale was reached before the cores could achieve the comparable level.

It's unlikely that the virtual switch will be the only application running on the SUT, so CPU utilization, Cache utilization, and Memory footprint should also be recorded for the virtual implementations of internetworking functions.

3.2. Continued Emphasis on Black-Box Benchmarks

External observations remain essential as the basis for Benchmarks. Internal observations with fixed specification and interpretation will be provided in parallel to assist the development of operations procedures when the technology is deployed.

3.3. New Configuration Parameters

A key consideration when conducting any sort of benchmark is trying to ensure the consistency and repeatability of test results. When benchmarking the performance of a vSwitch there are many factors that can affect the consistency of results, one key factor is matching the various hardware and software details of the SUT. This section lists

some of the many new parameters which this project believes are critical to report in order to achieve repeatability.

Hardware details including:

- Platform details
- Processor details
- Memory information (type and size)
- Number of enabled cores
- Number of cores used for the test
- Number of physical NICs, as well as their details (manufacturer, versions, type and the PCI slot they are plugged into)
- NIC interrupt configuration
- BIOS version, release date and any configurations that were modified
- CPU microcode level
- Memory DIMM configurations (quad rank performance may not be the same as dual rank) in size, freq and slot locations
- PCI configuration parameters (payload size, early ack option...)
- Power management at all levels (ACPI sleep states, processor package, OS...)

Software details including:

- OS parameters and behavior (text vs graphical no one typing at the console on one system)
- OS version (for host and VNF)
- Kernel version (for host and VNF)
- GRUB boot parameters (for host and VNF)
- Hypervisor details (Type and version)
- Selected vSwitch, version number or commit id used

- vSwitch launch command line if it has been parameterised
- Memory allocation to the vSwitch
- which NUMA node it is using, and how many memory channels
- DPDK or any other SW dependency version number or commit id used
- Memory allocation to a VM - if it’s from Hugpages/elsewhere
- VM storage type: snapshot/independent persistent/independent non-persistent
- Number of VMs
- Number of Virtual NICs (vNICs), versions, type and driver
- Number of virtual CPUs and their core affinity on the host
- Number vNIC interrupt configuration
- Thread affinitization for the applications (including the vSwitch itself) on the host
- Details of Resource isolation, such as CPUs designated for Host/ Kernel (isolcpu) and CPUs designated for specific processes (taskset). - Test duration. - Number of flows.

Test Traffic Information:

- Traffic type - UDP, TCP, IMIX / Other
- Packet Sizes
- Deployment Scenario

3.4. Flow classification

Virtual switches group packets into flows by processing and matching particular packet or frame header information, or by matching packets based on the input ports. Thus a flow can be thought of a sequence of packets that have the same set of header field values (5-tuple) or have arrived on the same port. Performance results can vary based on the parameters the vSwitch uses to match for a flow. The recommended flow classification parameters for any vSwitch performance tests are: the input port, the source IP address, the destination IP address and the Ethernet protocol type field. It is essential to increase the flow timeout time on a vSwitch before conducting any performance

tests that do not measure the flow setup time. Normally the first packet of a particular stream will install the flow in the virtual switch which adds an additional latency, subsequent packets of the same flow are not subject to this latency if the flow is already installed on the vSwitch.

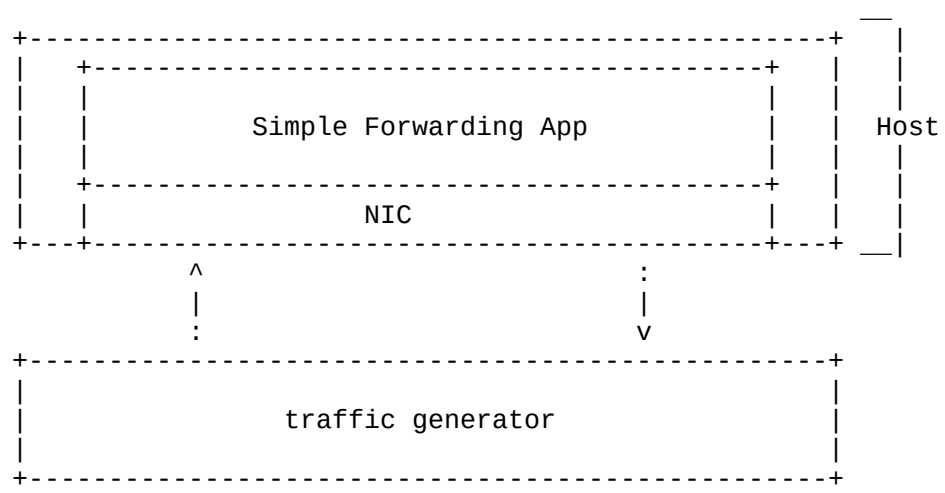
3.5. Benchmarks using Baselines with Resource Isolation

This outline describes measurement of baseline with isolated resources at a high level, which is the intended approach at this time.

1. Baselines:

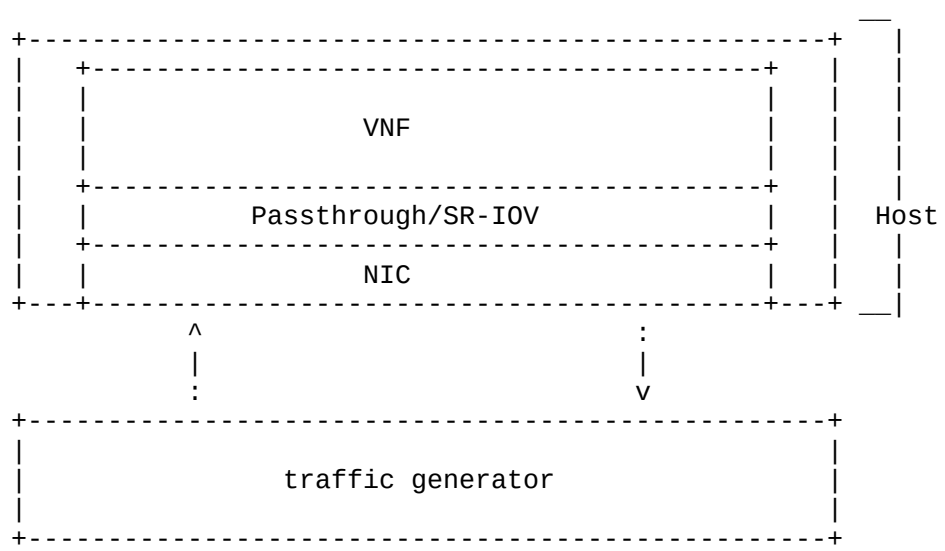
   * Optional: Benchmark platform forwarding capability without a vswitch or VNF for at least 72 hours (serves as a means of platform validation and a means to obtain the base performance for the platform in terms of its maximum forwarding rate and latency).

Benchmark platform forwarding capability



   * Benchmark VNF forwarding capability with direct connectivity (vSwitch bypass, e.g., SR/IOV) for at least 72 hours (serves as a means of VNF validation and a means to obtain the base performance for the VNF in terms of its maximum forwarding rate and latency). The metrics gathered from this test will serve as a key comparison point for vSwitch bypass technologies performance and vSwitch performance.

Benchmark VNF forwarding capability



* Benchmarking with isolated resources alone, with other resources (both HW&SW) disabled Example, vSw and VM are SUT

* Benchmarking with isolated resources alone, leaving some resources unused

* Benchmark with isolated resources and all resources occupied

2. Next Steps

* Limited sharing

* Production scenarios

* Stressful scenarios

4. VSWITCHPERF Specification Summary

The overall specification in preparation is referred to as a Level Test Design (LTD) document, which will contain a suite of performance tests. The base performance tests in the LTD are based on the pre-existing specifications developed by BMWG to test the performance of physical switches. These specifications include:

o [RFC2544] Benchmarking Methodology for Network Interconnect Devices

- [RFC2889] Benchmarking Methodology for LAN Switching
- [RFC6201] Device Reset Characterization
- [RFC5481] Packet Delay Variation Applicability Statement

Some of the above/newer RFCs are being applied in benchmarking for the first time, and represent a development challenge for test equipment developers. Fortunately, many members of the testing system community have engaged on the VSPERF project, including an open source test system.

In addition to this, the LTD also re-uses the terminology defined by:

- [RFC2285] Benchmarking Terminology for LAN Switching Devices
- [RFC5481] Packet Delay Variation Applicability Statement

Specifications to be included in future updates of the LTD include:

- [RFC3918] Methodology for IP Multicast Benchmarking
- [RFC4737] Packet Reordering Metrics

As one might expect, the most fundamental internetworking characteristics of Throughput and Latency remain important when the switch is virtualized, and these benchmarks figure prominently in the specification.

When considering characteristics important to "telco" network functions, we must begin to consider additional performance metrics. In this case, the project specifications have referenced metrics from the IETF IP Performance Metrics (IPPM) literature. This means that the [RFC2544] test of Latency is replaced by measurement of a metric derived from IPPM’s [RFC2679], where a set of statistical summaries will be provided (mean, max, min, etc.). Further metrics planned to be benchmarked include packet delay variation as defined by [RFC5481] , reordering, burst behaviour, DUT availability, DUT capacity and packet loss in long term testing at Throughput level, where some low-level of background loss may be present and characterized.

Tests have been (or will be) designed to collect the metrics below:

- Throughput Tests to measure the maximum forwarding rate (in frames per second or fps) and bit rate (in Mbps) for a constant load (as defined by [RFC1242]) without traffic loss.

- Packet and Frame Delay Distribution Tests to measure average, min and max packet and frame delay for constant loads.

- Packet Delay Tests to understand latency distribution for different packet sizes and over an extended test run to uncover outliers.

- Scalability Tests to understand how the virtual switch performs as the number of flows, active ports, complexity of the forwarding logic’s configuration... it has to deal with increases.

- Stream Performance Tests (TCP, UDP) to measure bulk data transfer performance, i.e. how fast systems can send and receive data through the switch.

- Control Path and Datapath Coupling Tests, to understand how closely coupled the datapath and the control path are as well as the effect of this coupling on the performance of the DUT (example: delay of the initial packet of a flow).

- CPU and Memory Consumption Tests to understand the virtual switch’s footprint on the system, usually conducted as auxiliary measurements with benchmarks above. They include: CPU utilization, Cache utilization and Memory footprint.

- The so-called "Soak" tests, where the selected test is conducted over a long period of time (with an ideal duration of 24 hours, but only long enough to determine that stability issues exist when found; there is no requirement to continue a test when a DUT exhibits instability over time). The key performance characteristics and benchmarks for a DUT are determined (using short duration tests) prior to conducting soak tests. The purpose of soak tests is to capture transient changes in performance which may occur due to infrequent processes, memory leaks, or the low probability coincidence of two or more processes. The stability of the DUT is the paramount consideration, so performance must be evaluated periodically during continuous testing, and this results in use of [RFC2889] Frame Rate metrics instead of [RFC2544] Throughput (which requires stopping traffic to allow time for all traffic to exit internal queues), for example.

Future/planned test specs include:

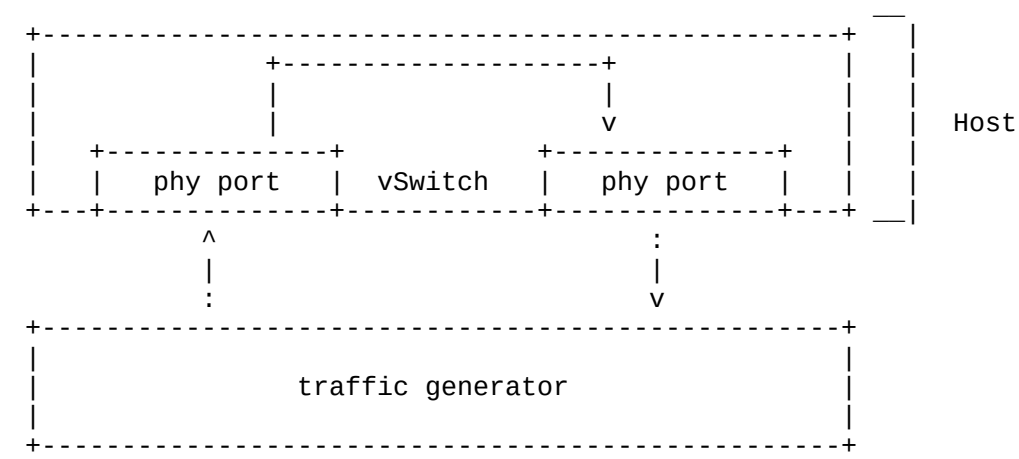
- Request/Response Performance Tests (TCP, UDP) which measure the transaction rate through the switch.

- Noisy Neighbour Tests, to understand the effects of resource sharing on the performance of a virtual switch.

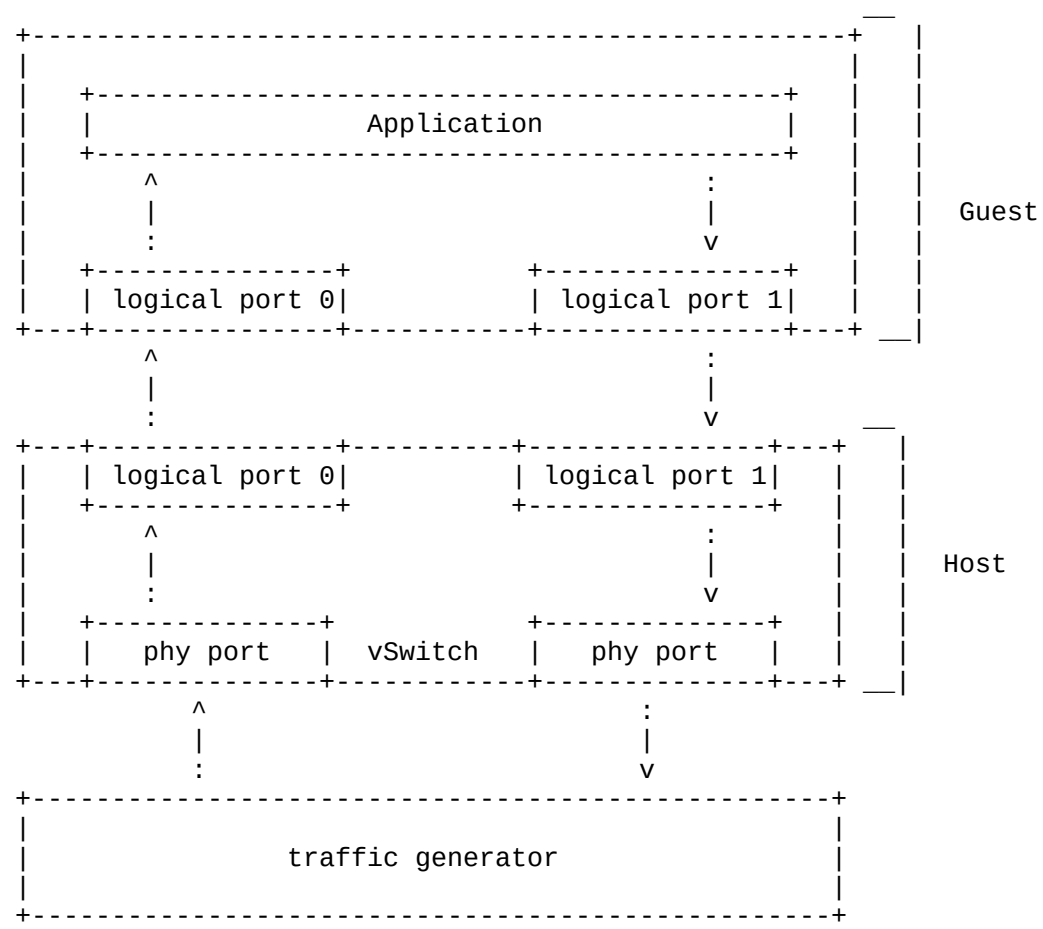
o Tests derived from examination of ETSI NFV Draft GS IFA003 requirements [IFA003] on characterization of acceleration technologies applied to vswitches.

The flexibility of deployment of a virtual switch within a network means that the BMWG IETF existing literature needs to be used to characterize the performance of a switch in various deployment scenarios. The deployment scenarios under consideration include:

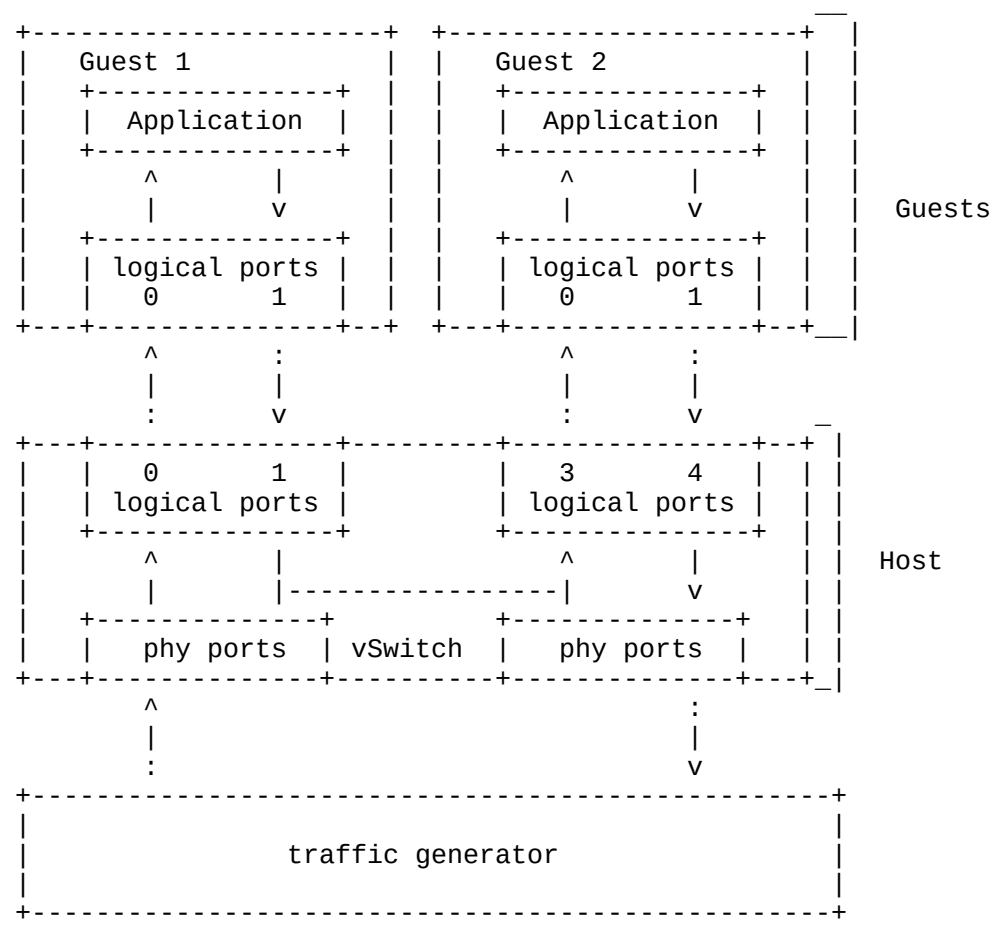
Physical port to virtual switch to physical port



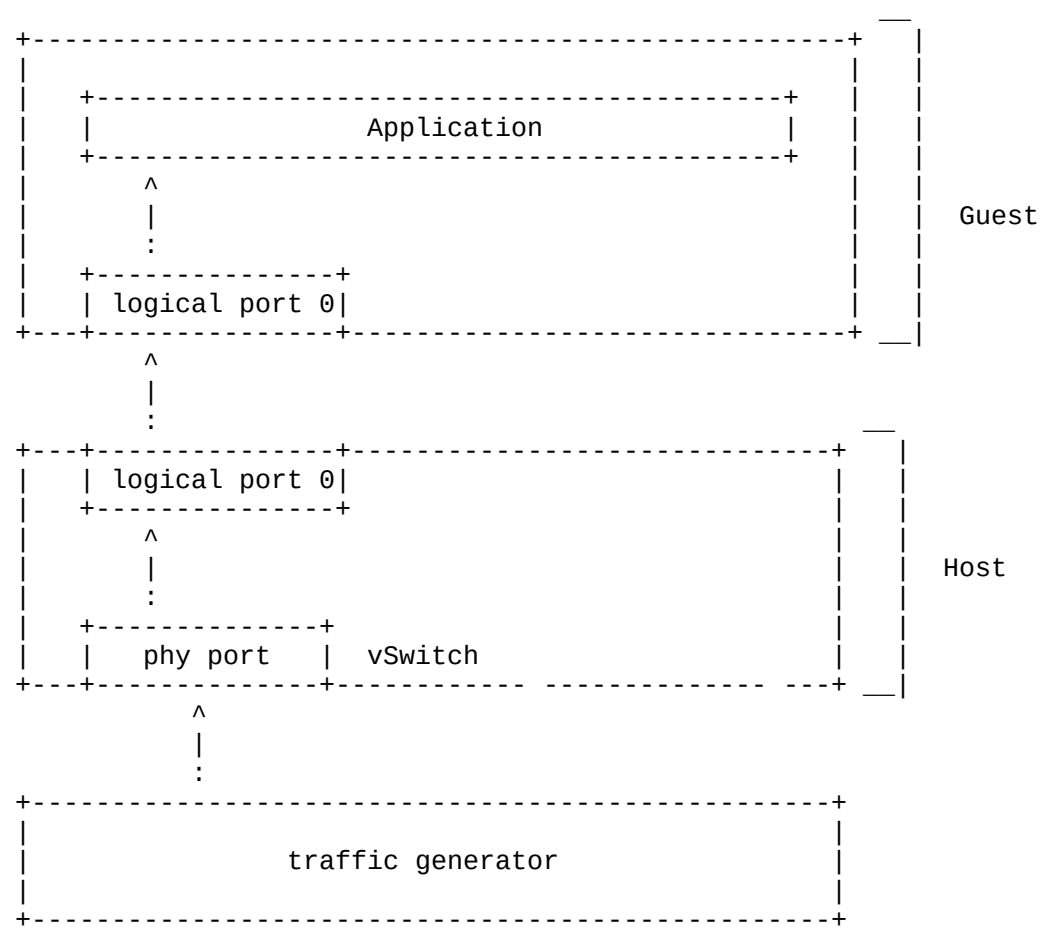
Physical port to virtual switch to VNF to virtual switch to physical port



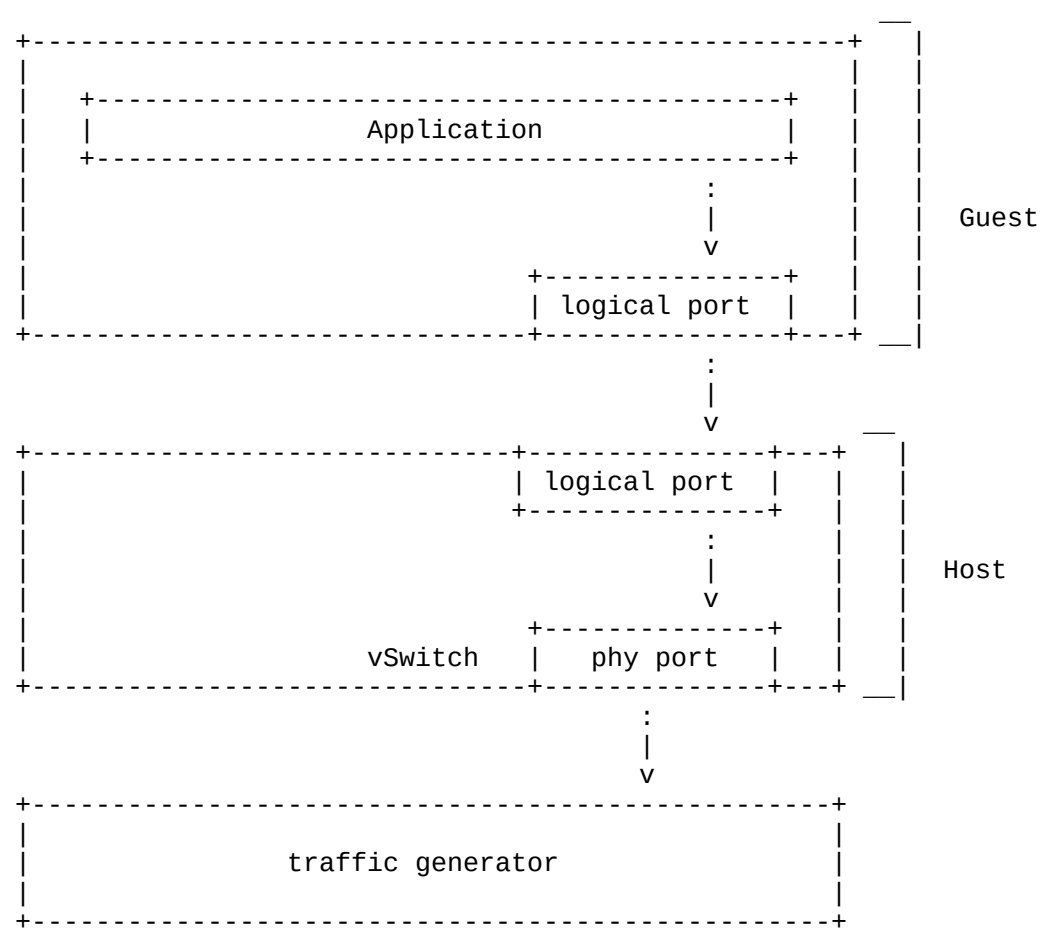
Physical port to virtual switch to VNF to virtual switch to VNF to virtual switch to physical port



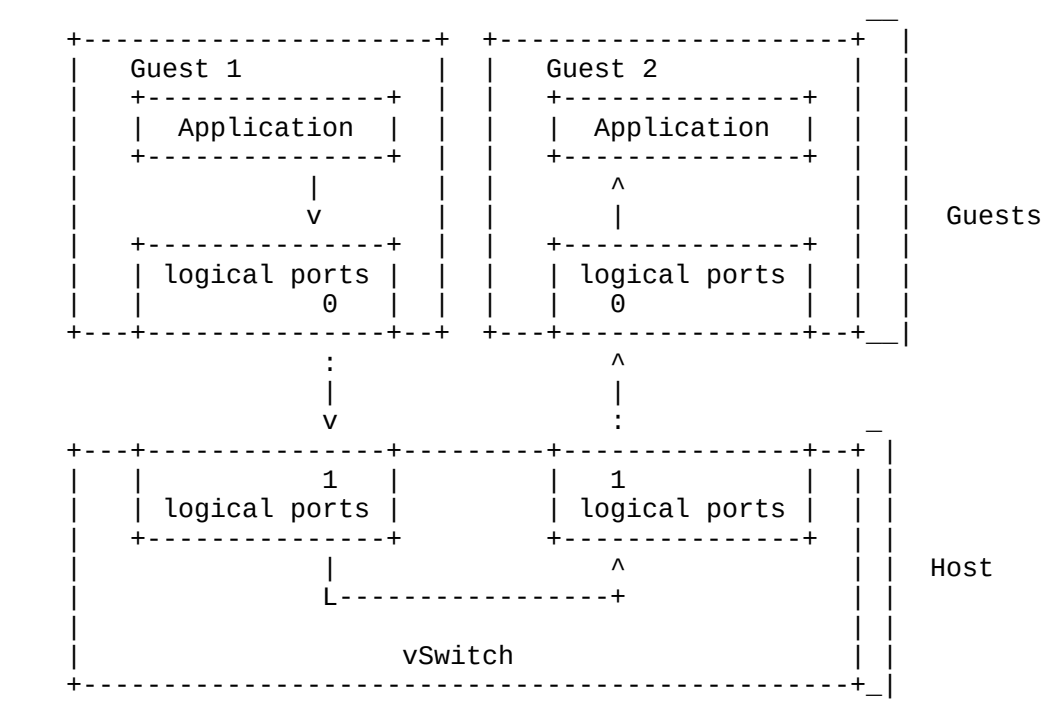
Physical port to virtual switch to VNF



VNF to virtual switch to physical port



VNF to virtual switch to VNF



A set of Deployment Scenario figures is available on the VSPERF Test Methodology Wiki page [TestTopo].

5. 3x3 Matrix Coverage

This section organizes the many existing test specifications into the "3x3" matrix (introduced in [I-D.ietf-bmwg-virtual-net]). Because the LTD specification ID names are quite long, this section is organized into lists for each occupied cell of the matrix (not all are occupied, also the matrix has grown to 3x4 to accommodate scale metrics when displaying the coverage of many metrics/benchmarks). The current version of the LTD specification is available [LTD].

The tests listed below assess the activation of paths in the data plane, rather than the control plane.

A complete list of tests with short summaries is available on the VSPERF "LTD Test Spec Overview" Wiki page [LTDoverV].

5.1. Speed of Activation

- Activation.RFC2889.AddressLearningRate
- PacketLatency.InitialPacketProcessingLatency

5.2. Accuracy of Activation section

- CPDP.Coupling.Flow.Addition

5.3. Reliability of Activation

- Throughput.RFC2544.SystemRecoveryTime
- Throughput.RFC2544.ResetTime

5.4. Scale of Activation

- Activation.RFC2889.AddressCachingCapacity

5.5. Speed of Operation

- Throughput.RFC2544.PacketLossRate
- CPU.RFC2544.0PacketLoss
- Throughput.RFC2544.PacketLossRateFrameModification
- Throughput.RFC2544.BackToBackFrames
- Throughput.RFC2889.MaxForwardingRate
- Throughput.RFC2889.ForwardPressure
- Throughput.RFC2889.BroadcastFrameForwarding

5.6. Accuracy of Operation

- Throughput.RFC2889.ErrorFramesFiltering
- Throughput.RFC2544.Profile

5.7. Reliability of Operation

- Throughput.RFC2889.Soak
- Throughput.RFC2889.SoakFrameModification

- o PacketDelayVariation.RFC3393.Soak

5.8. Scalability of Operation

- o Scalability.RFC2544.0PacketLoss
- o MemoryBandwidth.RFC2544.0PacketLoss.Scalability

5.9. Summary

	SPEED	ACCURACY	RELIABILITY	SCALE
Activation	X	X	X	X
Operation	X	X	X	X
De-activation				

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. Acknowledgements

The authors appreciate and acknowledge comments from Scott Bradner, Marius Georgescu, Ramki Krishnan, Doug Montgomery, Martin Klozik, Christian Trautman, and others for their reviews.

9. References

9.1. Normative References

[NFV.PER001] "Network Function Virtualization: Performance and Portability Best Practices", Group Specification ETSI GS NFV-PER 001 V1.1.1 (2014-06), June 2014.

[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>.

[RFC2285] Mandeville, R., "Benchmarking Terminology for LAN Switching Devices", RFC 2285, DOI 10.17487/RFC2285, February 1998, <http://www.rfc-editor.org/info/rfc2285>.

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <http://www.rfc-editor.org/info/rfc2330>.

[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>.

[RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, DOI 10.17487/RFC2679, September 1999, <http://www.rfc-editor.org/info/rfc2679>.

[RFC2680] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, DOI 10.17487/RFC2680, September 1999, <http://www.rfc-editor.org/info/rfc2680>.

[RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <http://www.rfc-editor.org/info/rfc2681>.

[RFC2889] Mandeville, R. and J. Perser, "Benchmarking Methodology for LAN Switching Devices", RFC 2889, DOI 10.17487/RFC2889, August 2000, <http://www.rfc-editor.org/info/rfc2889>.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <http://www.rfc-editor.org/info/rfc3393>.

[RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, DOI 10.17487/RFC3432, November 2002, <http://www.rfc-editor.org/info/rfc3432>.

[RFC3918] Stopp, D. and B. Hickman, "Methodology for IP Multicast Benchmarking", RFC 3918, DOI 10.17487/RFC3918, October 2004, <http://www.rfc-editor.org/info/rfc3918>.

[RFC4689] Poretsky, S., Perser, J., Erramilli, S., and S. Khurana, "Terminology for Benchmarking Network-layer Traffic Control Mechanisms", RFC 4689, DOI 10.17487/RFC4689, October 2006, <http://www.rfc-editor.org/info/rfc4689>.

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, DOI 10.17487/RFC4737, November 2006, <http://www.rfc-editor.org/info/rfc4737>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <http://www.rfc-editor.org/info/rfc5357>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <http://www.rfc-editor.org/info/rfc5905>.

[RFC6201] Asati, R., Pignataro, C., Calabria, F., and C. Olvera, "Device Reset Characterization", RFC 6201, DOI 10.17487/RFC6201, March 2011, <http://www.rfc-editor.org/info/rfc6201>.

9.2. Informative References

[BrahRel] "Brahmaputra, Second OPNFV Release https://www.opnfv.org/brahmaputra".

[I-D.huang-bmwg-virtual-network-performance] Huang, L., Rong, G., Mandeville, B., and B. Hickman, "Benchmarking Methodology for Virtualization Network Performance", draft-huang-bmwg-virtual-network-performance-01 (work in progress), April 2015.

[I-D.ietf-bmwg-virtual-net] Morton, A., "Considerations for Benchmarking Virtual Network Functions and Their Infrastructure", draft-ietf-bmwg-virtual-net-04 (work in progress), August 2016.

[IFA003] "https://docbox.etsi.org/ISG/NFV/Open/Drafts/IFA003_Acceleration_-_vSwitch_Spec/".

[LTD] "LTD Test Specification http://artifacts.opnfv.org/vswitchperf/brahmaputra/docs/requirements/index.html".

[LTDoverV] "LTD Test Spec Overview https://wiki.opnfv.org/wiki/vswitchperf_test_spec_review".

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

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <http://www.rfc-editor.org/info/rfc5481>.

[RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, DOI 10.17487/RFC6049, January 2011, <http://www.rfc-editor.org/info/rfc6049>.

[RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, DOI 10.17487/RFC6248, April 2011, <http://www.rfc-editor.org/info/rfc6248>.

[RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, DOI 10.17487/RFC6390, October 2011, <http://www.rfc-editor.org/info/rfc6390>.

[TestTopo] "Test Topologies https://wiki.opnfv.org/vsperf/test_methodology".

Authors' Addresses

Maryam Tahhan
Intel

Email: maryam.tahhan@intel.com

Billy O'Mahony
Intel

Email: billy.o.mahony@intel.com

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
URI: http://home.comcast.net/~acmacm/

Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 10, 2017

M. Tahhan
B. O'Mahony
Intel
A. Morton
AT&T Labs

June 8, 2017

Benchmarking Virtual Switches in OPNFV
draft-ietf-bmwg-vswitch-opnfv-04

Abstract

This memo describes the contributions of the Open Platform for NFV (OPNFV) project on virtual switch performance "VSPERF", particularly in the areas of test set-ups and configuration parameters for the system under test. This project has extended the current and completed work of the Benchmarking Methodology Working Group in IETF, and references existing literature. The Benchmarking Methodology Working Group has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions. Therefore, this memo describes the additional considerations when virtual switches are implemented in general-purpose hardware. The expanded tests and benchmarks are also influenced by the OPNFV mission to support virtualization of the "telco" infrastructure.

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 December 10, 2017.

Table of Contents

1. Introduction

Benchmarking Methodology Working Group (BMWG) has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions. The Black-box Benchmarks of Throughput, Latency, Forwarding Rates and others have served our industry for many years. Now, Network Function Virtualization (NFV) has the goal to transform how internetwork functions are implemented, and therefore has garnered much attention.

A virtual switch (vswitch) is an important aspect of the NFV infrastructure; it provides connectivity between and among physical network functions and virtual network functions. As a result, there are many vswitch benchmarking efforts, but few specifications to guide the many new test design choices. This is a complex problem and an industry-wide work-in-progress. In future, several of BMWG's fundamental specifications will likely be updated as more testing experience helps to form consensus around new methodologies, and BMWG should continue to collaborate with all organizations who share the same goal.

This memo describes the contributions of the Open Platform for NFV (OPNFV) project on virtual switch performance characterization, "VSPERF", through the Danube 3.0 (fourth) release [DanubeRel] to the chartered work of the BMWG (with stable references to their test descriptions). This project has extended the current and completed work of the BMWG in IETF, and references existing literature. For example, the most often referenced RFC is [RFC2544] (which depends on [RFC1242]), so the foundation of the benchmarking work in OPNFV is common and strong. The recommended extensions are specifically in the areas of test set-ups and configuration parameters for the system under test.

See [VSPERFhome] for more background, and the OPNFV website for general information [OPNFV].

The authors note that OPNFV distinguishes itself from other open source compute and networking projects through its emphasis on existing "telco" services as opposed to cloud-computing. There are many ways in which telco requirements have different emphasis on performance dimensions when compared to cloud computing: support for and transfer of isochronous media streams is one example.

1.1. Abbreviations

For the purposes of this document, the following abbreviations apply:

ACK Acknowledge
ACPI Advanced Configuration and Power Interface
BIOS Basic Input Output System
BMWG Benchmarking Methodology Working Group
CPDP Control Plane Data Plane
CPU Central Processing Unit
DIMM Dual In-line Memory Module
DPDK Data Plane Development Kit
DUT Device Under Test
GRUB Grand Unified Bootloader
ID Identification
IMIX Internet Mix
IP Internet Protocol
IPPM IP Performance Metrics
LAN Local Area Network
LTD Level Test Design
NFV Network Functions Virtualisation
NIC Network Interface Card
NUMA Non Uniform Memory Access
OPNFV Open Platform for NFV
OS Operating System
PCI Peripheral Component Interconnect
PDV Packet Delay Variation
SR/IOV Single Root/Input Output Virtualization
SUT System Under Test
SW Software
TCP Transmission control Protocol
TSO TCP Segment Offload
UDP User Datagram Protocol
VM Virtual Machine
VNF Virtualised Network Function
VSPERF OPNFV vSwitch Performance Project

2. Scope

The primary purpose and scope of the memo is to describe key aspects of vswitch benchmarking, particularly in the areas of test set-ups and configuration parameters for the system under test, and extend the body of extensive BMWG literature and experience. Initial feedback indicates that many of these extensions may be applicable beyond this memo's current scope (to hardware switches in the NFV Infrastructure and to virtual routers, for example). Additionally, this memo serves as a vehicle to include more detail and relevant commentary from BMWG and other Open Source communities, under BMWG's chartered work to characterize the NFV Infrastructure.

The benchmarking covered in this memo should be applicable to many types of vswitches, and remain vswitch-agnostic to great degree.

There has been no attempt to track and test all features of any specific vswitch implementation.

3. Benchmarking Considerations

This section highlights some specific considerations (from [I-D.ietf-bmwg-virtual-net])related to Benchmarks for virtual switches. The OPNFV project is sharing its present view on these areas, as they develop their specifications in the Level Test Design (LTD) document.

3.1. Comparison with Physical Network Functions

To compare the performance of virtual designs and implementations with their physical counterparts, identical benchmarks are needed. BMWG has developed specifications for many physical network functions. The BMWG has recommended to re-use existing benchmarks and methods in [I-D.ietf-bmwg-virtual-net], and the OPNFV LTD expands on them as described here. A key configuration aspect for vswitches is the number of parallel CPU cores required to achieve comparable performance with a given physical device, or whether some limit of scale will be reached before the vswitch can achieve the comparable performance level.

It’s unlikely that the virtual switch will be the only application running on the System Under Test (SUT), so CPU utilization, Cache utilization, and Memory footprint should also be recorded for the virtual implementations of internetworking functions. However, internally-measured metrics such as these are not benchmarks; they may be useful for the audience (operations) to know, and may also be useful if there is a problem encountered during testing.

Benchmark Comparability between virtual and physical/hardware implementations of equivalent functions will likely place more detailed and exact requirements on the *testing systems* (in terms of stream generation, algorithms to search for max values, and their configurations of course). This is another area for standards development to appreciate. However, the is a topic for a future draft.

3.2. Continued Emphasis on Black-Box Benchmarks

External observations remain essential as the basis for Benchmarks. Internal observations with fixed specification and interpretation will be provided in parallel to assist the development of operations procedures when the technology is deployed.

3.3. New Configuration Parameters

A key consideration when conducting any sort of benchmark is trying to ensure the consistency and repeatability of test results. When benchmarking the performance of a vswitch there are many factors that can affect the consistency of results, one key factor is matching the various hardware and software details of the SUT. This section lists some of the many new parameters which this project believes are critical to report in order to achieve repeatability.

It has been the goal of the project to produce repeatable results, and a large set of the parameters believed to be critical is provided so that the benchmarking community can better appreciate the increase in configuration complexity inherent in this work. The parameter set below is assumed sufficient for the infrastructure in use by the VSPERF project to obtain repeatable results from test-to-test.

Hardware details (platform, processor, memory, and network) including:

- BIOS version, release date and any configurations that were modified
- Power management at all levels (ACPI sleep states, processor package, OS...)
- CPU microcode level
- Number of enabled cores
- Number of cores used for the test
- Memory information (type and size)
- Memory DIMM configurations (quad rank performance may not be the same as dual rank) in size, freq and slot locations
- Number of physical NICs, as well as their details (manufacturer, versions, type and the PCI slot they are plugged into)
- NIC interrupt configuration (and any special features in use)
- PCI configuration parameters (payload size, early ACK option, etc.)

Software details including:

- OS parameters and behavior (text vs graphical no one typing at the console on one system)
- OS version (for host and VNF)
- Kernel version (for host and VNF)
- GRUB boot parameters (for host and VNF)
- Hypervisor details (Type and version)
- Selected vswitch, version number or commit id used
- vswitch launch command line if it has been parameterised
- Memory allocation to the vswitch
- which NUMA node it is using, and how many memory channels
- DPDK or any other SW dependency version number or commit id used
- Memory allocation to a VM - if it’s from Hugepages/elsewhere
- VM storage type: snapshot/independent persistent/independent non-persistent
- Number of VMs
- Number of Virtual NICs (vNICs), versions, type and driver
- Number of virtual CPUs and their core affinity on the host
- Number vNIC interrupt configuration
- Thread affinitization for the applications (including the vswitch itself) on the host
- Details of Resource isolation, such as CPUs designated for Host/ Kernel (isolcpu) and CPUs designated for specific processes (taskset). - Test duration. - Number of flows.

Test Traffic Information:

- Traffic type - UDP, TCP, others.
- Frame Sizes - fixed or IMIX [RFC6985](with [IEEE802.1ac], frames may be longer than 1500 bytes, and up to 2000 bytes)

- Deployment Scenario - defines the communications path in the SUT

3.4. Flow classification

Virtual switches group packets into flows by processing and matching particular packet or frame header information, or by matching packets based on the input ports. Thus a flow can be thought of a sequence of packets that have the same set of header field values, or have arrived on the same physical or logical port. Performance results can vary based on the parameters the vswitch uses to match for a flow. The recommended flow classification parameters for any vswitch performance tests are: the input port (physical or logical), the source MAC address, the destination MAC address, the source IP address, the destination IP address and the Ethernet protocol type field (although classification may take place on other fields, such as source and destination transport port numbers). It is essential to increase the flow timeout time on a vswitch before conducting any performance tests that do not intend to measure the flow setup time, see Section 3 of [RFC2889]. Normally the first packet of a particular stream will install the flow in the virtual switch which adds an additional latency, subsequent packets of the same flow are not subject to this latency if the flow is already installed on the vswitch.

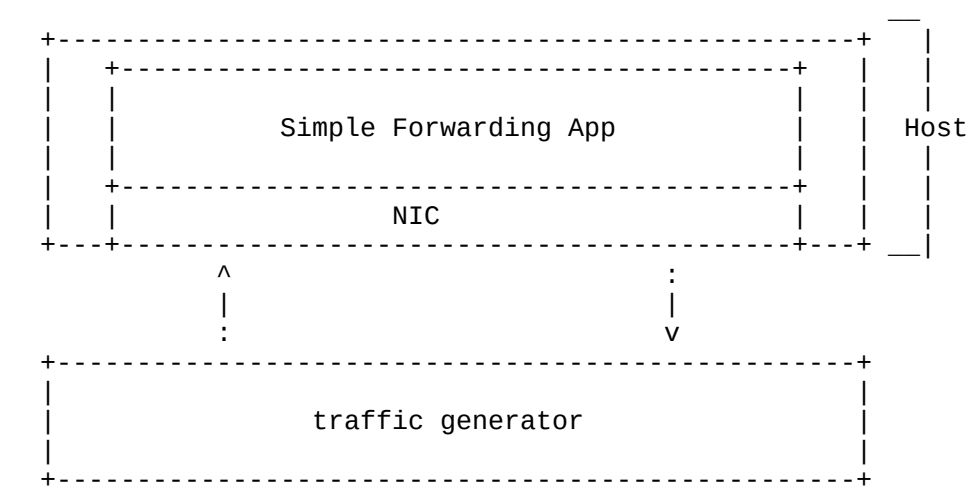
3.5. Benchmarks using Baselines with Resource Isolation

This outline describes measurement of baseline with isolated resources at a high level, which is the intended approach at this time.

1. Baselines:

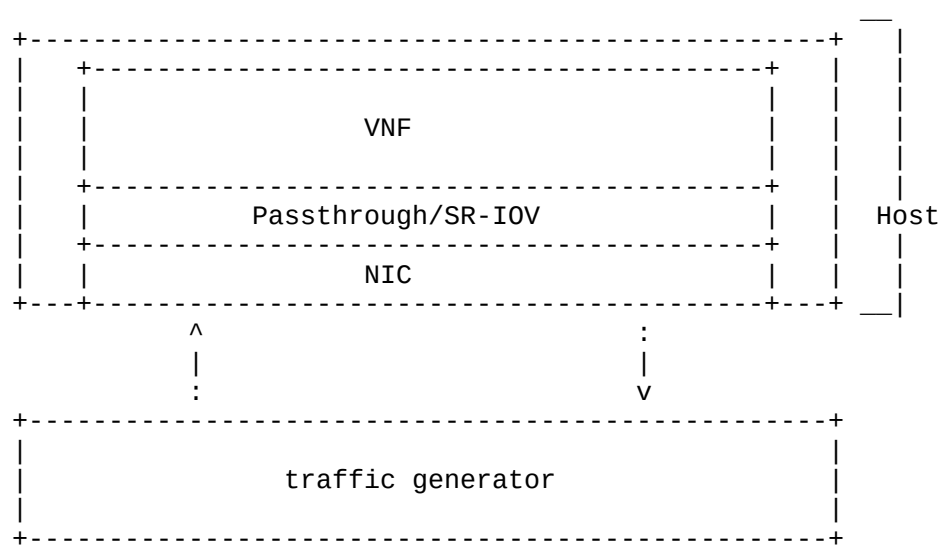
    * Optional: Benchmark platform forwarding capability without a vswitch or VNF for at least 72 hours (serves as a means of platform validation and a means to obtain the base performance for the platform in terms of its maximum forwarding rate and latency).

Figure 1 Benchmark platform forwarding capability



* Benchmark VNF forwarding capability with direct connectivity (vswitch bypass, e.g., SR/IOV) for at least 72 hours (serves as a means of VNF validation and a means to obtain the base performance for the VNF in terms of its maximum forwarding rate and latency). The metrics gathered from this test will serve as a key comparison point for vswitch bypass technologies performance and vswitch performance.

Figure 2 Benchmark VNF forwarding capability



* Benchmarking with isolated resources alone, with other resources (both HW&SW) disabled Example, vswitch and VM are SUT

* Benchmarking with isolated resources alone, leaving some resources unused

* Benchmark with isolated resources and all resources occupied

2. Next Steps

* Limited sharing

* Production scenarios

* Stressful scenarios

4. VSPERF Specification Summary

The overall specification in preparation is referred to as a Level Test Design (LTD) document, which will contain a suite of performance tests. The base performance tests in the LTD are based on the pre-existing specifications developed by BMWG to test the performance of physical switches. These specifications include:

- [RFC2544] Benchmarking Methodology for Network Interconnect Devices
- [RFC2889] Benchmarking Methodology for LAN Switching
- [RFC6201] Device Reset Characterization
- [RFC5481] Packet Delay Variation Applicability Statement

Some of the above/newer RFCs are being applied in benchmarking for the first time, and represent a development challenge for test equipment developers. Fortunately, many members of the testing system community have engaged on the VSPERF project, including an open source test system.

In addition to this, the LTD also re-uses the terminology defined by:

- [RFC2285] Benchmarking Terminology for LAN Switching Devices

It is recommended that these references are included in future benchmarking specifications:

- [RFC3918] Methodology for IP Multicast Benchmarking
- [RFC4737] Packet Reordering Metrics

As one might expect, the most fundamental internetworking characteristics of Throughput and Latency remain important when the switch is virtualized, and these benchmarks figure prominently in the specification.

When considering characteristics important to "telco" network functions, additional performance metrics are needed. In this case, the project specifications have referenced metrics from the IETF IP Performance Metrics (IPPM) literature. This means that the [RFC2544] test of Latency is replaced by measurement of a metric derived from IPPM’s [RFC2679], where a set of statistical summaries will be provided (mean, max, min, and percentiles). Further metrics planned to be benchmarked include packet delay variation as defined by [RFC5481] , reordering, burst behaviour, DUT availability, DUT capacity and packet loss in long term testing at Throughput level, where some low-level of background loss may be present and characterized.

Tests have been designed to collect the metrics below:

- Throughput Tests to measure the maximum forwarding rate (in frames per second or fps) and bit rate (in Mbps) for a constant load (as defined by [RFC1242]) without traffic loss.

- Packet and Frame Delay Distribution Tests to measure average, min and max packet and frame delay for constant loads.

- Packet Delay Tests to understand latency distribution for different packet sizes and over an extended test run to uncover outliers.

- Scalability Tests to understand how the virtual switch performs with increasing number of flows, number of active ports, configuration complexity of the forwarding logic, etc.

- Stream Performance Tests (TCP, UDP) to measure bulk data transfer performance, i.e. how fast systems can send and receive data through the switch.

- Control Path and Datapath Coupling Tests, to understand how closely the datapath and the control path are coupled, as well as the effect of this coupling on the performance of the DUT (example: delay of the initial packet of a flow).

- CPU and Memory Consumption Tests to understand the virtual switch’s footprint on the system, conducted as auxiliary measurements with benchmarks above. They include: CPU utilization, Cache utilization and Memory footprint.

- The so-called "Soak" tests, where the selected test is conducted over a long period of time (with an ideal duration of 24 hours, but only long enough to determine that stability issues exist when found; there is no requirement to continue a test when a DUT exhibits instability over time). The key performance characteristics and benchmarks for a DUT are determined (using short duration tests) prior to conducting soak tests. The purpose of soak tests is to capture transient changes in performance which may occur due to infrequent processes, memory leaks, or the low probability coincidence of two or more processes. The stability of the DUT is the paramount consideration, so performance must be evaluated periodically during continuous testing, and this results in use of [RFC2889] Frame Rate metrics instead of [RFC2544] Throughput (which requires stopping traffic to allow time for all traffic to exit internal queues), for example.

Additional test specification development should include:

- Request/Response Performance Tests (TCP, UDP) which measure the transaction rate through the switch.

- Noisy Neighbour Tests, to understand the effects of resource sharing on the performance of a virtual switch.

- Tests derived from examination of ETSI NFV Draft GS IFA003 requirements [IFA003] on characterization of acceleration technologies applied to vswitches.

The flexibility of deployment of a virtual switch within a network means that it is necessary to characterize the performance of a vswitch in various deployment scenarios. The deployment scenarios under consideration include:

Figure 3 Physical port to virtual switch to physical port

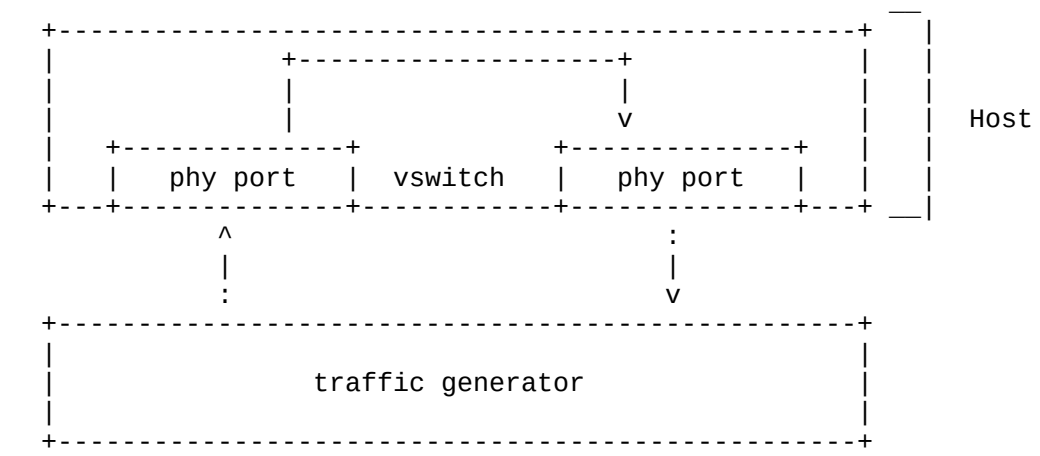


Figure 4 Physical port to virtual switch to VNF to virtual switch to physical port

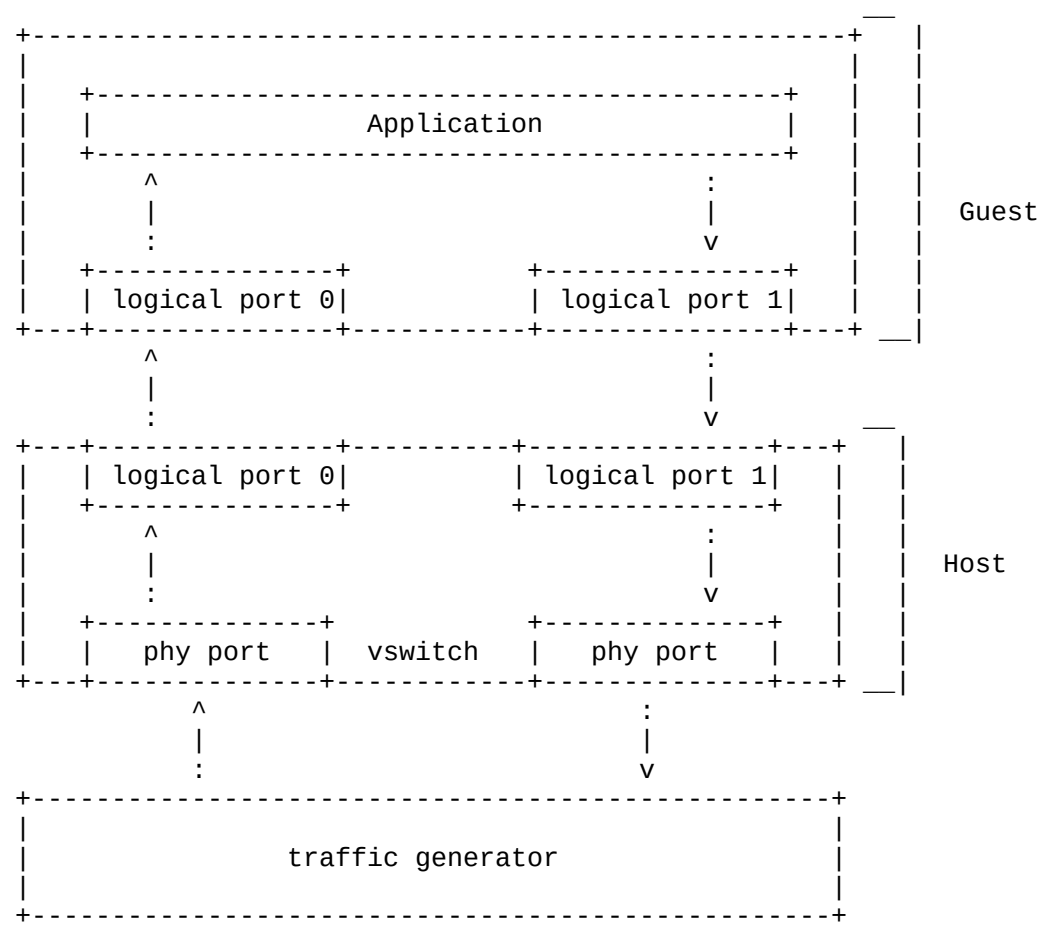


Figure 5 Physical port to virtual switch to VNF to virtual switch to VNF to virtual switch to physical port

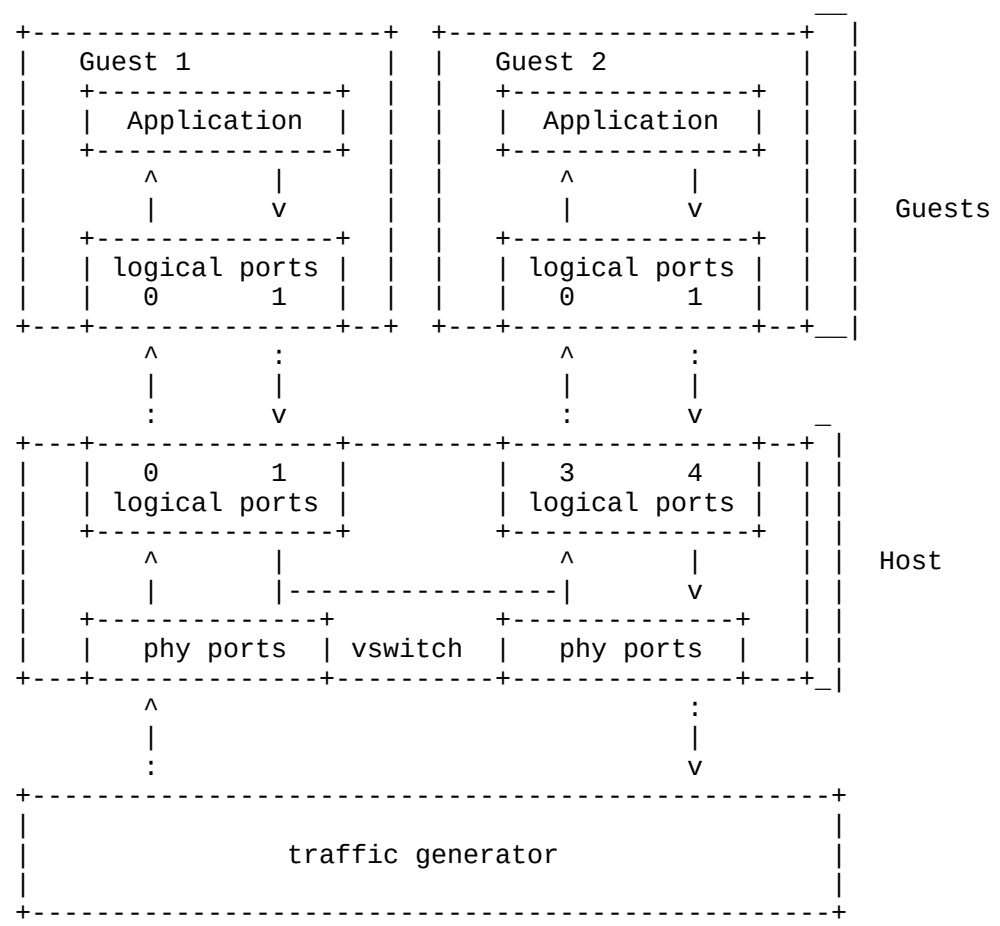


Figure 6 Physical port to virtual switch to VNF

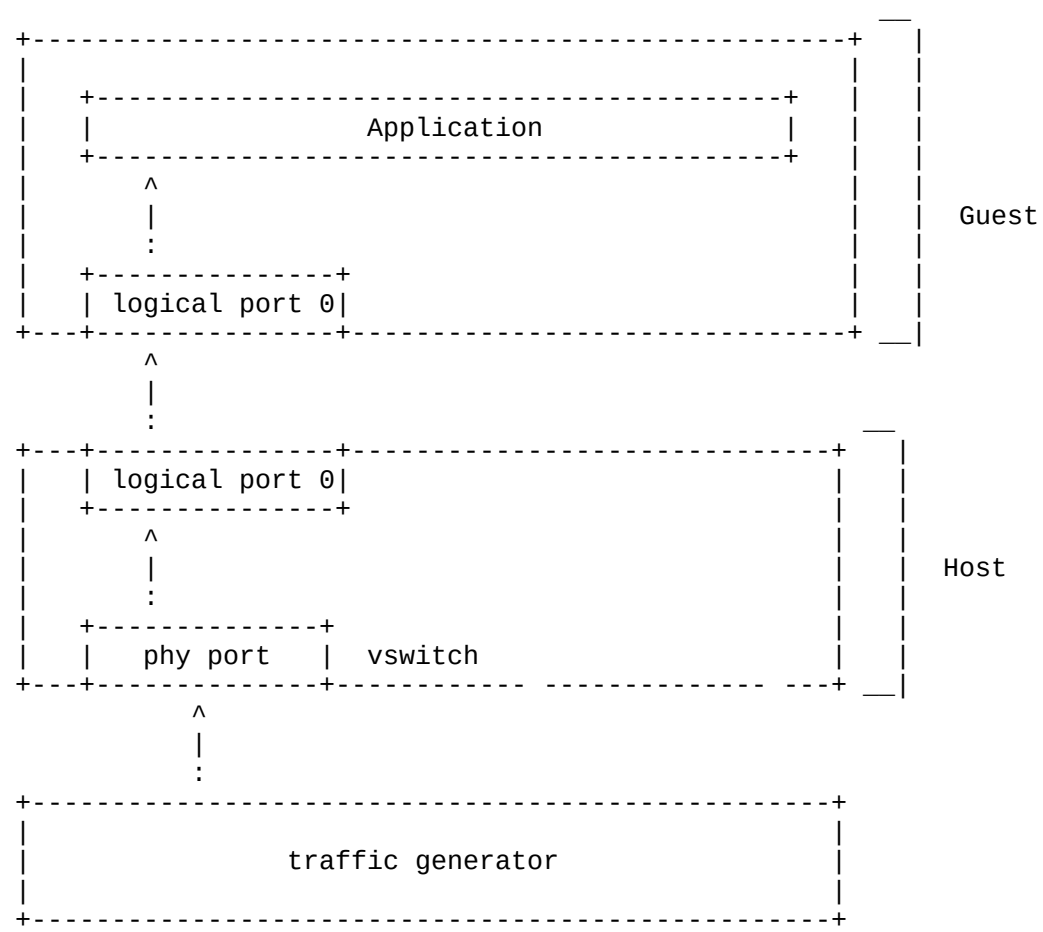


Figure 7 VNF to virtual switch to physical port

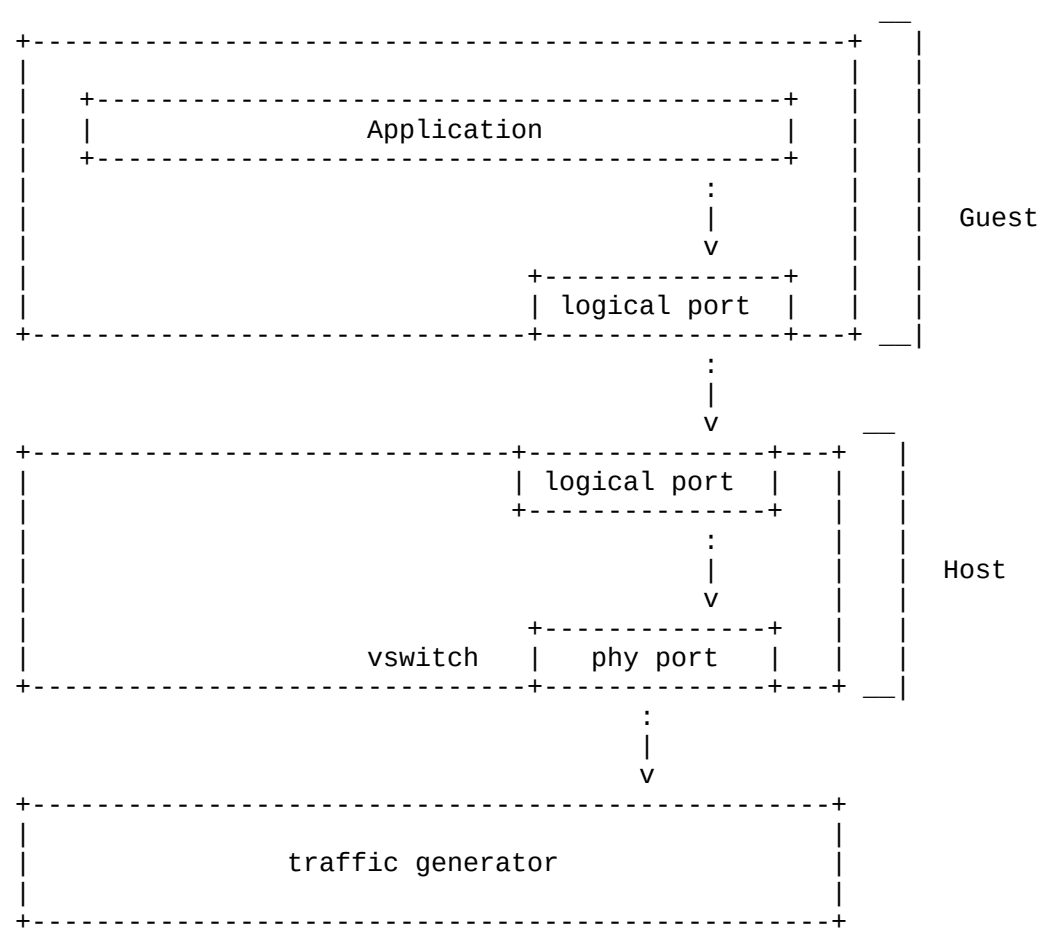
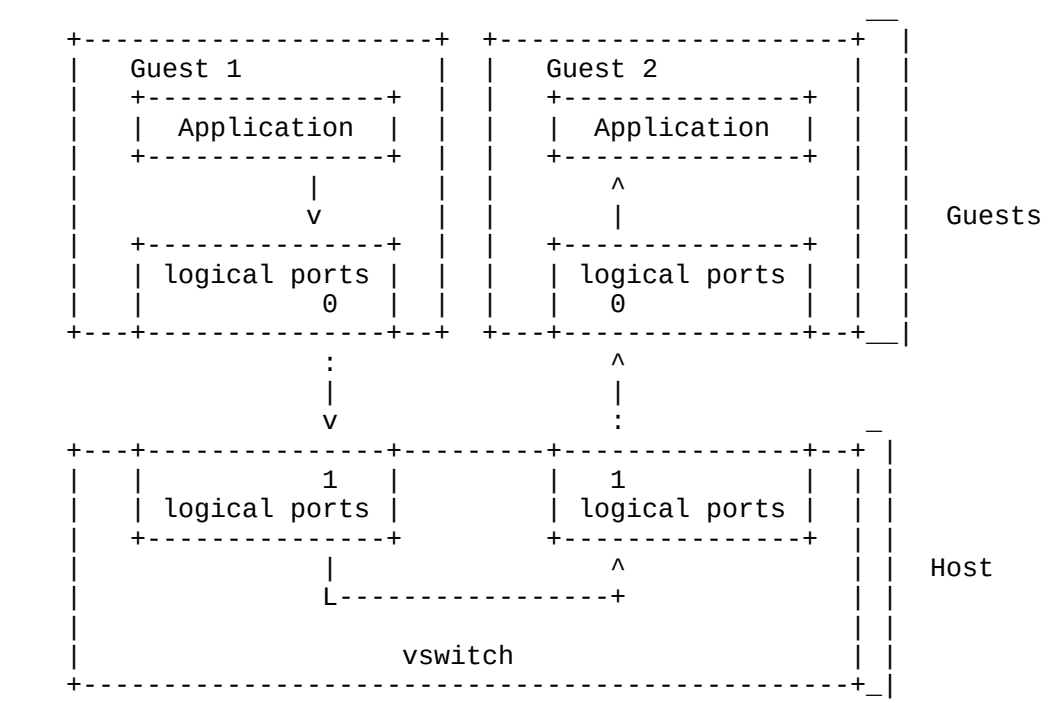


Figure 8 VNF to virtual switch to VNF



A set of Deployment Scenario figures is available on the VSPERF Test Methodology Wiki page [TestTopo].

5. 3x3 Matrix Coverage

This section organizes the many existing test specifications into the "3x3" matrix (introduced in [I-D.ietf-bmwg-virtual-net]). Because the LTD specification ID names are quite long, this section is organized into lists for each occupied cell of the matrix (not all are occupied, also the matrix has grown to 3x4 to accommodate scale metrics when displaying the coverage of many metrics/benchmarks). The current version of the LTD specification is available [LTD].

The tests listed below assess the activation of paths in the data plane, rather than the control plane.

A complete list of tests with short summaries is available on the VSPERF "LTD Test Spec Overview" Wiki page [LTDoverV].

5.1. Speed of Activation

- Activation.RFC2889.AddressLearningRate
- PacketLatency.InitialPacketProcessingLatency

5.2. Accuracy of Activation section

- CPDP.Coupling.Flow.Addition

5.3. Reliability of Activation

- Throughput.RFC2544.SystemRecoveryTime
- Throughput.RFC2544.ResetTime

5.4. Scale of Activation

- Activation.RFC2889.AddressCachingCapacity

5.5. Speed of Operation

- Throughput.RFC2544.PacketLossRate
- Stress.RFC2544.0PacketLoss
- Throughput.RFC2544.PacketLossRateFrameModification
- Throughput.RFC2544.BackToBackFrames
- Throughput.RFC2889.MaxForwardingRate
- Throughput.RFC2889.ForwardPressure
- Throughput.RFC2889.BroadcastFrameForwarding
- Throughput.RFC2544.WorstN-BestN
- Throughput.Overlay.Network.<tech>.RFC2544.PacketLossRatio

5.6. Accuracy of Operation

- Throughput.RFC2889.ErrorFramesFiltering
- Throughput.RFC2544.Profile

5.7. Reliability of Operation

- o Throughput.RFC2889.Soak
- o Throughput.RFC2889.SoakFrameModification
- o PacketDelayVariation.RFC3393.Soak

5.8. Scalability of Operation

- o Scalability.RFC2544.0PacketLoss
- o MemoryBandwidth.RFC2544.0PacketLoss.Scalability
- o Scalability.VNF.RFC2544.PacketLossProfile
- o Scalability.VNF.RFC2544.PacketLossRatio

5.9. Summary

	SPEED	ACCURACY	RELIABILITY	SCALE
Activation	X	X	X	X
Operation	X	X	X	X
De-activation				

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. Acknowledgements

The authors appreciate and acknowledge comments from Scott Bradner, Marius Georgescu, Ramki Krishnan, Doug Montgomery, Martin Klozik, Christian Trautman, and others for their reviews.

We also acknowledge the early work in [I-D.huang-bmwg-virtual-network-performance], and useful discussion with the authors.

9. References

9.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2285] Mandeville, R., "Benchmarking Terminology for LAN Switching Devices", RFC 2285, DOI 10.17487/RFC2285, February 1998, <http://www.rfc-editor.org/info/rfc2285>.

[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>.

[RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, DOI 10.17487/RFC2679, September 1999, <http://www.rfc-editor.org/info/rfc2679>.

[RFC2889] Mandeville, R. and J. Perser, "Benchmarking Methodology for LAN Switching Devices", RFC 2889, DOI 10.17487/RFC2889, August 2000, <http://www.rfc-editor.org/info/rfc2889>.

[RFC3918] Stopp, D. and B. Hickman, "Methodology for IP Multicast Benchmarking", RFC 3918, DOI 10.17487/RFC3918, October 2004, <http://www.rfc-editor.org/info/rfc3918>.

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, DOI 10.17487/RFC4737, November 2006, <http://www.rfc-editor.org/info/rfc4737>.

[RFC6201] Asati, R., Pignataro, C., Calabria, F., and C. Olvera, "Device Reset Characterization", RFC 6201, DOI 10.17487/RFC6201, March 2011, <http://www.rfc-editor.org/info/rfc6201>.

[RFC6985] Morton, A., "IMIX Genome: Specification of Variable Packet Sizes for Additional Testing", RFC 6985, DOI 10.17487/RFC6985, July 2013, <http://www.rfc-editor.org/info/rfc6985>.

9.2. Informative References

[DanubeRel]
"Danube, Fourth OPNFV Release https://wiki.opnfv.org/display/SWREL/Danube".

[I-D.huang-bmwg-virtual-network-performance]
Huang, L., Rong, G., Mandeville, B., and B. Hickman, "Benchmarking Methodology for Virtualization Network Performance", draft-huang-bmwg-virtual-network-performance-02 (work in progress), March 2017.

[I-D.ietf-bmwg-virtual-net]
Morton, A., "Considerations for Benchmarking Virtual Network Functions and Their Infrastructure", draft-ietf-bmwg-virtual-net-05 (work in progress), March 2017.

[IEEE802.1ac]
https://standards.ieee.org/findstds/standard/802.1AC-2016.html, "802.1AC-2016 - IEEE Standard for Local and metropolitan area networks -- Media Access Control (MAC) Service Definition", 2016.

[IFA003] "https://docbox.etsi.org/ISG/NFV/Open/Drafts/IFA003_Acceleration_-_vSwitch_Spec/".

[LTD] Note: if the Danube Release LTD is available in artifacts at publication, we will use that URL instead., "LTD Test Specificationhttp://artifacts.opnfv.org/vswitchperf/colorado/docs/requirements/vswitchperf_ltd.html".

[LTDoverV] "LTD Test Spec Overview https://wiki.opnfv.org/display/vsperf/LTD+Test+Spec+Overview".

[OPNFV] "OPNFV Home https://www.opnfv.org/".

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

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <http://www.rfc-editor.org/info/rfc5481>.

[TestTopo] "Test Topologies https://wiki.opnfv.org/display/vsperf/Test+Methodology".

[VSPERFhome] "VSPERF Home https://wiki.opnfv.org/display/vsperf/VSperf+Home".

Authors' Addresses

Maryam Tahhan
Intel

Email: maryam.tahhan@intel.com

Billy O'Mahony
Intel

Email: billy.o.mahony@intel.com

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
URI: http://home.comcast.net/~acmacm/

Network Working Group
Internet Draft
Intended Status: Informational
Expires: March 27,2017

Sudhin Jacob
Praveen Ananthasankaran
Juniper Networks
September 28,2016

Benchmarking of Y1731 Performance Monitoring
draft-jacpra-bmwg-pmtest-02

Abstract

The draft defines the methodologies for benchmarking of the Y1731 performance monitoring on DUT in various methods like Calculation of near-end and far-end data. Measurement is done in scenarios by using pre-defined COS and without COS in the network.The test includes Impairment test,High Availability test and soak tests.

Table of Contents

1. Introduction

Performance monitoring is explained in ITU Y1731.This document defines the methodologies for benchmarking performance of Y1731 over a point to point service. Performance Monitoring has been implemented with many varying designs in order to achieve their intended network functionality. The scope of this document is to define methodologies for benchmarking Y1731 performance measurement. The following protocols under Y.1731 will be benchmarke d.

1. Two-way delay measurement
2. One-way delay measurement
3. Loss measurement
4. Synthetic loss measurement

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

PM Performance monitoring

In-profile CIR termed as green packets.

Out-profile EIR Yellow/Amber packet.

LMM Loss Measurement Message

LMR Loss Measurement Reply

DMM Delay Measurement Message

DMR Delay MEasurement Reply

P Router Provider Router.

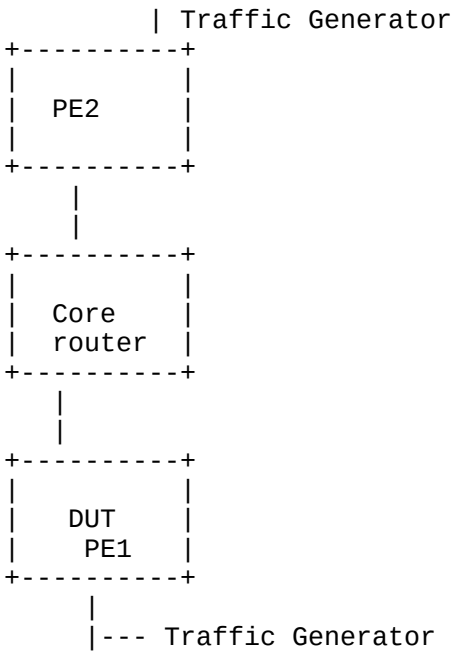
PE Router Provider Edge Router

CE Router customer Edge Router

DUT Device under Test.

CCM Continuity check messages

2.1 Test Topology



3. Network

The benchmarking topology consists of 3 routers and 2 traffic generators. DUT is PE1 connected to CE. The core router is the P router mentioned in the topology. There is layer two(point-to-point) services running from PE1 to PE2. On the top of that performance monitoring such as loss,delay and synthetic measurements are running.PE1 is acting as DUT.The traffic will be layer 2 with vlan tag.The frame size will be 64,128,512,1024 and 1400.The tests are carried out using these various frame size.The traffic will be uni directional or bi directional.

4. Test Procedure

The tests are defined to bench mark the Y1731 performance monitoring in High Availability,Impairment,SOAK,Scale,with traffic of various line rate and frame sizes.

4.1 Performance Monitoring with traffic

Traffic is send with different .1p priorities,line rate and frame size of 64, 128,512,1024,1400.The PM values are measured with each frame size with various line rates.

4.2 High Availability

The traffic is flowing bi direction.Then traffic is flowing at "P" packets per sec. The traffic generator is measuring the Tx and Rx packets, while the routing engine failover there should not any packet loss the router tester must show bot
h
"P" packet per seconds.The PM historical data should not reset.

4.3 Scale

This is to measure the performance of DUT in scaling to "X" CFM sessions with Performance monitoring running over it.There should not be any crashes,memory leaks.

4.4 SOAK

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

4.5 Measurement Statistics

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

5 Test Cases

5.1 Y.1731 Two-way Delay Measurement Test procedure

Basic Testing Objective

Check the round trip delay of the network in different conditions of traffic load in the network.

Test Procedure

Configure a layer 2 point-to-point service between PE1 and PE2.
Configure Y.1731 Two way delay measurement over the service.Observe the delay measurement in the following conditions of traffic in the network

a. Send 80% of Line-rate traffic with different priorities and frame size.
b. Send 40% of Line-rate traffic with different priorities and frame size.
c. Without any line traffic

The result of all the 3 conditions above are noted and correlated.

Test Measurement

The following factors needs are to be measured to benchmark the result
1. The average two-way delay
2. The average two-way delay variation

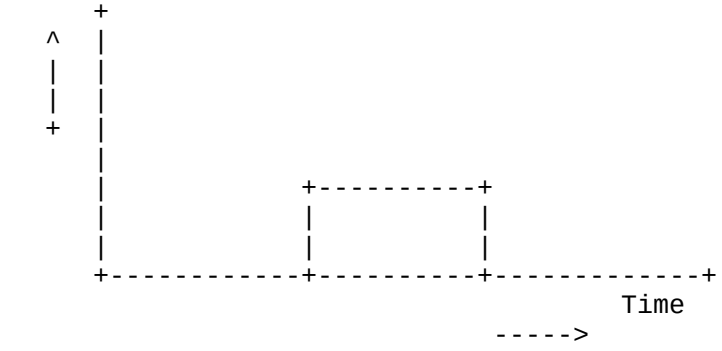
In the above 3 conditions the results obtained must be similar

1. Ideal case

In this case the hardware aspects of processing capacity and the link level anomalies are not considered. The benchmark is just on the protocol functioning.In such environment where for an ideal case the system should expect delay variation to be zero.

2. Practical case

This case is used to benchmark results when delay measurement is done on physical hardware (like a router).The factors of packet process jitter and link level delays needs to be considered here.The delay variation in such cases will defer based on the above parameters on different hardware systems. Result will very base on the exact hardware.



Traffic (0 to 100 percent line rate)

Impairment

This is to benchmark two-way delay measurement even when both data and PDUs are dropped in the network using the impairment tool.

Measurement

The results must show similar results before and after this test.

High Availability

During routing engine failover the historical data must not reset.

Scale

This is to measure the performance of DUT in scaling to "X" CFM sessions with Performance monitoring running over it.There should not be any crashes,memory leaks.

Soak

The bi directional traffic is send over service over 24 to 48 hours and measure after the stipulated time there must not be any change in behavior in the network for performance monitoring

Measurement

There should not be any core or crashes,memory leaks.

5.2 One-Way delay measurement Test Procedure

Basic Testing Objective

The test defined to measure the one-way delay measurement. One-way delay measurement as defined in Y.1731 is the delay of the packet to originate from a specific end-point till it reached the other end of the network. The measurement of this mandates the clock to be accurately synchronized as the delay is computed based on the time of two different end-points.

Test Procedure

Configure a layer2 point-to-point service between PE1 and PE2. Configure Y.1731 one-way delay measurement over the service. Observe the delay measurement delay measurement in the following conditions of traffic in the network

a. Send 80% of Line-rate traffic with different priorities with different frame size.
b. Send 40% of Line-rate traffic with different priorities with different frame size.
c. Without any line traffic

The result of all the 3 conditions above are noted and correlated.

Test Measurement

The following factors needs to be measured to benchmark the result

The average one-way delay
The average one-way delay variation

In the above 3 cases results obtained must be similar.

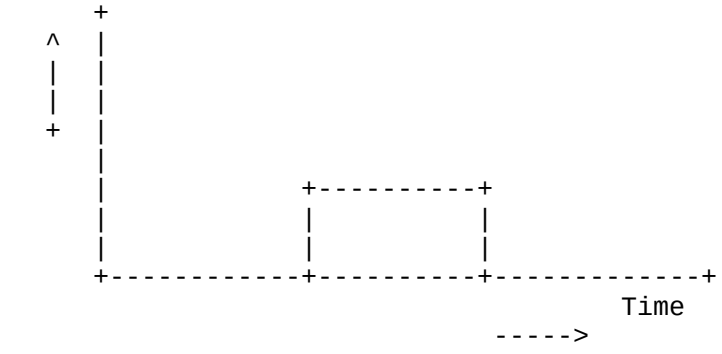
1. Ideal case

In this case the hardware aspects of processing capacity and the link level anomalies are not considered. The benchmark is just on the protocol functioning.In such environment where for an ideal case the system should expect delay variation to be zero.

2. Practical case

This case is used to benchmark results when delay measurement is done on physical hardware (like a router).The factors of packet process jitter and link level delays needs to be considered here.The delay variation in such cases will defer based on the above parameters on different hardware systems. Result will very base on the exact hardware.

Delay Variation



Traffic (0 to 100 percent line rate)

Impairment

This is to benchmark one-way delay measurement even when both data and PDUs are dropped in the network using the impairment tool.

Measurement

The results must show similar results before and after this test.

High Availability

During routing engine failover the historical data must not reset.

Scale

This is to measure the performance of DUT in scaling to "X" CFM sessions with Performance monitoring running over it.There should not be any crashes,memory leaks.

Soak

The bi directional traffic is send over service over 24 to 48 hours and measure after the stipulated time there must not be any change in behavior in the network for performance monitoring

Measurement

There should not be any core or crashes,memory leaks.

5.3 Loss measurement without COS Test Procedure

Basic Testing Objective

The test defined methodology for benchmarking data loss in the network on real customer traffic. The Y.1731 indicates to consider only in-profile (green) packet for loss measurement. For this, the testing needs to be done in multiple environment where

a.All data packets from traffic generator are sent with single 802.1p priority and the network do not have a COS profile defined.

b.All data packets from traffic generator are sent with 0 to 7 values for 802.1p priority and the network do not have a COS profile defined.

The objective is to benchmark the protocol behavior under different networking conditions and correlate the data.The objective is not to test the actual functioning of Y.1731 Loss measurement.The loss measurement must count only in profile packet, since there is no COS defined.All the packets must be recorded as green.

Test Procedure

Configure a layer2 point-to-point service between PE1 and PE2.
Configure Y.1731 loss measurement over the service.
Observe the loss measurement in the following conditions of traffic in

the network

a.Send 80% of Line-rate traffic with different priorities with different frame size.
b.Send 40% of Line-rate traffic with different priorities with different frame size.
c.Without any line traffic

The result of all the 3 conditions above are noted and correlated.

Test Measurement

The factors which need to be considered is the acceptable absolute loss for the given network.

Impairment

This is to benchmark loss measurement even when both data and PDUs are dropped in the network using the impairment tool.

Measurement

When the data is dropped it must show the loss correctly and PM PDUs are dropped the counting should not be affected,ther should not be any abnormal output.

High Availability

During routing engine failover the historical data must not reset.

Scale

This is to measure the performance of DUT in scaling to "X" CFM sessions with Performance monitoring running over it.There should not be any crashes,memory leaks.

Soak

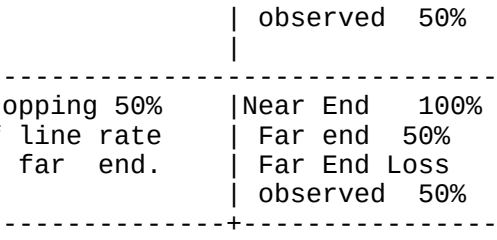
The bi directional traffic is send over service over 24 to 48 hours and measure after the stipulated time there must not be any change in behavior in the network for performance monitoring

Measurement

There should not be any core or crashes,memory leaks.

Result

Traffic sent over the service for bi direction	Loss measurement (without cos)
7 Streams at 100% line rate with priority from 0 to 7	Near End = 100% Far End = 100%
Dropping 50% of line rate at near end.	Near End 50% Far end 100% Near End loss



5.4. Loss measurement with COS Test Procedure

Basic Testing Objective

The test defined methodology for benchmarking data loss in the network on real customer traffic. The Y.1731 indicates to consider only in-profile(green) packet for loss measurement. For this, the testing needs to be done in multiple environment where

a. All data packets from traffic generator are sent with single 802.1p priority and the network have pre-defined COS profile defined.

b. All data packets from traffic generator are sent with 0 to 7 values for 802.1p priority and the network have pre-defined COS profile defined.

The COS profile defined needs to have 2 factors

a.COS needs to treat different 802.1p as separate class of packets.

b.Each Class of packets needs to be an defined CIR for the specific network.

The objective is to benchmark the protocol behavior under different networking conditions and correlate the data. The objective is not to test the actual functioning of Y.1731 Loss measurement.
The loss measurement must show in profile packet for each COS levels.
Each COS level must count only its own defined in profile packets.
The Packets, which are termed, as out profile by COS marking must not be counted.

Test Procedure

Configure a layer2 point-to-point service between PE1 and PE2.
Configure Y.1731 loss measurement over the service.
Observe the loss measurement in the following conditions of traffic in the network.

d.Send 80% of Line-rate traffic with different priorities with different frame size.
e.Send 40% of Line-rate traffic with different priorities with different frame size.
f. Without any line traffic

The result of all the 3 conditions above are noted and correlated.

Test Measurement

The factors which need to be considered is the acceptable absolute loss for the given network.

Impairment

This is to benchmark loss measurement even when both data and PDUs are dropped in the network using the impairment tool.

Measurement

When the data is dropped it must show the loss correctly and PM PDUs are dropped the counting should not be affected,there should not be any abonormal output.

High Availability

During routing engine failover the historical data must not reset.

Scale

This is to measure the performance of DUT in scaling to "X" CFM sessions with Performance monitoring running over it.There should not be any crashes,memory leaks.

Soak

The bi directional traffic is send over service over 24 to 48 hours and measure after the stipulated time there must not be any change in behavior in the network for performance monitoring

Measurement

There should not be any core or crashes,memory leaks.

Result

Traffic sent over the service for bi direction	Loss measurement (With cos)
7 Streams at 100% line rate with priority from 0 to 7	Near End = 100% Far End = 100%
Dropping 50% of line rate at near end for priority marked 0	Near End 50% Far end 100% Near End loss observed 50% (priority 0)
Dropping 50% of line rate at far end for priority 0	Near End 100% Far end 50% Far End Loss observed 50% (priority 0)

5.5. Synthetic Loss Measurement Test Procedure

5.5.1 Basic Testing Objective

The test defined methodology for benchmarking synthetic loss in the network. The testing needs to be done in multiple environment where

a. All data packets from traffic generator are sent with single 802.1p priority and the network do not have a COS profile defined.
The synthetic loss measurement also uses the same 802.1p priority as that of traffic.

b. All data packets from traffic generator are sent with single 802.1p priority and the network have pre-defined COS profile defined.The synthetic loss measurement also uses the same 802.1p priority as that of traffic.

c. All data packets from traffic generator are sent with 0 to 7 values for 802.1p priority and the network do not have a COS profile defined. The synthetic loss measurement also uses the same 802.1p priority as that of traffic. Hence 8 sessions are tested in parallel.

d. All data packets from traffic generator are sent with 0 to 7 values for 802.1p priority and the network have pre-defined COS profile defined. The synthetic loss measurement also uses the same 802.1p priority as that of traffic. Hence 8 sessions are tested in parallel.

The COS profile defined needs to have 2 factors

1.COS needs to treat different 802.1p as separate class of packets.

2.Each Class of packets needs to have defined CIR for the specific network.

The objective is to benchmark the protocol behavior under different networking conditions and correlate the data. The objective is not to test the actual functioning of Y.1731 Loss measurement.

Test Procedure

Configure a layer2 point-to-point service between PE1 and PE2.
Configure Y.1731 loss measurement over the service. Observe the synthetic loss measurement in the following conditions of traffic in the network

a. Send 80% of Line-rate traffic with different priorities
b. Send 40% of Line-rate traffic with different priorities
c. Without any line traffic
The result of all the 3 conditions above are noted and correlated.

Test Measurement

The factors which need to be considered is the acceptable absolute loss for the given network.

Impairment

This is to benchmark synthetic loss measurement even when both data and PDUs are dropped in the network using the impairment tool.

Measurement

When the data is dropped it must not affect the SLM counters but if synthetic frames are dropped the loss must be shown accordingly.

High Availability

During routing engine failover the historical data must not reset.

Scale

This is to measure the performance of DUT in scaling to "X" CFM sessions with Performance monitoring running over it.There should not be any crashes,memory leaks.

Soak

The bi directional traffic is send over service over 24 to 48 hours and measure after the stipulated time there must not be any change in behavior in the network for performance monitoring

Measurement

There should not be any core or crashes,memory leaks.

6. Acknowledgements

We would like to thank Al Morton of (ATT) for their support and encouragement. We would like to thank Fioccola Giuseppe of Telecom Italia reviewing our draft and commenting it.

7. Security Considerations

NA

8.IANA Considerations

NA

Appendix A. Appendix

Authors' Addresses

Sudhin Jacob
Juniper Networks
Bangalore

Email: sjacob@juniper.net
sudhinjacob@rediffmail.com

Praveen Ananthasankaran
Juniper Networks
Bangalore

Email: panantha@juniper.net

Internet Engineering Task Force S. Jacob, Ed.
Internet-Draft K. Tiruveedhula
Intended status: Standards Track Juniper Networks
Expires: December 2, 2016 May 31, 2016

Benchmarking Methodology for EVPN
draft-kishjac-bmwg-evpntest-01

Abstract

This document defines the methodologies for benchmarking performance of EVPN. EVPN is defined in RFC 7432. It is being deployed in provider network. This document provides the benchmarking methodologies for EVPN convergance, data plane and 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 December 2, 2016.

Table of Contents

1. Introduction

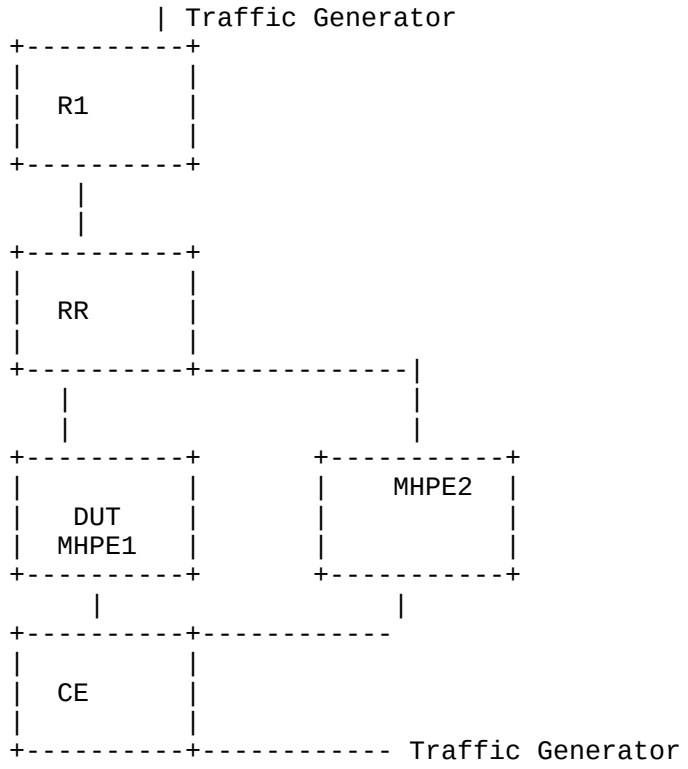
EVPN which is defined in RFC7432 which describes procedures for BGP MPLS-based Ethernet VPNs(EVPN). This document defines the methodologies for benchmarking performance of EVPN. EVPN has been implemented with many varying designs in order to achieve their intended network functionality. The scope of this document is to provide methodologies for benchmarking evpn data, control plane mac learning, mac flush, mac ageing, convergence, high availablity, reliablity 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].

2. Test Topology

Topology Diagram



Topology Diagram

Figure 1

3. Network

The network consists of 5 routers and 2 traffic generator ports. DUT is acting as one of MH PE 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. CE is a dual home connected to DUT and MH PE1. The testing will be done on DUT to bench mark the service. DUT and MHPE2 is running EVPN with SA/AA with CE. The DUT and and other PE's will be running 100 EVI's(EVPN instances) on 100 sub interfaces.

4. Test Procedure

The test defined to bench mark the performance of EVPN mac learning in control pland and data plane, Mac flush, High Availablity, convergence time in link failures, Reliablity and 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". The data plane learning will be 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’".

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" once the traffic is stopped. The time taken flush the remote mac which is learned by control plane ("X") macs when the traffic is stopped at remote side.Let the measured time is "T2".

4.3. High Availablity

The traffic is flowing bi direction. The bgp is converged, let the "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 failover there should not any packet loss the router tester must show both "P" packet per seconds.

4.4. Reliablity

This is to measure during any events like adding a new PE or rebooting on of the MHPE in the same ethernet segment,there should not any issues in DF election. There should not be any loop. The DF elction must take place in t time where 4>t>3. The DF election must work on V mod N. There should not be any loop in traffic.

4.5. Convergence Time

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

4.6. Scale

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

4.7. SOAK

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.

4.8. Measurement Statistics

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

5. Test Cases

5.1. MAC Learning in Control plane and Data Plane

The following tests are conducted to measure mac learning local and 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 100,000 unicast frames from CE to MHPE1(DUT) working in SA mode where DUT is the DF so that it can forward the traffic with different source and destination address,Measure the time taken to learn these mac in forwarding table and control plane.

b. Measure the time taken to send these 100,000 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 routers. Measurement The DUT mac table must learn the 100,000 mac address and measure the time taken to send

100,000 routes to its peers by DUT. Measurement The DUT mac table must learn the 100,000 measure the time taken to learn the 100,000 mac address in forwarding table and time taken to advertise to remote Peer.

5.1.2. To check the time taken to learn 100,000 routes from remote peer by DUT.

Objective:

Send 100,000 frames with different SA and DA to DUT from R1 using traffic generator. Measure the time taken to learn these 100,000 routes 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) to send the traffic to the routers.

Measurement:

The DUT mac table must learn the 100,000 measure the time taken to learn the 100,000 mac address from remote peers.

5.1.3. To check the time taken to flush the local entry due to CE link failure

Objective:

Send 100,000 frames with different SA and DA to DUT from CE using traffic generator. Then wait to learn all 100,000 mac address. Then fail the DUT CE link and measure the time taken to flush these 100,000 routes 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.

Measurement:

The DUT mac table must learn the 100,000 measure the time taken for flushing these 100,000 mac address.

5.1.4. To check the time taken by DUT to flush 100,000 routes learned from remote PE after R1 traffic generator link failure

Objective:

Send 100,000 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 routers.

Measurement:

Measure the time taken to flush 100,000 remote routes from mac table of DUT.

5.1.5. To measure the mac ageing time.

Objective:

Send 100,000 frames with different SA and DA to DUT from CE using traffic generator. Then wait to learn all 100,000 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.

Measurement:

The DUT mac table must learn the 100,000 measure the time taken for flushing these 100,000 mac address due to ageing.

5.1.6. To check the time taken by DUT to age 100,000 routes learned from remote PE after stopping the traffic at remote PE.

Objective:

Send 100,000 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 the routers.

Measurement:

Measure the time taken to flush 100,000 remote macs from mac table of DUT due to ageing.

5.2. Convergence

The following tests are executed to measure the convergence time in case of an event or by lerning the mac without any external trigger.

5.2.1. To check the time taken by DUT to learn 100,000 routes from local and 100,000 from remote and measure the time of flood from DUT.

Objective:

Send 100,000 frames with different SA and DA to DUT from R1 using traffic generator. Send 100,000 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 200,000 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.

Measurement:

Measure the time taken to learn 200,000 routes in control and mac address table in DUT and measure the flood time of DUT be measuring the time taken to stop the flood to unicast flows.

5.2.2. To measure the time taken to elect a new DF by adding a MHPE.

Objective:

Send 100,000 frames from CE to DUT from traffic generator with different SA and DA. Wait to learn all 100,000 mac address. Then add R1 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 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.

5.3. Reliability

These tests are conducted to check after an event there wont be any change in functionality.

5.3.1. To check the whether there is traffic loss due to routing engine failover for redundancy test.

Objective:

Send 100,000 frames from CE to DUT from traffic generator with different SA and DA. Send 100,000 frames from traffic generator to R1 with different SA and DA so that 200,000 mac address will be

learned in DUT. There is a bi directional traffic flow with 100,000 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 not be any traffic loss,No change in the DF role. No withdraw of any routes.

5.3.2. To check the whether DF election is taking place properly after the one of the MH PE reboot.

Objective:

Send 100,000 frames from CE to DUT from traffic generator with different SA and DA. Send 100,000 frames from traffic generator to R1 with different SA and DA so that 200,000 mac address will be learned in DUT. There is a bi directional traffic flow with 100,000 pps in each direction. Then reboot DUT since there are 2 MH PE’s the other PE will become the DF then after DUT comes back, then DF election has to re initiate.

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 DF election has to take place again once the DUT comes back online.There should not be any DF stuck casefor 100 EVI’s.

5.4. Scale

5.4.1. To Scale the DUT to 8k EVI and clear bgp in DUT with out traffic.

Objective:

The main purpose of the test the DUT performance to scale 8k 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 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 1,3 and 4 in DUT.

5.4.2. To Scale the DUT to 8k EVI and clear bgp in DUT with traffic. Measure the convergence time

Objective:

The main purpose of the test the DUT performance to scale 8k EVI's with traffic. Then clear bgp neighbor. There should not be any loss of routes or any crashes. Send 100,000 frames from CE to DUT from traffic generator with different SA and DA for 5 EVI's. Send 100,000 frames from traffic generator to R1 with different SA and DA for 5 EVI's.so that 1,000,000 mac address will be learned in DUT. There is a bi directional traffic flow with 500,000 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 1,000,000 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 1,000,000 mac address. Measure the time taken to learn 1,000,000 mac routes.

5.5. Soak Test

5.5.1. To Scale the DUT to 8k EVI in DUT with traffic and run the set up for 24hrs

Objective:

The main purpose of the test the DUT performance to scale 8k EVI's with traffic. Then clear bgp neighbor. There should not be any loss of routes or any crashes. Send 100,000 frames from CE to DUT from traffic generator with different SA and DA for 5 EVI's. Send 100,000 frames from traffic generator to R1 with different SA and DA for 5 EVI's.so that 1,000,000 mac address will be learned in DUT. There is a bi directional traffic flow with 500,000 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.

6. Acknowledgements

We would like to thank Bhuvaneswaran Vengainathan of Veryx Technologies and Al Morton of (ATT) for their support and encouragements.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

There is no additional consideration from RFC 6192.

9. References

9.1. 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, <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>.

9.2. 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>.

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

Internet Engineering Task Force S. Jacob, Ed.
Internet-Draft K. Tiruveedhula
Intended status: Standards Track Juniper Networks
Expires: December 18, 2016 June 16, 2016

Benchmarking Methodology for PBB-EVPN
draft-kishjac-bmwg-pbbevpn-00

Abstract

This document defines the methodologies for benchmarking performance of PBB-EVPN.PBB-EVPN is defined in RFC 7623. It is being deployed in provider network.This document provides the benchmarking methodologies for PBB-EVPN convergance,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 December 18, 2016.

Table of Contents

1. Introduction

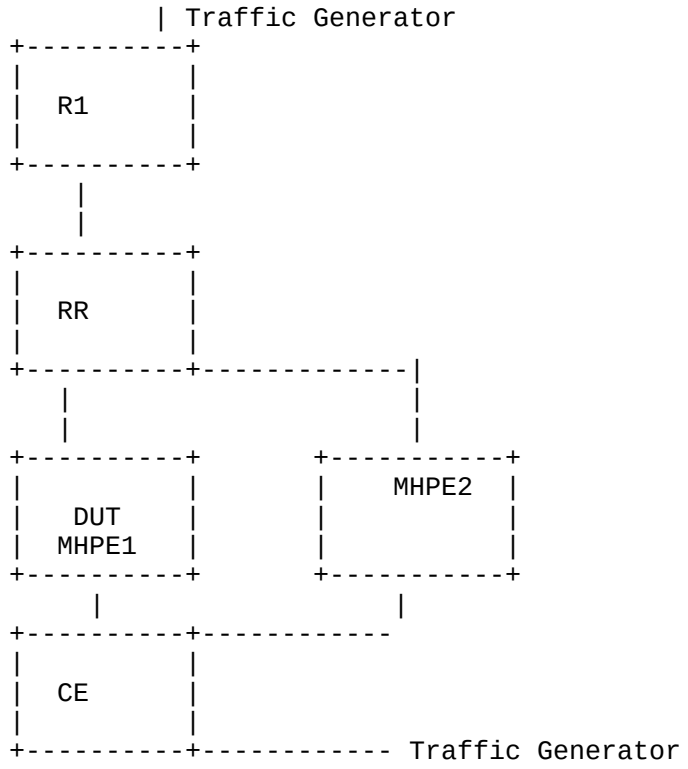
PBB-EVPN which is defined in RFC 7623 which describes procedures for BGP MPLS-based PBB-EVPN.This document defines the methodologies for benchmarking performance of PBB-EVPN.PBB-EVPN has been implemented with many varying designs in order to achieve their intended network functionality.The scope of this document is to provide methodologies for benchmarking pbb-evpn data, control plane mac learning, mac flush, mac ageing, convergence, high availablity, reliablity,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].

2. Test Topology

Topology Diagram



Topology Diagram

Figure 1

3. Network

The network consists of 5 routers and 2 traffic generator ports. DUT is acting as one of MH PE 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. CE is a dual home connected to DUT and MH PE1. The testing will be done on DUT to bench mark the service.DUT and MHPE2 is running PBB-EVPN with SA/AA with CE.The DUT and and other PE’s will be running 100 PBB-EVN instances on 100 sub interfaces.

4. Test Procedure

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

4.1. Customer MAC Learning in Data Plane and B-MAC in control plane

The Customer MAC will be learned in data plane, test is to measure the time taken to learn the "X" number of mac's in time "T". The data plane learning will from the locally connected interface. The control plane learning of B-MAC is through BGP advertisements from the remote PE.Let the local learning time be "T" and control plane learning of the B-MAC time be "T'".

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" once the traffic is stopped. The time taken flush the remote mac which learned by data plane that "X" macs when the traffic is stopped at remote side.Let the measured time is "T2".

4.3. High Availablity

The traffic is flowing bi direction. The bgp is converged, let the "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 failover there should not any packet loss the router tester must show both "P" packet per seconds.

4.4. Reliablity

This is to measure during any events like adding a new PE or rebooting on of the MHPE in the same ethernet segment,there should not any issues in DF election. There should not be any loop. The DF elction must take place in t time where 4>t>3. The DF election must work on V mod N. There should not be any loop in traffic.

4.5. Convergence Time

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

4.6. Scale

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

4.7. SOAK

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.

4.8. Measurement Statistics

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

5. Test Cases

5.1. MAC Learning in Control plane and Data Plane

The following tests are conducted to measure mac learning local and 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 100,000 unicast frames from CE to MHPE1(DUT) working in SA mode where DUT is the DF so that it can forward the traffic with different source and destination address,Measure the time taken to learn these mac in forwarding 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 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 100,000 mac address. Measurement The DUT mac table must

learn the 100,000 measure the time taken to learn the 100,000 mac address from locally.

5.1.2. To check the time taken to learn 100,000 routes from remote peer by DUT.

Objective:

Send 100,000 frames with different SA and DA to DUT from R1 using traffic generator. Measure the time taken to learn these 100,000 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 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 100,000 measure the time taken to learn the 100,000 mac address from remote peers.

5.1.3. To check the time taken to flush the local entry due to CE link failure

Objective:

Send 100,000 frames with different SA and DA to DUT from CE using traffic generator. Then wait to learn all 100,000 mac address. Then fail the DUT CE link and measure the time taken to flush these 100,000 routes from the 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:

The DUT mac table must learn the 100,000 measure the time taken for flushing these 100,000 mac address.

5.1.4. To check the time taken by DUT to flush 100,000 routes learned from remote PE after R1 traffic generator link failure

Objective:

Send 100,000 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 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 100,000 remote routes from mac table of DUT.

5.1.5. To measure the mac ageing time.

Objective:

Send 100,000 frames with different SA and DA to DUT from CE using traffic generator. Then wait to learn all 100,000 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 100,000 measure the time taken for flushing these 100,000 mac address due to ageing.

5.1.6. To check the time taken by DUT to age from remote PE after stopping the traffic at remote PE.

Objective:

Send 100,000 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 100,000 remote macs from mac table of DUT due to ageing.

5.2. Convergence

The following tests are executed to measure the convergence time in case of an event or by lerning the mac without any external trigger.

5.2.1. To check the time taken by DUT to learn 100,000 macs from local and 100,000 from remote and measure the time of flood from DUT.

Objective:

Send 100,000 frames with different SA and DA to DUT from R1 using traffic generator. Send 100,000 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 200,000 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 200,000 mac address table in DUT. and measure the flood time of DUT.

5.2.2. To measure the time taken to elect a new DF by adding a a MHPE.

Objective:

Send 100,000 frames from CE to DUT from traffic generator with different SA and DA. Wait to learn all 100,000 mac address. Then add R1 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.

5.3. Reliability

These tests are conducted to check after an event there wont be any change in functionality.

5.3.1. To check the whether there is traffic loss due to routing engine failover for redundancy test.

Objective:

Send 100,000 frames from CE to DUT from traffic generator with different SA and DA. Send 100,000 frames from traffic generator to R1 with different SA and DA so that 200,000 mac address will be learned in DUT. There is a bi directional traffic flow with 100,000 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 not be any traffic loss,No change in the DF role. No withdraw of any routes.

5.3.2. To check the whether DF election is taking place properly after the one of the MH PE reboot.

Objective:

Send 100,000 frames from CE to DUT from traffic generator with different SA and DA. Send 100,000 frames from traffic generator to R1 with different SA and DA so that 200,000 mac address will be learned in DUT. There is a bi directional traffic flow with 100,000 pps in each direction. Then reboot DUT since there are 2 MH PE’s the other PE will become the DF then after DUT comes back, then DF election has to re initiate.

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 DF election has to take place again once the DUT comes back online.There should not be any DF stuck casefor 100 PBB-EVPN instances.

5.4. Scale

5.4.1. To Scale the DUT to 4k PBB-EVPN instances and clear bgp in DUT without traffic.

Objective:

The main purpose of the test the DUT performance to scale 4k 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.

5.4.2. To Scale the DUT to 4k 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 4k pbb-evpn instances with traffic.Then clear bgp neighbor.There should not be any loss of routes or any crashes. Send 100,000 frames from CE to DUT from traffic generator with different SA and DA for 5 pbb-evpn instances.Send 100,000 frames from traffic generator to R1 with different SA and DA for 5 pbb-evpn instances.so that 1,000,000 mac address will be learned in DUT. There is a bi directional traffic flow with 500,000 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 1,000,000 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 1,000,000 mac address.Measure the time taken to learn 1,000,000 mac in DUT, measure the flood traffic time "T" of DUT

5.5. Soak Test

5.5.1. To Scale the DUT to 4k PBB-EVPN instances in DUT with traffic and run the set up for 24hrs

Objective:

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.

Procedure:

Measurement:

Take the hourly reading of CPU,process memory.There should not be any leak,crashes,CPU spikes.

6. Acknowledgements

We would like to thank Al Morton of (ATT) for their support and encouragement. We would like to thank Fioccola Giuseppe of Telecom Italia reviewing our draft and commenting it.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

There is no additional consideration from RFC 6192.

9. References

9.1. 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, <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>.

9.2. 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>.

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

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

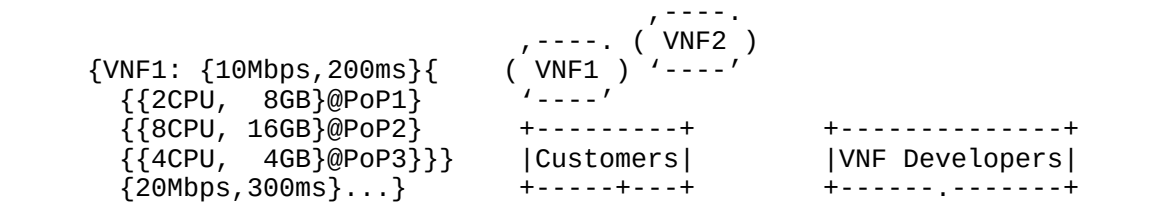
We assume a VNF benchmarking set-up as shown in Figure 1. Customers can request Network Services (NS) from an NFVO with associated service level specifications (e.g., throughput and delay). The NFVO, in turn, must select hosts and software resource allocations for the VNFs and build the necessary network 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.



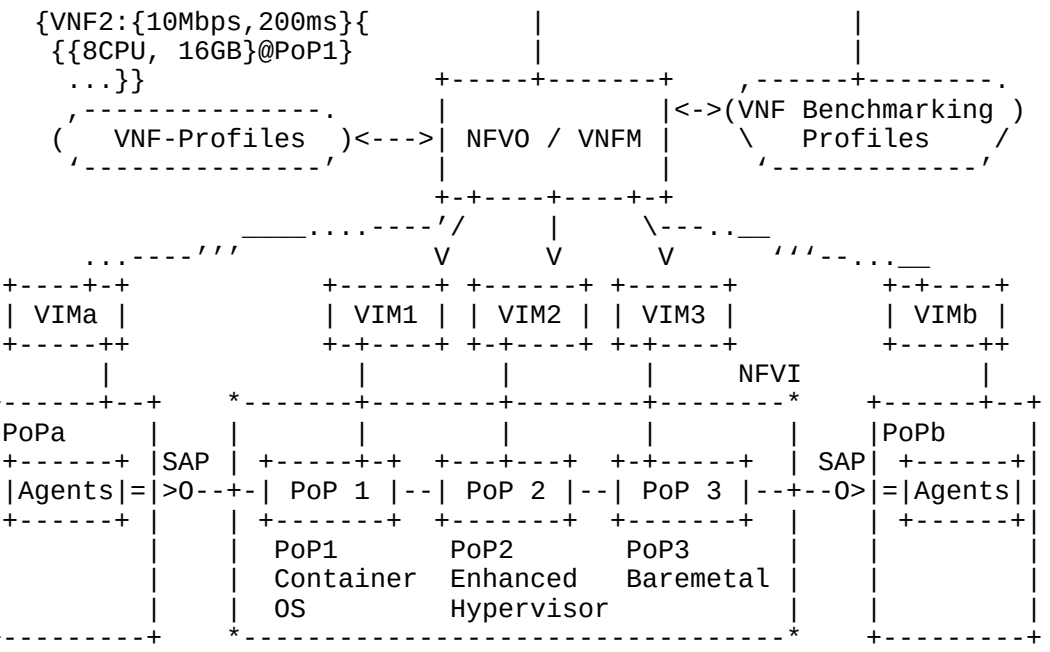


Figure 1: VNF Testing Scenario

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/