Network Working Group
Internet-Draft
Updates: 2544 (if approved)
Intended status: Informational
Expires: May 21, 2020

A. Morton
AT&T Labs

November 18, 2019

Updates for the Back-to-back Frame Benchmark in RFC 2544
draft-ietf-bmwg-b2b-frame-01

Abstract

Fundamental Benchmarking Methodologies for Network Interconnect Devices of interest to the IETF are defined in RFC 2544. This memo updates the procedures of the test to measure the Back-to-back frames Benchmark of RFC 2544, based on further experience.

This memo updates Section 26.4 of RFC 2544.

Requirements Language

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

Table of Contents

1. Introduction

The IETF's fundamental Benchmarking Methodologies are defined in [RFC2544], supported by the terms and definitions in [RFC1242], and [RFC2544] actually obsoletes an earlier specification, [RFC1944]. Over time, the benchmarking community has updated [RFC2544] several times, including the Device Reset Benchmark [RFC6201], and the important Applicability Statement [RFC6815] concerning use outside the Isolated Test Environment (ITE) required for accurate benchmarking. Other specifications implicitly update [RFC2544], such as the IPv6 Benchmarking Methodologies in [RFC5180].

Recent testing experience with the Back-to-back Frame test and Benchmark in Section 26.4 of [RFC2544] indicates that an update is warranted [OPNFV-2017] [VSPERF-b2b]. In particular, analysis of the results indicates that buffers size matters when compensating for disruptions in the software packet processor, and this finding increases the importance of the Back-to-back frame characterization described here. This memo describes additional rationale and provides the updated method.

[RFC2544] provides its own Requirements Language consistent with [RFC2119], since [RFC1944] predates [RFC2119]. Thus, the requirements presented in this memo are expressed in [RFC2119] terms, and intended for those performing/reporting laboratory tests to improve clarity and repeatability, and for those designing devices that facilitate these tests.

2. Scope and Goals

The scope of this memo is to define an updated method to unambiguously perform tests, measure the benchmark(s), and report the results for Back-to-back Frames (presently described Section 26.4 of [RFC2544]).

The goal is to provide more efficient test procedures where possible, and to expand reporting with additional interpretation of the results. The tests described in this memo address the cases where the maximum frame rate of a single ingress port cannot be transferred to an egress port loss-free (for some frame sizes of interest).

[RFC2544] Benchmarks rely on test conditions with constant frame sizes, with the goal of understanding what network device capability has been tested. Tests with the smallest size stress the header processing capacity, and tests with the largest size stress the overall bit processing capacity. Tests with sizes in-between may determine the transition between these two capacities. However, conditions simultaneously sending multiple frame sizes, such as those described in [RFC6985], MUST NOT be used in Back-to-back Frame testing.

Section 3 of [RFC8239] describes buffer size testing for physical networking devices in a Data Center. The [RFC8239] methods measure buffer latency directly with traffic on multiple ingress ports that overload an egress port on the Device Under Test (DUT), and are not subject to the revised calculations presented in this memo. Likewise, the methods of [RFC8239] SHOULD be used for test cases where the egress port buffer is the known point of overload.

3. Motivation

Section 3.1 of [RFC1242] describes the rationale for the Back-to-back Frames Benchmark. To summarize, there are several reasons that devices on a network produce bursts of frames at the minimum allowed spacing, and it is therefore worthwhile to understand the Device Under Test (DUT) limit on the length of such bursts in practice. Also, [RFC1242] states:

> "Tests of this parameter are intended to determine the extent of data buffering in the device."

After this test was defined, there have been occasional discussions of the stability and repeatability of the results, both over time and across labs. Fortunately, the Open Platform for Network Function Virtualization (OPNFV) VSPERF project's Continuous Integration (CI) testing routinely repeats Back-to-back Frame tests to verify that test functionality has been maintained through development of the test control programs. These tests were used as a basis to evaluate stability and repeatability, even across lab set-ups when the test platform was migrated to new DUT hardware at the end of 2016.

When the VSPERF CI results were examined [VSPERF-b2b], several aspects of the results were considered notable:

1. Back-to-back Frame Benchmark was very consistent for some fixed frame sizes, and somewhat variable for others.

2. The number of Back-to-back Frames with zero loss reported for large frame sizes was unexpectedly long (translating to 30 seconds of buffer time), and no explanation or measurement limit condition was indicated.

3. Calculation of the extent of buffer time in the DUT helped to explain the results observed with all frame sizes (for example, some frame sizes cannot exceed the frame header processing rate of the DUT and therefore no buffering occurs, therefore the results depended on the test equipment and not the DUT).

4. It was found that the actual buffer time in the DUT could be estimated using results from the Throughput tests conducted according to Section 26.1 of [RFC2544]. It is apparent that the DUT's frame processing rate tends to increase the "implied" estimate (measured according to Section 26.4 of [RFC2544]), and a calculation using the Throughput measurement can reveal a "corrected" estimate.

Further, if the Throughput tests of Section 26.1 of [RFC2544] are conducted as a prerequisite test, the number of frame sizes required for Back-to-back Frame Benchmarking can be reduced to one or more of the small frame sizes, or the results for large frame sizes can be noted as invalid in the results if tested anyway (these are the frame sizes for which the back-to-back frame rate cannot exceed the frame header processing rate of the DUT and no buffering occurs).

[VSPERF-b2b] provides the details of the calculation to estimate the actual buffer storage available in the DUT, using results from the Throughput tests for each frame size, and the maximum theoretical frame rate for the DUT links (which constrain the minimum frame spacing).

The simplified model used in these calculations for the DUT includes a packet header processing function with limited rate of operation, as shown below:

```
                     |------------ DUT --------|
Generator -> Ingress -> Buffer -> HeaderProc -> Egress -> Receiver
```

So, in the back2back frame testing:

1. The Ingress burst arrives at Max Theoretical Frame Rate, and initially the frames are buffered

2. The packet header processing function (HeaderProc) operates at approximately the "Measured Throughput", removing frames from the buffer

3. Frames that have been processed are clearly not in the buffer, so the Corrected DUT buffer time equation (Section 5.4) estimates and removes the frames that the DUT forwarded on Egress during the burst.

Knowledge of approximate buffer storage size (in time or bytes) may be useful to estimate whether frame losses will occur if DUT forwarding is temporarily suspended in a production deployment, due to an unexpected interruption of frame processing (an interruption of duration greater than the estimated buffer would certainly cause lost frames).

The presentation of OPNFV VSPERF evaluation and development of enhanced search alogorithms [VSPERF-BSLV] was discussed at IETF-102. The enhancements are intended to compensate for transient inerrrupts that may cause loss at near-Throughput levels of offered load. Subsequent analysis of the results indicates that buffers within the DUT can compensate for some interrupts, and this finding increases the importance of the Back-to-back frame characterization described here.

4. Prerequisites

The Test Setup MUST be consistent with Figure 1 of [RFC2544], or Figure 2 when the tester's sender and receiver are different devices. Other mandatory testing aspects described in [RFC2544] MUST be included, unless explicitly modified in the next section.

The ingress and egress link speeds and link layer protocols MUST be specified and used to compute the maximum theoretical frame rate when respecting the minimum inter-frame gap.

The test results for the Throughput Benchmark conducted according to Section 26.1 of [RFC2544] for all [RFC2544]-RECOMMENDED frame sizes MUST be available to reduce the tested frame size list, or to note invalid results for individual frame sizes (because the burst length may be essentially infinite for large frame sizes).

Note that:

- the Throughput and the Back-to-back Frame measurement configuration traffic characteristics (unidirectional or bi-directional) MUST match.

- the Throughput measurement MUST be under zero-loss conditions, according to Section 26.1 of [RFC2544].

The Back-to-back Benchmark described in Section 3.1 of [RFC1242] MUST be measured directly by the tester, where buffer size is inferred from packet loss measurements. Therefore, sources of packet loss that are un-related to consistent evaluation of buffer size SHOULD be identified and removed or mitigated. Example sources include:

- On-path active components that are external to the DUT

- Operating system environment interrupting DUT operation

- Shared resource contention between the DUT and other off-path component(s), impacting DUT's behaviour, sometimes called the "noisy neighbour" problem.

Mitigations applicable to some of the sources above are discussed in Section 5.2, with the other measurement requirements described below in Section 5.

5. Back-to-back Frames

Objective: To characterize the ability of a DUT to process back-to-back frames as defined in [RFC1242].

The Procedure follows.

5.1. Preparing the list of Frame sizes

From the list of RECOMMENDED Frame sizes (Section 9 of [RFC2544]), select the subset of Frame sizes whose measured Throughput was less than the maximum theoretical Frame Rate. These are the only Frame sizes where it is possible to produce a burst of frames that cause the DUT buffers to fill and eventually overflow, producing one or more discarded frames.

5.2. Test for a Single Frame Size

Each trial in the test requires the tester to send a burst of frames (after idle time) with the minimum inter-frame gap, and to count the corresponding frames forwarded by the DUT.

The duration of the trial MUST be at least 2 seconds, to allow DUT buffers to deplete.

If all frames have been received, the tester increases the length of the burst according to the search algorithm and performs another trial.

If the received frame count is less than the number of frames in the burst, then the limit of DUT processing and buffering may have been exceeded, and the burst length is determined by the search algorithm for the next trial.

Classic search algorithms have been adapted for use in benchmarking, where the search requires discovery of a pair of outcomes, one with no loss and another with loss, at load conditions within the acceptable tolerance. Also for conditions encountered when benchmarking the Infrastructure for Network Function Virtualization require algorithm enhancement. Fortunately, the adaptation of Binary Search, and an enhanced Binary Search with Loss Verification have been specified in [TST009]. These alogorithms (see clause 12.3) can easily be used for Back-to-back Frame benchmarking by replacing the Offered Load level with burst length in frames. [TST009] Annex B describes the theory behind the enhanced Binary Search algorithm.

There is also promising work-in-progress that may prove useful in for Back-to-back Frame benchmarking.

[I-D.vpolak-mkonstan-bmwg-mlrsearch] and [I-D.vpolak-bmwg-plrsearch] are two such examples.

Either the [TST009] Binary Search or Binary Search with Loss Verification algorithms MUST be used, and input parameters to the algorithm(s) MUST be reported.

The Back-to-back Frame value is the longest burst of frames that the DUT can successfully process and buffer without frame loss, as determined from the series of trials. The tester may impose a (configurable) minimum step size for burst length, and the step size MUST be reported with the results (as this influences the accuracy and variation of test results).

5.3. Test Repetition

The test MUST be repeated N times for each frame size in the subset list, and each Back-to-back Frame value made available for further processing (below).

5.4. Benchmark Calculations

For each Frame size, calculate the following summary statistics for Back-to-back Frame values over the N tests:

- o Average (Benchmark)
- o Minimum
- o Maximum
- o Standard Deviation

Further, calculate the Implied DUT Buffer Time and the Corrected DUT Buffer Time in seconds, as follows:

Implied DUT Buffer Time =

Average num of Back-to-back Frames / Max Theoretical Frame Rate

The formula above is simply expressing the Burst of Frames in units of time.

The next step is to apply a correction factor that accounts for the DUT's frame forwarding operation during the test (assuming the simple model of the DUT composed of a buffer and a forwarding function, described in Section 3).

Corrected DUT Buffer Time =

```
                                  Measured Throughput
     = Implied DUT Buffer Time * --------------------------
                                Max Theoretical Frame Rate
```

where:

1. The "Measured Throughput" is the [RFC2544] Throughput Benchmark for the frame size tested, as augmented by methods including the Binary Search with Loss Verification aglorithm in [TST009] where applicable, and MUST be expressed in Frames per second in this equation.

2. The "Max Theoretical Frame Rate" is a calculated value for the interface speed and link layer technology used, and MUST be expressed in Frames per second in this equation.

The term on the far right in the formula for Corrected DUT Buffer Time accounts for all the frames in the Burst that were transmitted by the DUT *while the Burst of frames were sent in*. So, these frames are not in the Buffer and the Buffer size is more accurately estimated by excluding them.

6. Reporting

The back-to-back results SHOULD be reported in the format of a table with a row for each of the tested frame sizes. There SHOULD be columns for the frame size and for the resultant average frame count for each type of data stream tested.

The number of tests Averaged for the Benchmark, N, MUST be reported.

The Minimum, Maximum, and Standard Deviation across all complete tests SHOULD also be reported (they are referred to as "Min,Max,StdDev" in the table below).

The Corrected DUT Buffer Time SHOULD also be reported.

If the tester operates using a maximum burst length in frames, then this maximum length SHOULD be reported.

Frame Size, octets	Ave B2B Length, frames	Min,Max,StdDev	Corrected Buff Time, Sec
64	26000	25500,27000,20	0.00004

Back-to-Back Frame Results

Static and configuration parameters:

Number of test repetitions, N

Minimum Step Size (during searches), in frames.

If the tester has an actual frame rate of interest (less than the Throughput rate), it is useful to estimate the buffer time at that frame rate:

```
Actual Buffer Time =
                                      Measured Throughput
     = Corrected DUT Buffer Time * --------------------------
                                       Actual Frame Rate
```

and report this value, properly labeled.

7. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization using controlled stimuli in a laboratory environment, with dedicated address space and the other constraints of[RFC2544].

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network, or misroute traffic to the test management network. See [RFC6815].

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

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

8. IANA Considerations

This memo makes no requests of IANA.

9. Acknowledgements

Thanks to Trevor Cooper, Sridhar Rao, and Martin Klozik of the VSPERF project for many contributions to the testing [VSPERF-b2b]. Yoshiaki Itou has also investigated the topic, and made useful suggestions. Maciek Konstantyowicz and Vratko Polak also provided many comments and suggestions based on extensive integration testing and resulting search algorithm proposals - the most up-to-date feedback possible. Tim Carlin also provided comments and support for the draft.

10. References

10.1. Normative References

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

[RFC1944] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 1944, DOI 10.17487/RFC1944, May 1996, <https://www.rfc-editor.org/info/rfc1944>.

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

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

[RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, DOI 10.17487/RFC5180, May 2008, <https://www.rfc-editor.org/info/rfc5180>.

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

[RFC6815] Bradner, S., Dubray, K., McQuaid, J., and A. Morton, "Applicability Statement for RFC 2544: Use on Production Networks Considered Harmful", RFC 6815, DOI 10.17487/RFC6815, November 2012, <https://www.rfc-editor.org/info/rfc6815>.

[RFC6985] Morton, A., "IMIX Genome: Specification of Variable Packet Sizes for Additional Testing", RFC 6985, DOI 10.17487/RFC6985, July 2013, <https://www.rfc-editor.org/info/rfc6985>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[I-D.vpolak-bmwg-plrsearch]
Konstantynowicz, M. and V. Polak, "Probabilistic Loss Ratio Search for Packet Throughput (PLRsearch)", draft-vpolak-bmwg-plrsearch-02 (work in progress), July 2019.

[I-D.vpolak-mkonstan-bmwg-mlrsearch]
Konstantynowicz, M. and V. Polak, "Multiple Loss Ratio Search for Packet Throughput (MLRsearch)", draft-vpolak-mkonstan-bmwg-mlrsearch-02 (work in progress), July 2019.

[OPNFV-2017]
Cooper, T., Morton, A., and S. Rao, "Dataplane Performance, Capacity, and Benchmarking in OPNFV", June 2017, <https://wiki.opnfv.org/download/attachments/10293193/VSPERF-Dataplane-Perf-Cap-Bench.pptx?api=v2>.

[RFC8239] Avramov, L. and J. Rapp, "Data Center Benchmarking Methodology", RFC 8239, DOI 10.17487/RFC8239, August 2017, <https://www.rfc-editor.org/info/rfc8239>.

[TST009] Morton, R. A., "ETSI GS NFV-TST 009 V3.2.1 (2019-06), "Network Functions Virtualisation (NFV) Release 3; Testing; Specification of Networking Benchmarks and Measurement Methods for NFVI"", June 2019, <https://www.etsi.org/deliver/etsi_gs/NFV-TST/001_099/009/03.01.01_60/gs_NFV-TST009v030101p.pdf>.

[VSPERF-b2b]
Morton, A., "Back2Back Testing Time Series (from CI)", June 2017, <https://wiki.opnfv.org/display/vsperf/Traffic+Generator+Testing#TrafficGeneratorTesting-AppendixB:Back2BackTestingTimeSeries(fromCI)>.

[VSPERF-BSLV]
Morton, A. and S. Rao, "Evolution of Repeatability in Benchmarking: Fraser Plugfest (Summary for IETF BMWG)", July 2018, <https://datatracker.ietf.org/meeting/102/materials/slides-102-bmwg-evolution-of-repeatability-in-benchmarking-fraser-plugfest-summary-for-ietf-bmwg-00>.

Author's Address

Al Morton
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 12, 2020
S. Jacob, Ed.
K. Tiruveedhula
Juniper Networks
March 11, 2020

Benchmarking Methodology for EVPN and PBB-EVPN
draft-ietf-bmwg-evpntest-05

Abstract

This document defines methodologies for benchmarking EVPN and PBB-EVPN performance.EVPN is defined in RFC 7432, and is being deployed in Service Provider networks.Specifically, this document defines the methodologies for benchmarking EVPN/PBB-EVPN convergence, data plane performance, and control plane performance.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

EVPN is defined in RFC 7432, and describes BGP MPLS based Ethernet VPNs (EVPN). PBB-EVPN is defined in RFC 7623, discusses how Ethernet Provider backbone Bridging can be combined with EVPNs to provide a new/combined solution. This draft defines methodologies that can be used to benchmark both RFC 7432 and RFC 7623 solutions.Further, this draft provides methodologies for benchmarking the performance of

EVPN data and control planes, MAC learning, MAC flushing, MAC aging, convergence, high availability, and scale.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 8174 [RFC8174].

1.2. Terminologies

All-Active Redundancy Mode: When all PEs attached to an Ethernet segment are allowed to forward known unicast traffic to/from that Ethernet segment for a given VLAN, then the Ethernet segment is defined to be operating in All-Active redundancy mode.

AA: All Active mode

CE: Customer Router/Devices/Switch.

DF: Designated Forwarder

DUT: Device under test.

Ethernet Segment (ES): When a customer site (device or network) is connected to one or more PEs via a set of Ethernet links, then that set of links is referred to as an 'Ethernet segment'.

EVI: An EVPN instance spanning the Provider Edge (PE) devices participating in that EVPN.

Ethernet Segment Identifier (ESI): A unique non-zero identifier that identifies an Ethernet segment is called an 'Ethernet Segment Identifier'.

Ethernet Tag: An Ethernet tag identifies a particular broadcast domain, e.g., a VLAN. An EVPN instance consists of one or more broadcast domains.

Interface: Physical interface of a router/switch.

IRB: Integrated routing and bridging interface

MAC: Media Access Control addresses on a PE.

MHPE2: Multi homed Provider Edge router 2.

MHPE1: Multi homed Provider Edge router 1.

SHPE3: Single homed Provider Edge Router 3.

PE: Provider Edge device.

P: Provider Router.

RR: Route Reflector.

RT: Traffic Generator.

Sub Interface: Each physical Interfaces is subdivided into Logical units.

SA: Single Active

Single-Active Redundancy Mode: When only a single PE, among all the PEs attached to an Ethernet segment, is allowed to forward traffic to/from that Ethernet segment for a given VLAN, then the Ethernet segment is defined to be operating in Single-Active redundancy mode.

2. Test Topology

There are five routers in the Test setup.SHPE3, RR/P, MHPE1 and MHPE2 emulating a service provider network.CE is a customer device connected to MHPE1 and MHPE2, it is configured with bridge domains in multiple vlans. The traffic generator is connected to CE and SHPE3.The MHPE1 acts as DUT.The traffic generator will be used as sender and receiver of traffic.The DUT will be the reference point for all the test cases. MHPE1 and MHPE2 are mulihome routers connected to CE running single active mode.The traffic generator will be generating traffic at 10% of the line rate.

```
             +-----------------+                  +--------------------------+
             |                 |                  |                          |
             |                 |                  |Traffic Generator sender/ |
             |  SHPE3          |                  |receiver of layer 2 traffic|
             |                 +------------------+  with multiple Vlans     |
             |                 |                  +--------------------------+
             +---------+-------+
                       |  Core Link
                       |
              +--------+-----+
              |              |
              |   RR/P       |
              |              |          Core link
              |              +-----------------+
              +--+-----------+                 |
                 |                             |
                 |    core link                |
                 |                             |
   +-------------+---+                        ++-------------------+
   |                 |                        |                    |
   |                 |                        |   MHPE2            |
   |MHPE1(DUT)       |                        |                    |
   |                 |                        |                    |
   |                 |                        |                    |
   +-----------------+------+            +-----+-------------------+
                            |            |
          PE-CE link        |            | PE-CE link
                            |            |
                            |            |
                            |            |
                      +-----+----------+----+          +------------------------
----+
                      | CE/Layer 2 bridge   +------------| Traffic Generator  send
er/ |
                      |                     |          |receiver  of layer 2 tra
ffic|
                      |                     |          | with multiple Vlans
    |
                      |                     |          +------------------------
----+
                      |                     |
                      +---------------------+
```

Topology 1

Test Setup

Figure 1

```
+-----------------+---------------------+---------------------+------------------
---+----------------------+-----------------------+
|                 |                     |                     |
   |                      |                       |
|                 |                     |                     |
   |                      |                       |
|                 |                     |                     |
   |                      |                       |
|                 |                     |                     |
   |                      |                       |
| Mode            |                     |                     |
   |Receiver              |                       |
|                 |  Test               |Traffic Direction    |Sender
   |                      |                       |
|                 |                     |                     |
   |                      |                       |
|                 |                     |                     |
   |                      |                       |
|                 |                     |                     |
   |                      |                       |
+--------------------------------------------------------------------------------
--------------------------------------------------+
|                 |                     |                     |
   |                      |                       |
|                 |                     |                     |
   |      SHPE3           |                       |
|Single Active    |  Local MAC          |                     |CE
   |                      |Layer 2 traffic        |
|                 | Learning            | Uni                 |
   |                      |                       |
|                 |                     |                     |
   |                      | multiple MAC          |
|                 |                     |                     |
   |                      |                       |
+--------------------------------------------------------------------------------
---------------------------------------------------+
|                 |                     |                     |
   |                      |                       |
|Single Active    | Remote MAC          |                     |
   |         CE           |Layer 2 traffic        |
|                 | Learning            | uni                 | SHPE3
   |                      |                       |
|                 |                     |                     |
   |                      |multiple MAC           |
|                 |                     |                     |
   |                      |                      ++
+--------------------------------------------------------------------------------
--------------------------------------------------+
|                 |                     |                     |
   |                      |                       |
|Single Active    | Scale Convergence   |   Bi                |
   |  CE/SHPE3            |                       |
|                 |                     |                     |   CE/SHPE3
   |                      |Layer 2 traffic        |
|                 | Local& Remote       |                     |
   |                      |multiple MAC& vlans    |
|                 | Learning            |                     |
   |                      |                       |
+-----------------+---------------------+---------------------+------------------
```

```
--------------------------+-----------------------+

   |

  ++
```

Table showing the traffic directions of various EVPN/PBB-EVPN benchmarking test cases. Depends on the test scenario the traffic can be uni/bi directional generated by the traffic generator.

Figure 2

Test Setup Configurations:

SHPE3 is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support.This router must be configured with N EVPN/PBB-EVPN instances for testing.Traffic generator is connected to this router for sending and receiving traffic.

RR is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support.This router acts as a provider router and as a route reflector.

MHPE1 is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support.This router must be configured with N EVPN/PBB-EVPN instances for testing.This router is configured with ESI per vlan or ESI per interface.It is functioning as multi homing PE working on Single Active EVPN mode.This router serves as the DUT and it is connected to CE.MHPE1 is acting as DUT for all the test cases.

MHPE2 is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support.This router must be configured with N EVPN/PBB-EVPN instances for testing.This router is configured with ESI per vlan or ESI per interface.It is functioning as multi homing PE working on Single Active EVPN mode. It is connected to CE.

CE is acting as bridge configured with multiple vlans,the same vlans are configured on MHPE1,MHPE2,SHPE3.Traffic generator is connected to CE. the traffic generator acts as sender or receiver of traffic.

Depending up on the test scenarios the traffic generators will be used to generate uni directional or bi directional flows.

The above configuration will be serving as the base configuration for all test cases.

3. Test Cases for EVPN Benchmarking

3.1. Data Plane MAC Learning

Objective:

Measure the time taken to learn the Data Plane MAC in DUT.

Topology : Topology 1

Procedure:

The data plane MAC learning can be measured using the parameters defined in RFC 2889 section 5.8.

Confirm the DUT is up and running with EVPN.

Traffic generator connected to CE must send frames with "X" different source and destination MAC address for one vlan, the same vlan must be present in all the devices except RR.

Send "X" unicast frames from CE to MHPE1(DUT) for one EVPN instance working in SA mode.

The DUT will learn these "X" MAC in data plane.

Measurement :

Measure the time taken to learn "X" MAC locally in DUT evpn MAC table.The data plane measurement is taken by considering DUT as black box. The range of MAC are known from traffic generator,the same must be learned in DUT, the time taken to learn "X" MAC is measured.The measurement is carried out using external server which polls the DUT using automated scripts.

The test is repeated for "N" times and the values are collected.The MAC learning rate is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.

MAC learning rate = (T1+T2+..Tn)/N

3.2. Control Plane MAC Learning

Objective:

Measure the time taken to learn the control plane MAC.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Traffic generator connected to SHPE3 must send frames with "X" different source and destination MAC address for one vlan, the same vlan must be present in all the devices except RR.

Ensure the frames must be destined to one EVPN instance.

The DUT will learn these "X" MAC in control plane.

Measurement :

Measure the time taken by the DUT to learn the "X" MAC in the data plane.The test is repeated for "N" times and the values are collected. The remote MAC learning rate is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

MAC learning rate = (T1+T2+..Tn)/N

3.3. MAC Flush-Local Link Failure and Relearning

Objective:

Measure the time taken to flush the Data Plane MAC and the time taken to relearn the same amount of MAC.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X frames with X different source and destination MAC addresses to DUT from CE using traffic generator for one vlan.

Ensure the DUT learns all X MAC addresses in data plane.

Fail the DUT-CE link and measure the time taken to flush these X MAC from the EVPN MAC table.

Bring up the link which was made Down(the link between DUT and CE).Measure time taken by the DUT to relearn these "X" MAC.

The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken for flushing these X MAC addresses. Measure the time taken to relearn these X MAC in DUT.The test is repeated for "N" times and the values are collected. The flush and the relearning time is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Flush rate = (T1+T2+..Tn)/N

Relearning rate = (T1+T2+..Tn)/N

3.4. MAC Flush-Remote Link Failure and Relearning.

Objective:

Measure the time taken to flush the Control plane MAC learned in DUT during remote link failure and the time taken to relearn.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X frames with X different source and destination MAC addresses to DUT from SHPE3 using traffic generator for one vlan.

Bring down the link between SHPE3 and traffic generator.

SHPE3 will withdraw the routes from DUT due to link failure.

Measure the time taken to flush the DUT EVPN MAC table. The DUT and MHPE2 are running SA mode.

Bring up the link which was made Down(the link between SHPE3 and traffic generator).

Measure time taken by the DUT to relearn these "X" MAC from control plane.

Measurement :

Measure the time taken to flush X remote MAC from EVPN MAC table of the DUT.Measure the time taken to relearn these X MAC in DUT.The test is repeated for "N" times and the values are collected.The flush rate is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Flush rate = (T1+T2+..Tn)/N

Relearning rate = (T1+T2+..Tn)/N

3.5. MAC Aging

Objective:

To measure the MAC aging time.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X frames with X different source and destination MAC addresses to DUT from CE using traffic generator for one vlan.

Ensure these X MAC addresses are learned in DUT.

Then stop the traffic.

Ensure the DUT and other devices in the test are using the default timers for aging.

Measure the time taken to flush X MAC from DUT EVPN MAC table due to aging.

The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to flush X MAC addresses due to aging. The test is repeated for "N" times and the values are collected. The aging is calculated averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts.

Aging time for X MAC in sec = (T1+T2+..Tn)/N

3.6. Remote MAC Aging

Objective:

Measure the control plane learned MAC aging time.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X frames with X different source and destination MAC addresses to DUT from SHPE3 using traffic generator for one vlan.

Ensure these X MAC addresses are learned in DUT via control plane.

Then stop the traffic.

Ensure the DUT and other devices in the test are using the default timers for aging.

Measure the time taken to flush X MAC from DUT EVPN MAC table due to aging.

The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to flush X remote MAC learned in DUT EVPN MAC table due to aging.The test is repeated for "N" times and the values are collected.The aging is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Aging time for X MAC in sec = (T1+T2+..Tn)/N

3.7. Control and Data plane MAC Learning

Objective:

To record the time taken to learn both local and remote MAC.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X frames with X different source and destination MAC addresses to DUT from SHPE3 using traffic generator for one vlan.

Send X frames with different source and destination MAC addresses from traffic generator connected to CE for one vlan.

The source and destination addresses of flows must be complimentary to have unicast flows.

Measure the time taken by the DUT to learn 2X in EVPN MAC table.

DUT and MHPE2 are running in SA mode.

Measurement :

Measure the time taken to learn 2X MAC addresses in DUT EVPN MAC table.The test is repeated for "N" times and the values are collected. The MAC learning time is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts

MAC learning rate = (T1+T2+..Tn)/N

3.8. High Availability.

Objective:

Measure traffic loss during routing engine fail over.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X frames from CE to DUT from traffic generator with X different source and destination MAC addresses.

Send X frames from traffic generator to SHPE3 with X different source and destination MAC addresses, so that 2X MAC address will be learned in the DUT.

There is a bi directional traffic flow with X pps in each direction.

Ensure the DUT learn 2X MAC.

Then do a routing engine fail-over.

Measurement :

The expectation of the test is 0 traffic loss with no change in the DF role. DUT should not withdraw any routes.But in cases where the DUT is not property synchronized between master and standby,due to that packet loss are observed. In that scenario the packet loss is measured.The test is repeated for "N" times and the values are collected.The packet loss is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts to ensure the DUT learned 2X MAC. The packet drop is measured using traffic generator.

Packet loss in sec with 2X MAC addresses = (T1+T2+..Tn)/N

3.9. ARP/ND Scale

Measure the DUT scaling limit of ARP/ND.

Objective:

Measure the ARP/ND scale of the DUT.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Send X arp/neighbor discovery(ND) from the traffic generator to DUT with different sender ip/ipv6,MAC addresses to the target IRB address configured in EVPN instance.

The EVPN instance learns the MAC+ip and MAC+ipv6 addresses from these request and advertise as type 2 MAC+ip/MAC+ipv6 route to remote provide edge routers which have same EVPN configurations.

The value of X must be increased at a incremental value of 5% of X, till the limit is reached.The limit is where the DUT cant learn any more type 2 MAC+ip/MAC+ipv6.The test must be separately conducted for arp and ND.

Measurement :

Measure the scale limit of type 2 MAC+ip/MAC+ipv6 route which DUT can learn.The test is repeated for "N" times and the values are collected.The scale limit is calculated by averaging the values obtained by "N" samples for both MAC+ip and MAC+ipv6."N" is an arbitrary number to get a sufficient sample.The scale value obtained by each sample be v1,v2..vn.The measurement is carried out using external server which polls the DUT using automated scripts to find the scale limit of MAC+ipv4/MAC+ipv6.

Scale limit for MAC+ip = (v1+v2+..vn)/N

Scale limit for MAC+ipv6 = (v1+v2+..vn)/N

3.10. Scaling of Services

Objective:

Measure the scale of EVPN instances that a DUT can hold.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

The DUT,MHPE2 and SHPE3 are scaled to "N" EVI.

Ensure routes received from MHPE2 and SHPE3 for "N" EVI in the DUT.

Then increment the scale of N by 5% of N till the limit is reached.

The limit is where the DUT cant learn any EVPN routes from its peers.

Measurement :

There should not be any loss of route types 1,2,3 and 4 in DUT. DUT must relearn all type 1,2,3 and 4 from remote routers.The DUT must be subjected to various values of N to find the optimal scale limit. The scope of the test is find out the maximum evpn instance that a DUT can hold.The measurement is carried out using external server which polls the DUT using automated scripts to find the scale limit of EVPN instances.

3.11. Scale Convergence

Objective:

Measure the convergence time of DUT when the DUT is scaled with EVPN instance along with traffic.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Scale N EVIs in DUT,SHPE3 and MHPE2.

Send F frames to DUT from CE using traffic generator with X different source and destination MAC addresses for N EVI's.

Send F frames from traffic generator to SHPE3 with X different source and destination MAC addresses.

There will be 2X number of MAC addresses will be learned in DUT EVPN MAC table.

There is a bi directional traffic flow with F pps in each direction.

Then clear the BGP neighbors in the DUT.

Once the BGP session is in established state in DUT.

Measure the time taken to learn 2X MAC address in DUT MAC table.

Measurement :

The DUT must learn 2X MAC addresses. Measure the time taken to learn 2X MAC in DUT.The test is repeated for "N" times and the values are collected. The convergence time is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Time taken to learn 2X MAC in DUT = (T1+T2+..Tn)/N

3.12. SOAK Test.

Objective:

This test is carried out to measure the stability of the DUT in a scaled environment with traffic over a period of time "T’". In each interval "t1" the DUT CPU usage, memory usage are measured.The DUT is checked for any crashes during this time period.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN.

Scale N EVI’s in DUT,SHPE3 and MHPE2.Send F frames to DUT from CE using traffic generator with different X source and destination MAC addresses for N EVI’s.

Send F frames from traffic generator to SHPE3 with X different source and destination MAC addresses.

There will be 2X number of MAC addresses will be learned in DUT EVPN MAC table.

There is a bi directional traffic flow with F pps in each direction.

The DUT must run with traffic for 24 hours.

Every hour check for memory leak in EVPN process, CPU usage and crashes in DUT.

Measurement :

Take the hourly reading of CPU, process memory. There should not be any leak, crashes, CPU spikes. Th CPU spike is determined as the CPU usage which shoots at 40 to 50 percent of the average usage. The average value vary from device to device. Memory leak is determined by increase usage of the memory for EVPN process. The expectation is under steady state the memory usage for EVPN process should not increase. The measurement is carried out using external server which polls the DUT using automated scripts which captures the CPU usage and process memory.

4. Test Cases for PBB-EVPN Benchmarking

4.1. Data Plane Local MAC Learning

Objective:

Measure the time taken to learn the Data Plane MAC in DUT.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Traffic generator connected to CE must send frames with "X" different source and destination MAC address for one vlan, the same vlan must be present in all the devices except RR.

Send "X" unicast frames from CE to MHPE1(DUT) for one PBB-EVPN instance working in SA mode.

The DUT will learn these "X" MAC in data plane.

Measurement :

Measure the time taken to learn "X" MAC locally in DUT PBB-EVPN MAC table. The data plane measurement is taken by considering DUT as black box. The range of MAC are known from traffic generator,the same must be learned in DUT, the time taken to learn "X" MAC is measured.The measurement is carried out using external server which polls the DUT using automated scripts.

The test is repeated for "N" times and the values are collected.The MAC learning rate is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.

MAC learning rate = (T1+T2+..Tn)/N

4.2. Data Plane Remote MAC Learning

Objective:

To Record the time taken to learn the remote MAC.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Traffic generator connected to SHPE3 must send frames with "X" different source and destination MAC address for one vlan, the same vlan must be present in all the devices except RR.

Ensure the frames must be destined to one PBB-EVPN instance.

The DUT will learn these "X" MAC in data plane.

Measurement :

Measure the time taken by the DUT to learn the "X" MAC in the data plane.The test is repeated for "N" times and the values are collected. The remote MAC learning rate is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

MAC learning rate = (T1+T2+..Tn)/N

4.3. MAC Flush-Local Link Failure

Objective:

Measure the time taken to flush the locally learned MAC and the time taken to relearn the same amount of MAC.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Send X frames with X different source and destination MAC addresses to DUT from CE using traffic generator for one vlan.

Ensure the DUT learns all X MAC addresses in data plane.

Fail the DUT-CE link and measure the time taken to flush these X MAC from the PBB-EVPN MAC table.

Bring up the link which was made Down(the link between DUT and CE).Measure time taken by the DUT to relearn these "X" MAC.

The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken for flushing these X MAC addresses. Measure the time taken to relearn these X MAC in DUT.The test is repeated for "N" times and the values are collected. The flush and the relearning time is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts.

Flush rate = (T1+T2+..Tn)/N

Relearning rate = (T1+T2+..Tn)/N

4.4. MAC Flush-Remote Link Failure

Objective:

Measure the time taken to flush the remote MAC learned in DUT due to remote link failure and relearning it.

Topology : Topology 1

Procedure:

confirm the DUT is up and running with PBB-EVPN.

Send X frames with X different source and destination MAC addresses to DUT from SHPE3 using traffic generator for one vlan.

Bring down the link between SHPE3 and traffic generator.

Measure the time taken to flush the DUT PBB-EVPN MAC table. The DUT and MHPE2 are running SA mode.

Bring up the link which was made Down(the link between SHPE3 and traffic generator).

Measure time taken by the DUT to relearn these "X" MAC

Measurement :

Measure the time taken to flush X remote MAC from PBB-EVPN MAC table of the DUT.Measure the time taken to relearn these X MAC in DUT. The test is repeated for "N" times and the values are collected. The flush rate is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Flush rate = (T1+T2+..Tn)/N

Relearning rate = (T1+T2+..Tn)/N

4.5. MAC Aging

Objective:

Measure the MAC aging time.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Send X frames with X different source and destination MAC addresses to DUT from CE using traffic generator for one vlan.

Ensure these X MAC addresses are learned in DUT.

Then stop the traffic.

Ensure the DUT and other devices in the test are using the default timers for aging.

Measure the time taken to flush X MAC from DUT PBB-EVPN MAC table due to aging.

The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to flush X MAC addresses due to aging. The test is repeated for "N" times and the values are collected. The aging is calculated averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts.

Aging time for X MAC in sec = (T1+T2+..Tn)/N

4.6. Remote MAC Aging.

Objective:

Measure the remote MAC aging time.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Send X frames with X different source and destination MAC addresses to DUT from SHPE3 using traffic generator for one vlan.

Ensure these X MAC addresses are learned in DUT.

Then stop the traffic.

Ensure the DUT and other devices in the test are using the default timers for aging.

Measure the time taken to flush X MAC from DUT PBB-EVPN MAC table due to aging.

The DUT and MHPE2 are running SA mode.

Measurement :

Measure the time taken to flush X remote MAC learned in DUT EVPN MAC table due to aging.The test is repeated for "N" times and the values are collected. The aging is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts.

Aging time for X MAC in sec = (T1+T2+..Tn)/N

4.7. Local and Remote MAC Learning

Objective:

Measure the time taken to learn both local and remote MAC.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Send X frames with X different source and destination MAC addresses to DUT from SHPE3 using traffic generator for one vlan.

Send X frames with different source and destination MAC addresses from traffic generator connected to CE for one vlan.

The source and destination addresses of flows must be complimentary to have unicast flows.

Measure the time taken by the DUT to learn 2X in PBB-EVPN MAC table.

DUT and MHPE2 are running in SA mode.

Measurement :

Measure the time taken to learn 2X MAC addresses in DUT PBB-EVPN MAC table. The test is repeated for "N" times and the values are collected. The MAC learning time is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts

MAC learning rate = (T1+T2+..Tn)/N

4.8. High Availability

Objective:

Measure traffic loss during routing engine failover.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Send X frames from CE to DUT from traffic generator with X different source and destination MAC addresses.

Send X frames from traffic generator to SHPE3 with X different source and destination MAC addresses, so that 2X MAC address will be learned in the DUT.

There is a bi directional traffic flow with X pps in each direction.

Ensure the DUT learn 2X MAC.

Then do a routing engine fail-over.

Measurement :

The expectation of the test is 0 traffic loss with no change in the DF role. DUT should not withdraw any routes.But in cases where the DUT is not property synchronized between master and standby,due to that packet loss are observed. In that scenario the packet loss is measured.The test is repeated for "N" times and the values are collected.The packet loss is calculated by averaging the values obtained by "N" samples."N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn. The measurement is carried out using external server which polls the DUT using automated scripts to ensure the DUT learned 2X MAC. The packet drop is measured using traffic generator.

Packet loss in sec with 2X MAC addresses = (T1+T2+..Tn)/N

4.9. Scale

Objective:

Measure the scale limit of DUT for PBB-EVPN.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

The DUT,MHPE2 and SHPE3 are scaled to "N" PBB-EVI.

Ensure routes received from MHPE2 and SHPE3 for "N" PBB-EVI in the DUT.

Then increment the scale of N by 5% of N till the limit is reached.

The limit is where the DUT cant learn any EVPN routes from its peers.

Measurement :

There should not be any loss of route types 2,3 and 4 in DUT. DUT must relearn all type 2,3 and 4 from remote routers. The DUT must be subjected to various values of N to find the optimal scale limit. The scope of the test is find out the maximum evpn instance that a DUT can hold.The measurement is carried out using external server which polls the DUT using automated scripts to find the scale limit of PBB-EVPN instances.

4.10. Scale Convergence

Objective:

To measure the convergence time of DUT when the DUT is scaled with EVPN instance along with traffic.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Scale N PBB-EVIs in DUT,SHPE3 and MHPE2.

Send F frames to DUT from CE using traffic generator with X different source and destination MAC addresses for N PBB-EVI's.

Send F frames from traffic generator to SHPE3 with X different source and destination MAC addresses.

There will be 2X number of MAC addresses will be learned in DUT PBB-EVPN MAC table.

There is a bi directional traffic flow with F pps in each direction.

Then clear the BGP neighbors in the DUT.

Once the BGP session is in established state in DUT.

Measure the time taken to learn 2X MAC address in DUT MAC table.

Measurement :

The DUT must learn 2X MAC addresses. Measure the time taken to learn 2X MAC in DUT.The test is repeated for "N" times and the values are collected. The convergence time is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Time taken to learn 2X MAC in DUT = (T1+T2+..Tn)/N

4.11. Soak Test

Objective:

To measure the stability of the DUT in a scaled environment with traffic.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with PBB-EVPN.

Scale N PBB-EVI's in DUT,SHPE3 and MHPE2.Send F frames to DUT from CE using traffic generator with different X source and destination MAC addresses for N EVI's.

Send F frames from traffic generator to SHPE3 with X different source and destination MAC addresses.

There will be 2X number of MAC addresses will be learned in DUT PBB-EVPN MAC table.

There is a bi directional traffic flow with F pps in each direction.

The DUT must run with traffic for 24 hours.

Every hour check for memory leak in PBB-EVPN process,CPU usage and crashes in DUT.

Measurement :

Take the hourly reading of CPU, process memory. There should not be any leak, crashes, CPU spikes. Th CPU spike is determined as the CPU usage which shoots at 40 to 50 percent of the average usage. The average value vary from device to device. Memory leak is determined by increase usage of the memory for PBB-EVPN process. The expectation is under steady state the memory usage for PBB-EVPN process should not increase. The measurement is carried out using external server which polls the DUT using automated scripts which captures the CPU usage and process memory.

5. Acknowledgements

We would like to thank Fioccola Giuseppe of Telecom Italia reviewing our draft and commenting it. We would like to thank Sarah Banks for guiding and mentoring us.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

The benchmarking tests described in this document are limited to the performance characterization of controllers in a lab environment with isolated networks. 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.

8. References

8.1. Normative References

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

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

[RFC2899] Ginoza, S., "Request for Comments Summary RFC Numbers 2800-2899", RFC 2899, DOI 10.17487/RFC2899, May 2001, <https://www.rfc-editor.org/info/rfc2899>.

8.2. Informative References

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RFC7623] Sajassi, A., Ed., Salam, S., Bitar, N., Isaac, A., and W. Henderickx, "Provider Backbone Bridging Combined with Ethernet VPN (PBB-EVPN)", RFC 7623, DOI 10.17487/RFC7623, September 2015, <https://www.rfc-editor.org/info/rfc7623>.

Appendix A. Appendix

Authors' Addresses

Sudhin Jacob (editor)
Juniper Networks
Bangalore
India

Phone: +91 8061212543
Email: sjacob@juniper.net

Kishore Tiruveedhula
Juniper Networks
10 Technology Park Dr
Westford, MA 01886
USA

Phone: +1 9785898861
Email: kishoret@juniper.net

Benchmarking Methodology Working Group B. Balarajah
Internet-Draft
Intended status: Informational C. Rossenhoevel
Expires: September 10, 2020 EANTC AG
B. Monkman
NetSecOPEN
March 9, 2020

Benchmarking Methodology for Network Security Device Performance
draft-ietf-bmwg-ngfw-performance-03

Abstract

This document provides benchmarking terminology and methodology for next-generation network security devices including next-generation firewalls (NGFW), intrusion detection and prevention solutions (IDS/IPS) and unified threat management (UTM) implementations. This document aims to strongly improve the applicability, reproducibility, and transparency of benchmarks and to align the test methodology with today's increasingly complex layer 7 application use cases. The main areas covered in this document are test terminology, traffic profiles and benchmarking methodology for NGFWs to start with.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

15 years have passed since IETF recommended test methodology and terminology for firewalls initially ([RFC2647], [RFC3511]). The requirements for network security element performance and effectiveness have increased tremendously since then. Security function implementations have evolved to more advanced areas and have diversified into intrusion detection and prevention, threat management, analysis of encrypted traffic, etc. In an industry of growing importance, well-defined and reproducible key performance indicators (KPIs) are increasingly needed as they enable fair and reasonable comparison of network security functions. All these reasons have led to the creation of a new next-generation firewall benchmarking document.

2. Requirements

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

3. Scope

This document provides testing terminology and testing methodology for next-generation firewalls security devices. It covers security effectiveness configurations, followed by performance benchmark testing. This document focuses on advanced, realistic, and reproducible testing methods. Additionally, it describes test bed environments, test tool requirements and test result formats.

4. Test Setup

Test setup defined in this document is applicable to all benchmarking test scenarios described in Section 7.

4.1. Testbed Configuration

Testbed configuration MUST ensure that any performance implications that are discovered during the benchmark testing aren't due to the inherent physical network limitations such as number of physical links and forwarding performance capabilities (throughput and latency) of the network devise in the testbed. For this reason, this document recommends avoiding external devices such as switches and routers in the testbed wherever possible.

However, in the typical deployment, the security devices (Device Under Test/System Under Test) are connected to routers and switches which will reduce the number of entries in MAC or ARP tables of the Device Under Test/System Under Test (DUT/SUT). If MAC or ARP tables have many entries, this may impact the actual DUT/SUT performance due to MAC and ARP/ND table lookup processes. Therefore, it is RECOMMENDED to connect aggregation switches or routers between test equipment and DUT/SUT as shown in Figure 1. The aggregation switches or routers can be also used to aggregate the test equipment or DUT/SUT ports, if the numbers of used ports are mismatched between test equipment and DUT/SUT.

If the test equipment is capable of emulating layer 3 routing functionality and there is no need for test equipment port aggregation, it is RECOMMENDED to configure the test setup as shown in Figure 2.

```
    +-------------------+      +-----------+      +--------------------+
    |Aggregation Switch/|      |           |      | Aggregation Switch/|
    | Router            +------+  DUT/SUT  +------+ Router             |
    |                   |      |           |      |                    |
    +----------+--------+      +-----------+      +--------+-----------+
               |                                           |
               |                                           |
   +-----------+-----------+                  +-----------+-----------+
   |                       |                  |                       |
   | +--------------------+|                  | +--------------------+|
   | | Emulated Router(s) ||                  | | Emulated Router(s) ||
   | |     (Optional)     ||                  | |     (Optional)     ||
   | +--------------------+|                  | +--------------------+|
   | +--------------------+|                  | +--------------------+|
   | |      Clients       ||                  | |      Servers       ||
   | +--------------------+|                  | +--------------------+|
   |                       |                  |                       |
   |   Test Equipment      |                  |   Test Equipment      |
   +-----------------------+                  +-----------------------+
```

Figure 1: Testbed Setup - Option 1

```
   +-----------------------+                  +-----------------------+
   | +--------------------+|   +-----------+  | +--------------------+|
   | | Emulated Router(s) ||   |           |  | | Emulated Router(s) ||
   | |    (Optional)      |+------ DUT/SUT  +------+   (Optional)    ||
   | +--------------------+|   |           |  | +--------------------+|
   | +--------------------+|   +-----------+  | +--------------------+|
   | |     Clients        ||                  | |      Servers       ||
   | +--------------------+|                  | +--------------------+|
   |                       |                  |                       |
   |  Test Equipment       |                  |  Test Equipment       |
   +-----------------------+                  +-----------------------+
```

Figure 2: Testbed Setup - Option 2

4.2. DUT/SUT Configuration

A unique DUT/SUT configuration MUST be used for all benchmarking tests described in Section 7. Since each DUT/SUT will have their own unique configuration, users SHOULD configure their device with the same parameters and security features that would be used in the actual deployment of the device or a typical deployment in order to achieve maximum security coverage.

This document attempts to define the recommended security features which SHOULD be consistently enabled for all the benchmarking tests described in Section 7. Table 1 below describes the sets of security feature list which SHOULD be configured on the DUT/SUT.

Based on customer use case, users MAY enable or disable SSL inspection feature for "Throughput Performance with NetSecOPEN Traffic Mix" test scenario described in Section 7.1

To improve repeatability, a summary of the DUT configuration including description of all enabled DUT/SUT features MUST be published with the benchmarking results.

	NGFW	
DUT Features	RECOMMENDED	OPTIONAL
SSL Inspection	x	
IDS/IPS	x	
Web Filtering		x
Antivirus	x	
Anti Spyware	x	
Anti Botnet	x	
DLP		x
DDoS		x
Certificate Validation		x
Logging and Reporting	x	
Application Identification	x	

Table 1: DUT/SUT Feature List

In summary, DUT/SUT SHOULD be configured as follows:

- o All security inspection enabled

- o Disposition of all flows of traffic are logged - Logging to an external device is permissible

- o Detection of Common Vulnerabilities and Exposures (CVE) matching the following characteristics when searching the National Vulnerability Database (NVD)

  * Common Vulnerability Scoring System (CVSS) Version: 2

  * CVSS V2 Metrics: AV:N/Au:N/I:C/A:C

  * AV=Attack Vector, Au=Authentication, I=Integrity and A=Availability

  * CVSS V2 Severity: High (7-10)

  * If doing a group test the published start date and published end date SHOULD be the same

- o Geographical location filtering and Application Identification and Control configured to be triggered based on a site or application from the defined traffic mix

In addition, a realistic number of access control rules (ACL) MUST be configured on the DUT/SUT. However, this is applicable only for the security devices where ACL's are configurable. This document determines the number of access policy rules for four different classes of DUT/SUT. The classification of the DUT/SUT MAY be based on its maximum supported firewall throughput performance number defined in the vendor data sheet. This document classifies the DUT/SUT in four different categories; namely Extra Small, Small, Medium, and Large.

The RECOMMENDED throughput values for the following classes are:

Extra Small (XS) - supported throughput less than 1Gbit/s

Small (S) - supported throughput less than 5Gbit/s

Medium (M) - supported throughput greater than 5Gbit/s and less than 10Gbit/s

Large (L) - supported throughput greater than 10Gbit/s

The Access Control Rules (ACL) defined in Table 2 MUST be configured from top to bottom in the correct order as shown in the table. The ACL entries MUST be configured in Forward Information Base (FIB) table of the DUT/SUT. (Note: There will be differences between how security vendors implement ACL decision making.) The configured ACL MUST NOT block the security and performance test traffic used for the benchmarking test scenarios.

				DUT/SUT Classification #rules			
Rules Type	Match Criteria	Description	Action	XS	S	M	L
Application layer	Application	Any application traffic NOT included in the test traffic	block	5	10	20	50
Transport layer	Src IP and TCP/UDP Dst ports	Any src IP subnet used in the test AND any dst ports NOT used in the test traffic	block	25	50	100	250
IP layer	Src/Dst IP	Any src/dst IP subnet NOT used in the test	block	25	50	100	250
Application layer	Application	Applications included in the test traffic	allow	10	10	10	10
Transport layer	Src IP and TCP/UDP Dst ports	Half of the src IP used in the test AND any dst ports used in the test traffic. One rule per subnet	allow	1	1	1	1
IP layer	Src IP	The rest of the src IP subnet range used in the test. One rule per subnet	allow	1	1	1	1

Table 2: DUT/SUT Access List

4.3. Test Equipment Configuration

In general, test equipment allows configuring parameters in different protocol layers. These parameters thereby influence the traffic flows which will be offered and impact performance measurements.

This section specifies common test equipment configuration parameters applicable for all test scenarios defined in Section 7. Any test scenario specific parameters are described under the test setup section of each test scenario individually.

4.3.1. Client Configuration

This section specifies which parameters SHOULD be considered while configuring clients using test equipment. Also, this section specifies the RECOMMENDED values for certain parameters.

4.3.1.1. TCP Stack Attributes

The TCP stack SHOULD use a TCP Reno [RFC5681] variant, which include congestion avoidance, back off and windowing, fast retransmission, and fast recovery on every TCP connection between client and server endpoints. The default IPv4 and IPv6 MSS segments size MUST be set to 1460 bytes and 1440 bytes respectively and a TX and RX receive windows of 64 KByte. Client initial congestion window MUST NOT exceed 10 times the MSS. Delayed ACKs are permitted and the maximum client delayed Ack MUST NOT exceed 10 times the MSS before a forced ACK. Up to 3 retries SHOULD be allowed before a timeout event is declared. All traffic MUST set the TCP PSH flag to high. The source port range SHOULD be in the range of 1024 - 65535. Internal timeout SHOULD be dynamically scalable per RFC 793. Client SHOULD initiate and close TCP connections. TCP connections MUST be closed via FIN.

4.3.1.2. Client IP Address Space

The sum of the client IP space SHOULD contain the following attributes. The IP blocks SHOULD consist of multiple unique, discontinuous static address blocks. A default gateway is permitted. The IPv4 Type of Service (ToS) byte or IPv6 traffic class should be set to '00' or '000000' respectively.

The following equation can be used to determine the required total number of client IP addresses.

Desired total number of client IP = Target throughput [Mbit/s] / Throughput per IP address [Mbit/s]

Based on deployment and use case scenario, the value for "Throughput per IP address" can be varied.

(Option 1) DUT/SUT deployment scenario 1 : 6-7 Mbit/s per IP (e.g. 1,400-1,700 IPs per 10Gbit/s throughput)

(Option 2) DUT/SUT deployment scenario 2 : 0.1-0.2 Mbit/s per IP (e.g. 50,000-100,000 IPs per 10Gbit/s throughput)

Based on deployment and use case scenario, client IP addresses SHOULD be distributed between IPv4 and IPv6 type. The Following options can be considered for a selection of traffic mix ratio.

(Option 1) 100 % IPv4, no IPv6

(Option 2) 80 % IPv4, 20% IPv6

(Option 3) 50 % IPv4, 50% IPv6

(Option 4) 20 % IPv4, 80% IPv6

(Option 5) no IPv4, 100% IPv6

4.3.1.3. Emulated Web Browser Attributes

The emulated web browser contains attributes that will materially affect how traffic is loaded. The objective is to emulate modern, typical browser attributes to improve realism of the result set.

For HTTP traffic emulation, the emulated browser MUST negotiate HTTP 1.1. HTTP persistency MAY be enabled depending on test scenario. The browser MAY open multiple TCP connections per Server endpoint IP at any time depending on how many sequential transactions are needed to be processed. Within the TCP connection multiple transactions MAY be processed if the emulated browser has available connections. The browser SHOULD advertise a User-Agent header. Headers MUST be sent uncompressed. The browser SHOULD enforce content length validation.

For encrypted traffic, the following attributes SHALL define the negotiated encryption parameters. The test clients MUST use TLSv1.2 or higher. TLS record size MAY be optimized for the HTTPS response object size up to a record size of 16 KByte. The client endpoint MUST send TLS Extension Server Name Indication (SNI) information when opening a security tunnel. Each client connection MUST perform a full handshake with server certificate and MUST NOT use session reuse or resumption. Cipher suite and key size are defined in the parameter section of the specific test scenarios.

4.3.2. Backend Server Configuration

This section specifies which parameters should be considered while configuring emulated backend servers using test equipment.

4.3.2.1. TCP Stack Attributes

The TCP stack on the server side SHOULD be configured similar to the client side configuration described in Section 4.3.1.1. In addition, server initial congestion window MUST NOT exceed 10 times the MSS. Delayed ACKs are permitted and the maximum server delayed ACK MUST NOT exceed 10 times the MSS before a forced ACK.

4.3.2.2. Server Endpoint IP Addressing

The server IP blocks SHOULD consist of unique, discontinuous static address blocks with one IP per Server Fully Qualified Domain Name (FQDN) endpoint per test port. The IPv4 ToS byte and IPv6 traffic class bytes should be set to '00' and '000000' respectively.

4.3.2.3. HTTP / HTTPS Server Pool Endpoint Attributes

The server pool for HTTP SHOULD listen on TCP port 80 and emulate HTTP version 1.1 with persistence. The Server MUST advertise server type in the Server response header [RFC2616]. For HTTPS server, TLS 1.2 or higher MUST be used with a maximum record size of 16 KByte and MUST NOT use ticket resumption or Session ID reuse . The server MUST listen on port TCP 443. The server SHALL serve a certificate to the client. It is REQUIRED that the HTTPS server also check Host SNI information with the FQDN. Cipher suite and key size are defined in the parameter section of the specific test scenarios.

4.3.3. Traffic Flow Definition

This section describes the traffic pattern between client and server endpoints. At the beginning of the test, the server endpoint initializes and will be ready to accept connection states including initialization of the TCP stack as well as bound HTTP and HTTPS servers. When a client endpoint is needed, it will initialize and be given attributes such as a MAC and IP address. The behavior of the client is to sweep though the given server IP space, sequentially generating a recognizable service by the DUT. Thus, a balanced, mesh between client endpoints and server endpoints will be generated in a client port server port combination. Each client endpoint performs the same actions as other endpoints, with the difference being the source IP of the client endpoint and the target server IP pool. The client SHALL use Fully Qualified Domain Names (FQDN) in Host Headers and for TLS Server Name Indication (SNI).

4.3.3.1. Description of Intra-Client Behavior

Client endpoints are independent of other clients that are concurrently executing. When a client endpoint initiates traffic, this section describes how the client steps though different services. Once the test is initialized, the client endpoints SHOULD randomly hold (perform no operation) for a few milliseconds to allow for better randomization of start of client traffic. Each client will either open a new TCP connection or connect to a TCP persistence stack still open to that specific server. At any point that the service profile may require encryption, a TLS encryption tunnel will form presenting the URL request to the server. The server will then perform an SNI name check with the proposed FQDN compared to the domain embedded in the certificate. Only when correct, will the server process the HTTPS response object. The initial response object to the server MUST NOT have a fixed size; its size is based on benchmarking tests described in Section 7. Multiple additional sub-URLs (response objects on the service page) MAY be requested simultaneously. This MAY be to the same server IP as the initial URL. Each sub-object will also use a conical FQDN and URL path, as observed in the traffic mix used.

4.3.4. Traffic Load Profile

The loading of traffic is described in this section. The loading of a traffic load profile has five distinct phases: Init, ramp up, sustain, ramp down, and collection.

1. During the Init phase, test bed devices including the client and server endpoints should negotiate layer 2-3 connectivity such as MAC learning and ARP. Only after successful MAC learning or ARP/ND resolution SHALL the test iteration move to the next phase. No measurements are made in this phase. The minimum RECOMMEND time for Init phase is 5 seconds. During this phase, the emulated clients SHOULD NOT initiate any sessions with the DUT/SUT, in contrast, the emulated servers should be ready to accept requests from DUT/SUT or from emulated clients.

2. In the ramp up phase, the test equipment SHOULD start to generate the test traffic. It SHOULD use a set approximate number of unique client IP addresses actively to generate traffic. The traffic SHOULD ramp from zero to desired target objective. The target objective will be defined for each benchmarking test. The duration for the ramp up phase MUST be configured long enough, so that the test equipment does not overwhelm DUT/SUT's supported performance metrics namely; connections per second, throughput, concurrent TCP connections, and application transactions per second. No measurements are made in this phase.

3. In the sustain phase, the test equipment SHOULD continue generating traffic to constant target value for a constant number of active client IPs. The mininum RECOMMENDED time duration for sustain phase is 300 seconds. This is the phase where measurements occur.

4. In the ramp down/close phase, no new connections are established, and no measurements are made. The time duration for ramp up and ramp down phase SHOULD be same.

5. The last phase is administrative and will occur when the test equipment merges and collates the report data.

5. Test Bed Considerations

This section recommends steps to control the test environment and test equipment, specifically focusing on virtualized environments and virtualized test equipment.

1. Ensure that any ancillary switching or routing functions between the system under test and the test equipment do not limit the performance of the traffic generator. This is specifically important for virtualized components (vSwitches, vRouters).

2. Verify that the performance of the test equipment matches and reasonably exceeds the expected maximum performance of the system under test.

3. Assert that the test bed characteristics are stable during the entire test session. Several factors might influence stability specifically for virtualized test beds. For example additional workloads in a virtualized system, load balancing and movement of virtual machines during the test, or simple issues such as additional heat created by high workloads leading to an emergency CPU performance reduction.

Test bed reference pre-tests help to ensure that the maximum desired traffic generator aspects such as throughput, transaction per second, connection per second, concurrent connection and latency.

Once the desired maximum performance goals for the system under test have been identified, a safety margin of 10% SHOULD be added for throughput and subtracted for maximum latency and maximum packet loss.

Test bed preparation may be performed either by configuring the DUT in the most trivial setup (fast forwarding) or without presence of DUT.

6. Reporting

This section describes how the final report should be formatted and presented. The final test report MAY have two major sections; Introduction and result sections. The following attributes SHOULD be present in the introduction section of the test report.

1. The name of the NetSecOPEN traffic mix (see Appendix A) MUST be prominent.

2. The time and date of the execution of the test MUST be prominent.

3. Summary of testbed software and Hardware details

   A. DUT Hardware/Virtual Configuration

      + This section SHOULD clearly identify the make and model of the DUT

      + The port interfaces, including speed and link information MUST be documented.

      + If the DUT is a virtual VNF, interface acceleration such as DPDK and SR-IOV MUST be documented as well as cores used, RAM used, and the pinning / resource sharing configuration. The Hypervisor and version MUST be documented.

      + Any additional hardware relevant to the DUT such as controllers MUST be documented

   B. DUT Software

      + The operating system name MUST be documented

      + The version MUST be documented

      + The specific configuration MUST be documented

   C. DUT Enabled Features

      + Configured DUT/SUT features (see Table 1) MUST be documented

      + Attributes of those featured MUST be documented

      + Any additional relevant information about features MUST be documented

D. Test equipment hardware and software

   + Test equipment vendor name

   + Hardware details including model number, interface type

   + Test equipment firmware and test application software version

4. Results Summary / Executive Summary

   1. Results SHOULD resemble a pyramid in how it is reported, with the introduction section documenting the summary of results in a prominent, easy to read block.

   2. In the result section of the test report, the following attributes should be present for each test scenario.

      a. KPIs MUST be documented separately for each test scenario. The format of the KPI metrics should be presented as described in Section 6.1.

      b. The next level of details SHOULD be graphs showing each of these metrics over the duration (sustain phase) of the test. This allows the user to see the measured performance stability changes over time.

6.1. Key Performance Indicators

This section lists KPIs for overall benchmarking tests scenarios. All KPIs MUST be measured during the sustain phase of the traffic load profile described in Section 4.3.4. All KPIs MUST be measured from the result output of test equipment.

o Concurrent TCP Connections
  This key performance indicator measures the average concurrent open TCP connections in the sustaining period.

o TCP Connections Per Second
  This key performance indicator measures the average established TCP connections per second in the sustaining period. For "TCP/HTTP(S) Connection Per Second" benchmarking test scenario, the KPI is measured average established and terminated TCP connections per second simultaneously.

o Application Transactions Per Second

This key performance indicator measures the average successfully completed application transactions per second in the sustaining period.

- TLS Handshake Rate
  This key performance indicator measures the average TLS 1.2 or higher session formation rate within the sustaining period.

- Throughput
  This key performance indicator measures the average Layer 2 throughput within the sustaining period as well as average packets per seconds within the same period. The value of throughput SHOULD be presented in Gbit/s rounded to two places of precision with a more specific Kbit/s in parenthesis. Optionally, goodput MAY also be logged as an average goodput rate measured over the same period. Goodput result SHALL also be presented in the same format as throughput.

- URL Response time / Time to Last Byte (TTLB)
  This key performance indicator measures the minimum, average and maximum per URL response time in the sustaining period. The latency is measured at Client and in this case would be the time duration between sending a GET request from Client and the receival of the complete response from the server.

- Time to First Byte (TTFB)
  This key performance indicator will measure minimum, average and maximum the time to first byte. TTFB is the elapsed time between sending the SYN packet from the client and receiving the first byte of application date from the DUT/SUT. TTFB SHOULD be expressed in millisecond.

7. Benchmarking Tests

7.1. Throughput Performance With NetSecOPEN Traffic Mix

7.1.1. Objective

Using NetSecOPEN traffic mix, determine the maximum sustainable throughput performance supported by the DUT/SUT. (see Appendix A for details about traffic mix)

This test scenario is RECOMMENDED to perform twice; one with SSL inspection feature enabled and the second scenario with SSL inspection feature disabled on the DUT/SUT.

7.1.2. Test Setup

Test bed setup MUST be configured as defined in Section 4. Any test scenario specific test bed configuration changes MUST be documented.

7.1.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.1.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.1.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be noted for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target throughput: It can be defined based on requirements. Otherwise it represents aggregated line rate of interface(s) used in the DUT/SUT

Initial throughput: 10% of the "Target throughput"

One of the following ciphers and keys are RECOMMENDED to use for this test scenarios.

1. ECHDE-ECDSA-AES128-GCM-SHA256 with Prime256v1 (Signature Hash Algorithm: ecdsa_secp256r1_sha256 and Supported group: sepc256r1)

2. ECDHE-RSA-AES128-GCM-SHA256 with RSA 2048 (Signature Hash Algorithm: rsa_pkcs1_sha256 and Supported group: sepc256)

3. ECDHE-ECDSA-AES256-GCM-SHA384 with Secp521 (Signature Hash Algorithm: ecdsa_secp384r1_sha384 and Supported group: sepc521r1)

4. ECDHE-RSA-AES256-GCM-SHA384 with RSA 4096 (Signature Hash Algorithm: rsa_pkcs1_sha384 and Supported group: secp256)

7.1.3.3. Traffic Profile

Traffic profile: Test scenario MUST be run with a single application traffic mix profile (see Appendix A for details about traffic mix). The name of the NetSecOPEN traffic mix MUST be documented.

7.1.3.4. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile.

a. Number of failed application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of total attempt transactions

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. Maximum deviation (max. dev) of URL Response Time or TTLB (Time To Last Byte) MUST be less than X (The value for "X" will be finalized and updated after completion of PoC test)
The following equation MUST be used to calculate the deviation of URL Response Time or TTLB
max. dev = max((avg_latency - min_latency),(max_latency - avg_latency)) / (Initial latency)
Where, the initial latency is calculated using the following equation. For this calculation, the latency values (min', avg' and max') MUST be measured during test procedure step 1 as defined in Section 7.1.4.1.
The variable latency represents URL Response Time or TTLB.
Initial latency:= min((avg' latency - min' latency) | (max' latency - avg' latency))

d. Maximum value of Time to First Byte (TTFB) MUST be less than X

7.1.3.5. Measurement

Following KPI metrics MUST be reported for this test scenario.

Mandatory KPIs: average Throughput, TTFB (minimum, average and maximum), TTLB (minimum, average and maximum) and average Application Transactions Per Second

Note: TTLB MUST be reported along with min, max and avg object size used in the traffic profile.

Optional KPIs: average TCP Connections Per Second and average TLS Handshake Rate

7.1.4. Test Procedures and expected Results

The test procedures are designed to measure the throughput performance of the DUT/SUT at the sustaining period of traffic load profile. The test procedure consists of three major steps.

7.1.4.1. Step 1: Test Initialization and Qualification

Verify the link status of the all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure traffic load profile of the test equipment to generate test traffic at the "Initial throughput" rate as described in the parameters Section 7.1.3.2. The test equipment SHOULD follow the traffic load profile definition as described in Section 4.3.4. The DUT/SUT SHOULD reach the "Initial throughput" during the sustain phase. Measure all KPI as defined in Section 7.1.3.5. The measured KPIs during the sustain phase MUST meet validation criteria "a" and "b" defined in Section 7.1.3.4.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to step 2.

7.1.4.2. Step 2: Test Run with Target Objective

Configure test equipment to generate traffic at the "Target throughput" rate defined in the parameter table. The test equipment SHOULD follow the traffic load profile definition as described in Section 4.3.4. The test equipment SHOULD start to measure and record all specified KPIs. The frequency of KPI metric measurements SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired target throughput during the sustain phase. In addition, the measured KPIs MUST meet all validation criteria. Follow step 3, if the KPI metrics do not meet the validation criteria.

7.1.4.3. Step 3: Test Iteration

Determine the maximum and average achievable throughput within the validation criteria. Final test iteration MUST be performed for the test duration defined in Section 4.3.4.

7.2. TCP/HTTP Connections Per Second

7.2.1. Objective

Using HTTP traffic, determine the maximum sustainable TCP connection establishment rate supported by the DUT/SUT under different throughput load conditions.

To measure connections per second, test iterations MUST use different fixed HTTP response object sizes defined in Section 7.2.3.2.

7.2.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. MUST be documented.

7.2.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.2.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.2.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target connections per second: Initial value from product data sheet (if known)

Initial connections per second: 10% of "Target connections per second" (an optional parameter for documentation)

The client SHOULD negotiate HTTP 1.1 and close the connection with FIN immediately after completion of one transaction. In each test iteration, client MUST send GET command requesting a fixed HTTP response object size.

The RECOMMENDED response object sizes are 1, 2, 4, 16, 64 KByte

7.2.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile.

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of total attempt transactions

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. During the sustain phase, traffic should be forwarded at a constant rate

d. Concurrent TCP connections MUST be constant during steady state and any deviation of concurrent TCP connections SHOULD be less than 10%. This confirms the DUT opens and closes TCP connections almost at the same rate

7.2.3.4. Measurement

Following KPI metric MUST be reported for each test iteration.

average TCP Connections Per Second

7.2.4. Test Procedures and Expected Results

The test procedure is designed to measure the TCP connections per second rate of the DUT/SUT at the sustaining period of the traffic load profile. The test procedure consists of three major steps. This test procedure MAY be repeated multiple times with different IP types; IPv4 only, IPv6 only and IPv4 and IPv6 mixed traffic distribution.

7.2.4.1. Step 1: Test Initialization and Qualification

Verify the link status of all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure the traffic load profile of the test equipment to establish "initial connections per second" as defined in the parameters Section 7.2.3.2. The traffic load profile SHOULD be defined as described in Section 4.3.4.

The DUT/SUT SHOULD reach the "Initial connections per second" before the sustain phase. The measured KPIs during the sustain phase MUST meet validation criteria a, b, c, and d defined in Section 7.2.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.2.4.2. Step 2: Test Run with Target Objective

Configure test equipment to establish "Target connections per second" defined in the parameters table. The test equipment SHOULD follow the traffic load profile definition as described in Section 4.3.4.

During the ramp up and sustain phase of each test iteration, other KPIs such as throughput, concurrent TCP connections and application transactions per second MUST NOT reach to the maximum value the DUT/SUT can support. The test results for specific test iterations SHOULD NOT be reported, if the above mentioned KPI (especially throughput) reaches the maximum value. (Example: If the test iteration with 64 KByte of HTTP response object size reached the maximum throughput limitation of the DUT, the test iteration MAY be interrupted and the result for 64 KByte SHOULD NOT be reported).

The test equipment SHOULD start to measure and record all specified KPIs. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired target connections per second rate at the sustain phase. In addition, the measured KPIs MUST meet all validation criteria.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.2.4.3. Step 3: Test Iteration

Determine the maximum and average achievable connections per second within the validation criteria.

7.3. HTTP Throughput

7.3.1. Objective

Determine the throughput for HTTP transactions varying the HTTP response object size.

7.3.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. must be documented.

7.3.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.3.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.3.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target Throughput: Initial value from product data sheet (if known)

Initial Throughput: 10% of "Target Throughput" (an optional parameter for documentation)

Number of HTTP response object requests (transactions) per connection: 10

RECOMMENDED HTTP response object size: 1 KByte, 16 KByte, 64 KByte, 256 KByte and mixed objects defined in the table

Object size (KByte)	Number of requests/ Weight
0.2	1
6	1
8	1
9	1
10	1
25	1
26	1
35	1
59	1
347	1

Table 3: Mixed Objects

7.3.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of attempt transactions.

b. Traffic should be forwarded constantly.

c. Concurrent TCP connections MUST be constant during steady state and any deviation of concurrent TCP connections SHOULD be less than 10%. This confirms the DUT opens and closes TCP connections almost at the same rate

7.3.3.4. Measurement

The KPI metrics MUST be reported for this test scenario:

average Throughput and average HTTP Transactions per Second

7.3.4. Test Procedures and Expected Results

The test procedure is designed to measure HTTP throughput of the DUT/SUT. The test procedure consists of three major steps. This test procedure MAY be repeated multiple times with different IPv4 and IPv6 traffic distribution and HTTP response object sizes.

7.3.4.1. Step 1: Test Initialization and Qualification

Verify the link status of the all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure traffic load profile of the test equipment to establish "Initial Throughput" as defined in the parameters Section 7.3.3.2.

The traffic load profile SHOULD be defined as described in Section 4.3.4. The DUT/SUT SHOULD reach the "Initial Throughput" during the sustain phase. Measure all KPI as defined in Section 7.3.3.4.

The measured KPIs during the sustain phase MUST meet the validation criteria "a" defined in Section 7.3.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.3.4.2. Step 2: Test Run with Target Objective

The test equipment SHOULD start to measure and record all specified KPIs. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired "Target Throughput" at the sustain phase. In addition, the measured KPIs must meet all validation criteria.

Perform the test separately for each HTTP response object size.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.3.4.3. Step 3: Test Iteration

Determine the maximum and average achievable throughput within the validation criteria. Final test iteration MUST be performed for the test duration defined in Section 4.3.4.

7.4. TCP/HTTP Transaction Latency

7.4.1. Objective

Using HTTP traffic, determine the average HTTP transaction latency when DUT is running with sustainable HTTP transactions per second supported by the DUT/SUT under different HTTP response object sizes.

Test iterations MUST be performed with different HTTP response object sizes in two different scenarios.one with a single transaction and the other with multiple transactions within a single TCP connection. For consistency both the single and multiple transaction test MUST be configured with HTTP 1.1.

Scenario 1: The client MUST negotiate HTTP 1.1 and close the connection with FIN immediately after completion of a single transaction (GET and RESPONSE).

Scenario 2: The client MUST negotiate HTTP 1.1 and close the connection FIN immediately after completion of 10 transactions (GET and RESPONSE) within a single TCP connection.

7.4.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. MUST be documented.

7.4.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.4.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.4.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3 . Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target objective for scenario 1: 50% of the maximum connection per second measured in test scenario TCP/HTTP Connections Per Second (Section 7.2)

Target objective for scenario 2: 50% of the maximum throughput measured in test scenario HTTP Throughput (Section 7.3)

Initial objective for scenario 1: 10% of Target objective for scenario 1" (an optional parameter for documentation)

Initial objective for scenario 2: 10% of "Target objective for scenario 2" (an optional parameter for documentation)

HTTP transaction per TCP connection: test scenario 1 with single transaction and the second scenario with 10 transactions

HTTP 1.1 with GET command requesting a single object. The RECOMMENDED object sizes are 1, 16 or 64 KByte. For each test iteration, client MUST request a single HTTP response object size.

7.4.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile. Ramp up and ramp down phase SHOULD NOT be considered.

Generic criteria:

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of attempt transactions.

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. During the sustain phase, traffic should be forwarded at a constant rate.

d. Concurrent TCP connections MUST be constant during steady state and any deviation of concurrent TCP connections SHOULD be less than 10%. This confirms the DUT opens and closes TCP connections almost at the same rate

e. After ramp up the DUT MUST achieve the "Target objective" defined in the parameter Section 7.4.3.2 and remain in that state for the entire test duration (sustain phase).

7.4.3.4. Measurement

Following KPI metrics MUST be reported for each test scenario and HTTP response object sizes separately:

TTFB (minimum, average and maximum) and TTLB (minimum, average and maximum)

All KPI's are measured once the target throughput achieves the steady state.

7.4.4. Test Procedures and Expected Results

The test procedure is designed to measure the average application transaction latencies or TTLB when the DUT is operating close to 50% of its maximum achievable throughput or connections per second. This test procedure CAN be repeated multiple times with different IP types (IPv4 only, IPv6 only and IPv4 and IPv6 mixed traffic distribution), HTTP response object sizes and single and multiple transactions per connection scenarios.

7.4.4.1. Step 1: Test Initialization and Qualification

Verify the link status of the all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure traffic load profile of the test equipment to establish "Initial objective" as defined in the parameters Section 7.4.3.2. The traffic load profile can be defined as described in Section 4.3.4.

The DUT/SUT SHOULD reach the "Initial objective" before the sustain phase. The measured KPIs during the sustain phase MUST meet the validation criteria a, b, c, d, e and f defined in Section 7.4.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.4.4.2. Step 2: Test Run with Target Objective

Configure test equipment to establish "Target objective" defined in the parameters table. The test equipment SHOULD follow the traffic load profile definition as described in Section 4.3.4.

During the ramp up and sustain phase, other KPIs such as throughput, concurrent TCP connections and application transactions per second MUST NOT reach to the maximum value that the DUT/SUT can support. The test results for specific test iterations SHOULD NOT be reported, if the above mentioned KPI (especially throughput) reaches to the maximum value. (Example: If the test iteration with 64 KByte of HTTP response object size reached the maximum throughput limitation of the DUT, the test iteration MAY be interrupted and the result for 64 KByte SHOULD NOT be reported).

The test equipment SHOULD start to measure and record all specified KPIs. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed. DUT/SUT is expected to reach the desired "Target objective" at the sustain phase. In addition, the measured KPIs MUST meet all validation criteria.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.4.4.3. Step 3: Test Iteration

Determine the maximum achievable connections per second within the validation criteria and measure the latency values.

7.5. Concurrent TCP/HTTP Connection Capacity

7.5.1. Objective

Determine the maximum number of concurrent TCP connections that the DUT/ SUT sustains when using HTTP traffic.

7.5.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. must be documented.

7.5.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.5.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.5.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be noted for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target concurrent connection: Initial value from product data sheet (if known)

Initial concurrent connection: 10% of "Target concurrent connection" (an optional parameter for documentation)

Maximum connections per second during ramp up phase: 50% of maximum connections per second measured in test scenario TCP/HTTP Connections per second (Section 7.2)

Ramp up time (in traffic load profile for "Target concurrent connection"): "Target concurrent connection" / "Maximum connections per second during ramp up phase"

Ramp up time (in traffic load profile for "Initial concurrent connection"): "Initial concurrent connection" / "Maximum connections per second during ramp up phase"

The client MUST negotiate HTTP 1.1 with persistence and each client MAY open multiple concurrent TCP connections per server endpoint IP.

Each client sends 10 GET commands requesting 1 KByte HTTP response object in the same TCP connection (10 transactions/TCP connection) and the delay (think time) between the transaction MUST be X seconds.

X = ("Ramp up time" + "steady state time") /10

The established connections SHOULD remain open until the ramp down phase of the test. During the ramp down phase, all connections SHOULD be successfully closed with FIN.

7.5.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile.

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transaction) of total attempted transactions

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. During the sustain phase, traffic SHOULD be forwarded constantly

7.5.3.4. Measurement

Following KPI metric MUST be reported for this test scenario:

average Concurrent TCP Connections

7.5.4. Test Procedures and expected Results

The test procedure is designed to measure the concurrent TCP connection capacity of the DUT/SUT at the sustaining period of traffic load profile. The test procedure consists of three major steps. This test procedure MAY be repeated multiple times with different IPv4 and IPv6 traffic distribution.

7.5.4.1. Step 1: Test Initialization and Qualification

Verify the link status of the all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure test equipment to establish "Initial concurrent TCP connections" defined in Section 7.5.3.2. Except ramp up time, the traffic load profile SHOULD be defined as described in Section 4.3.4.

During the sustain phase, the DUT/SUT SHOULD reach the "Initial concurrent TCP connections". The measured KPIs during the sustain phase MUST meet the validation criteria "a" and "b" defined in Section 7.5.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.5.4.2. Step 2: Test Run with Target Objective

Configure test equipment to establish "Target concurrent TCP connections". The test equipment SHOULD follow the traffic load profile definition (except ramp up time) as described in Section 4.3.4.

During the ramp up and sustain phase, the other KPIs such as throughput, TCP connections per second and application transactions per second MUST NOT reach to the maximum value that the DUT/SUT can support.

The test equipment SHOULD start to measure and record KPIs defined in Section 7.5.3.4. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired target concurrent connection at the sustain phase. In addition, the measured KPIs must meet all validation criteria.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.5.4.3. Step 3: Test Iteration

Determine the maximum and average achievable concurrent TCP connections capacity within the validation criteria.

7.6. TCP/HTTPS Connections per second

7.6.1. Objective

Using HTTPS traffic, determine the maximum sustainable SSL/TLS session establishment rate supported by the DUT/SUT under different throughput load conditions.

Test iterations MUST include common cipher suites and key strengths as well as forward looking stronger keys. Specific test iterations MUST include ciphers and keys defined in Section 7.6.3.2.

For each cipher suite and key strengths, test iterations MUST use a single HTTPS response object size defined in the test equipment configuration parameters Section 7.6.3.2 to measure connections per second performance under a variety of DUT Security inspection load conditions.

7.6.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. MUST be documented.

7.6.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.6.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.6.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target connections per second: Initial value from product data sheet (if known)

Initial connections per second: 10% of "Target connections per second" (an optional parameter for documentation)

RECOMMENDED ciphers and keys:

1. ECHDE-ECDSA-AES128-GCM-SHA256 with Prime256v1 (Signature Hash Algorithm: ecdsa_secp256r1_sha256 and Supported group: sepc256r1)

2. ECDHE-RSA-AES128-GCM-SHA256 with RSA 2048 (Signature Hash Algorithm: rsa_pkcs1_sha256 and Supported group: sepc256)

3. ECDHE-ECDSA-AES256-GCM-SHA384 with Secp521 (Signature Hash Algorithm: ecdsa_secp384r1_sha384 and Supported group: sepc521r1)

4. ECDHE-RSA-AES256-GCM-SHA384 with RSA 4096 (Signature Hash Algorithm: rsa_pkcs1_sha384 and Supported group: secp256)

The client MUST negotiate HTTPS 1.1 and close the connection with FIN immediately after completion of one transaction. In each test iteration, client MUST send GET command requesting a fixed HTTPS response object size. The RECOMMENDED object sizes are 1, 2, 4, 16, 64 KByte.

7.6.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria:

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of attempt transactions

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. During the sustain phase, traffic should be forwarded at a constant rate

d. Concurrent TCP connections MUST be constant during steady state and any deviation of concurrent TCP connections SHOULD be less than 10%. This confirms the DUT opens and closes TCP connections almost at the same rate

7.6.3.4. Measurement

Following KPI metrics MUST be reported for this test scenario:

average TCP Connections Per Second, average TLS Handshake Rate (TLS Handshake Rate can be measured in the test scenario using 1KB object size)

7.6.4. Test Procedures and expected Results

The test procedure is designed to measure the TCP connections per second rate of the DUT/SUT at the sustaining period of traffic load profile. The test procedure consists of three major steps. This test procedure MAY be repeated multiple times with different IPv4 and IPv6 traffic distribution.

7.6.4.1. Step 1: Test Initialization and Qualification

Verify the link status of all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure traffic load profile of the test equipment to establish "Initial connections per second" as defined in Section 7.6.3.2. The traffic load profile CAN be defined as described in Section 4.3.4.

The DUT/SUT SHOULD reach the "Initial connections per second" before the sustain phase. The measured KPIs during the sustain phase MUST meet the validation criteria a, b, c, and d defined in Section 7.6.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.6.4.2. Step 2: Test Run with Target Objective

Configure test equipment to establish "Target connections per second" defined in the parameters table. The test equipment SHOULD follow the traffic load profile definition as described in Section 4.3.4.

During the ramp up and sustain phase, other KPIs such as throughput, concurrent TCP connections and application transactions per second MUST NOT reach the maximum value that the DUT/SUT can support. The test results for specific test iteration SHOULD NOT be reported, if the above mentioned KPI (especially throughput) reaches the maximum value. (Example: If the test iteration with 64 KByte of HTTPS response object size reached the maximum throughput limitation of the DUT, the test iteration can be interrupted and the result for 64 KByte SHOULD NOT be reported).

The test equipment SHOULD start to measure and record all specified KPIs. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired target connections per second rate at the sustain phase. In addition, the measured KPIs must meet all validation criteria.

Follow the step 3, if the KPI metrics do not meet the validation criteria.

7.6.4.3. Step 3: Test Iteration

Determine the maximum and average achievable connections per second within the validation criteria.

7.7. HTTPS Throughput

7.7.1. Objective

Determine the throughput for HTTPS transactions varying the HTTPS response object size.

Test iterations MUST include common cipher suites and key strengths as well as forward looking stronger keys. Specific test iterations MUST include the ciphers and keys defined in the parameter Section 7.7.3.2.

7.7.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. must be documented.

7.7.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.7.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.7.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

Target Throughput: Initial value from product data sheet (if known)

Initial Throughput: 10% of "Target Throughput" (an optional parameter for documentation)

Number of HTTPS response object requests (transactions) per connection: 10

RECOMMENDED ciphers and keys:

1. ECHDE-ECDSA-AES128-GCM-SHA256 with Prime256v1 (Signature Hash Algorithm: ecdsa_secp256r1_sha256 and Supported group: sepc256r1)

2. ECDHE-RSA-AES128-GCM-SHA256 with RSA 2048 (Signature Hash Algorithm: rsa_pkcs1_sha256 and Supported group: sepc256)

3. ECDHE-ECDSA-AES256-GCM-SHA384 with Secp521 (Signature Hash Algorithm: ecdsa_secp384r1_sha384 and Supported group: sepc521r1)

4. ECDHE-RSA-AES256-GCM-SHA384 with RSA 4096 (Signature Hash Algorithm: rsa_pkcs1_sha384 and Supported group: secp256)

RECOMMENDED HTTPS response object size: 1 KByte, 2 KByte, 4 KByte, 16 KByte, 64 KByte, 256 KByte and mixed object defined in the table below.

Object size (KByte)	Number of requests/ Weight
0.2	1
6	1
8	1
9	1
10	1
25	1
26	1
35	1
59	1
347	1

Table 4: Mixed Objects

7.7.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile.

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of attempt transactions.

b. Traffic should be forwarded constantly.

c. Concurrent TCP connections MUST be constant during steady state and any deviation of concurrent TCP connections SHOULD be less than 10%. This confirms the DUT opens and closes TCP connections almost at the same rate

7.7.3.4. Measurement

The KPI metrics MUST be reported for this test scenario:

average Throughput and average HTTPS Transactions Per Second

7.7.4. Test Procedures and Expected Results

The test procedure consists of three major steps. This test procedure MAY be repeated multiple times with different IPv4 and IPv6 traffic distribution and HTTPS response object sizes.

7.7.4.1. Step 1: Test Initialization and Qualification

Verify the link status of the all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure traffic load profile of the test equipment to establish "initial throughput" as defined in the parameters Section 7.7.3.2.

The traffic load profile should be defined as described in Section 4.3.4. The DUT/SUT SHOULD reach the "Initial Throughput" during the sustain phase. Measure all KPI as defined in Section 7.7.3.4.

The measured KPIs during the sustain phase MUST meet the validation criteria "a" defined in Section 7.7.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.7.4.2. Step 2: Test Run with Target Objective

The test equipment SHOULD start to measure and record all specified KPIs. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired "Target Throughput" at the sustain phase. In addition, the measured KPIs MUST meet all validation criteria.

Perform the test separately for each HTTPS response object size.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.7.4.3. Step 3: Test Iteration

Determine the maximum and average achievable throughput within the validation criteria. Final test iteration MUST be performed for the test duration defined in Section 4.3.4.

7.8. HTTPS Transaction Latency

7.8.1. Objective

Using HTTPS traffic, determine the average HTTPS transaction latency when DUT is running with sustainable HTTPS transactions per second supported by the DUT/SUT under different HTTPS response object size.

Scenario 1: The client MUST negotiate HTTPS and close the connection with FIN immediately after completion of a single transaction (GET and RESPONSE).

Scenario 2: The client MUST negotiate HTTPS and close the connection with FIN immediately after completion of 10 transactions (GET and RESPONSE) within a single TCP connection.

7.8.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. MUST be documented.

7.8.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.8.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.8.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

RECOMMENDED cipher suites and key size: ECDHE-ECDSA-AES256-GCM-SHA384 with Secp521 bits key size (Signature Hash Algorithm: ecdsa_secp384r1_sha384 and Supported group: sepc521r1)

Target objective for scenario 1: 50% of the maximum connections per second measured in test scenario TCP/HTTPS Connections per second (Section 7.6)

Target objective for scenario 2: 50% of the maximum throughput measured in test scenario HTTPS Throughput (Section 7.7)

Initial objective for scenario 1: 10% of Target objective for scenario 1" (an optional parameter for documentation)

Initial objective for scenario 2: 10% of "Target objective for scenario 2" (an optional parameter for documentation)

HTTPS transaction per TCP connection: test scenario 1 with single transaction and the second scenario with 10 transactions

HTTPS 1.1 with GET command requesting a single 1, 16 or 64 KByte object. For each test iteration, client MUST request a single HTTPS response object size.

7.8.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile. Ramp up and ramp down phase SHOULD NOT be considered.

Generic criteria:

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of attempt transactions.

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. During the sustain phase, traffic should be forwarded at a constant rate.

d. Concurrent TCP connections MUST be constant during steady state and any deviation of concurrent TCP connections SHOULD be less than 10%. This confirms the DUT opens and closes TCP connections almost at the same rate

e. After ramp up the DUT MUST achieve the "Target objective" defined in the parameter Section 7.8.3.2 and remain in that state for the entire test duration (sustain phase).

7.8.3.4. Measurement

Following KPI metrics MUST be reported for each test scenario and HTTPS response object sizes separately:

TTFB (minimum, average and maximum) and TTLB (minimum, average and maximum)

All KPI’s are measured once the target connections per second achieves the steady state.

7.8.4. Test Procedures and Expected Results

The test procedure is designed to measure average TTFB or TTLB when the DUT is operating close to 50% of its maximum achievable connections per second. This test procedure can be repeated multiple times with different IP types (IPv4 only, IPv6 only and IPv4 and IPv6 mixed traffic distribution), HTTPS response object sizes and single and multiple transactions per connection scenarios.

7.8.4.1. Step 1: Test Initialization and Qualification

Verify the link status of the all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure traffic load profile of the test equipment to establish "Initial objective" as defined in the parameters Section 7.8.3.2. The traffic load profile can be defined as described in Section 4.3.4.

The DUT/SUT SHOULD reach the "Initial objective" before the sustain phase. The measured KPIs during the sustain phase MUST meet the validation criteria a, b, c, d, e and f defined in Section 7.8.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.8.4.2. Step 2: Test Run with Target Objective

Configure test equipment to establish "Target objective" defined in the parameters table. The test equipment SHOULD follow the traffic load profile definition as described in Section 4.3.4.

During the ramp up and sustain phase, other KPIs such as throughput, concurrent TCP connections and application transactions per second MUST NOT reach to the maximum value that the DUT/SUT can support. The test results for specific test iterations SHOULD NOT be reported, if the above mentioned KPI (especially throughput) reaches to the maximum value. (Example: If the test iteration with 64 KByte of HTTP response object size reached the maximum throughput limitation of the DUT, the test iteration MAY be interrupted and the result for 64 KByte SHOULD NOT be reported).

The test equipment SHOULD start to measure and record all specified KPIs. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed. DUT/SUT is expected to reach the desired "Target objective" at the sustain phase. In addition, the measured KPIs MUST meet all validation criteria.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.8.4.3. Step 3: Test Iteration

Determine the maximum achievable connections per second within the validation criteria and measure the latency values.

7.9. Concurrent TCP/HTTPS Connection Capacity

7.9.1. Objective

Determine the maximum number of concurrent TCP connections that the DUT/SUT sustains when using HTTPS traffic.

7.9.2. Test Setup

Test bed setup SHOULD be configured as defined in Section 4. Any specific test bed configuration changes such as number of interfaces and interface type, etc. MUST be documented.

7.9.3. Test Parameters

In this section, test scenario specific parameters SHOULD be defined.

7.9.3.1. DUT/SUT Configuration Parameters

DUT/SUT parameters MUST conform to the requirements defined in Section 4.2. Any configuration changes for this specific test scenario MUST be documented.

7.9.3.2. Test Equipment Configuration Parameters

Test equipment configuration parameters MUST conform to the requirements defined in Section 4.3. Following parameters MUST be documented for this test scenario:

Client IP address range defined in Section 4.3.1.2

Server IP address range defined in Section 4.3.2.2

Traffic distribution ratio between IPv4 and IPv6 defined in Section 4.3.1.2

RECOMMENDED cipher suites and key size: ECDHE-ECDSA-AES256-GCM-SHA384 with Secp521 bits key size (Signature Hash Algorithm: ecdsa_secp384r1_sha384 and Supported group: sepc521r1)

Target concurrent connections: Initial value from product data sheet (if known)

Initial concurrent connections: 10% of "Target concurrent connections" (an optional parameter for documentation)

Connections per second during ramp up phase: 50% of maximum connections per second measured in test scenario TCP/HTTPS Connections per second (Section 7.6)

Ramp up time (in traffic load profile for "Target concurrent connections"): "Target concurrent connections" / "Maximum connections per second during ramp up phase"

Ramp up time (in traffic load profile for "Initial concurrent connections"): "Initial concurrent connections" / "Maximum connections per second during ramp up phase"

The client MUST perform HTTPS transaction with persistence and each client can open multiple concurrent TCP connections per server endpoint IP.

Each client sends 10 GET commands requesting 1 KByte HTTPS response objects in the same TCP connections (10 transactions/TCP connection) and the delay (think time) between each transactions MUST be X seconds.

X = ("Ramp up time" + "steady state time") /10

The established connections SHOULD remain open until the ramp down phase of the test. During the ramp down phase, all connections SHOULD be successfully closed with FIN.

7.9.3.3. Test Results Validation Criteria

The following test Criteria is defined as test results validation criteria. Test results validation criteria MUST be monitored during the whole sustain phase of the traffic load profile.

a. Number of failed Application transactions (receiving any HTTP response code other than 200 OK) MUST be less than 0.001% (1 out of 100,000 transactions) of total attempted transactions

b. Number of Terminated TCP connections due to unexpected TCP RST sent by DUT/SUT MUST be less than 0.001% (1 out of 100,000 connections) of total initiated TCP connections

c. During the sustain phase, traffic SHOULD be forwarded constantly

7.9.3.4. Measurement

Following KPI metric MUST be reported for this test scenario:

average Concurrent TCP Connections

7.9.4. Test Procedures and expected Results

The test procedure is designed to measure the concurrent TCP connection capacity of the DUT/SUT at the sustaining period of traffic load profile. The test procedure consists of three major steps. This test procedure MAY be repeated multiple times with different IPv4 and IPv6 traffic distribution.

7.9.4.1. Step 1: Test Initialization and Qualification

Verify the link status of all connected physical interfaces. All interfaces are expected to be in "UP" status.

Configure test equipment to establish "initial concurrent TCP connections" defined in Section 7.9.3.2. Except ramp up time, the traffic load profile SHOULD be defined as described in Section 4.3.4.

During the sustain phase, the DUT/SUT SHOULD reach the "Initial concurrent TCP connections". The measured KPIs during the sustain phase MUST meet the validation criteria "a" and "b" defined in Section 7.9.3.3.

If the KPI metrics do not meet the validation criteria, the test procedure MUST NOT be continued to "Step 2".

7.9.4.2. Step 2: Test Run with Target Objective

Configure test equipment to establish "Target concurrent TCP connections". The test equipment SHOULD follow the traffic load profile definition (except ramp up time) as described in Section 4.3.4.

During the ramp up and sustain phase, the other KPIs such as throughput, TCP connections per second and application transactions per second MUST NOT reach to the maximum value that the DUT/SUT can support.

The test equipment SHOULD start to measure and record KPIs defined in Section 7.9.3.4. The frequency of measurement SHOULD be 2 seconds. Continue the test until all traffic profile phases are completed.

The DUT/SUT is expected to reach the desired target concurrent connections at the sustain phase. In addition, the measured KPIs MUST meet all validation criteria.

Follow step 3, if the KPI metrics do not meet the validation criteria.

7.9.4.3. Step 3: Test Iteration

Determine the maximum and average achievable concurrent TCP connections within the validation criteria.

8. Formal Syntax

9. IANA Considerations

This document makes no request of IANA.

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

10. Security Considerations

The primary goal of this document is to provide benchmarking terminology and methodology for next-generation network security devices. However, readers should be aware that there is some overlap between performance and security issues. Specifically, the optimal configuration for network security device performance may not be the most secure, and vice-versa. The Cipher suites recommended in this document are just for test purpose only. The Cipher suite recommendation for a real deployment is outside the scope of this document.

11. Acknowledgements

Acknowledgements will be added in the future release.

12. Contributors

The authors would like to thank the many people that contributed their time and knowledge to this effort.

Specifically, to the co-chairs of the NetSecOPEN Test Methodology working group and the NetSecOPEN Security Effectiveness working group - Alex Samonte, Aria Eslambolchizadeh, Carsten Rossenhoevel and David DeSanto.

Additionally, the following people provided input, comments and spent time reviewing the myriad of drafts. If we have missed anyone the fault is entirely our own. Thanks to - Amritam Putatunda, Chao Guo, Chris Chapman, Chris Pearson, Chuck McAuley, David White, Jurrie Van Den Breekel, Michelle Rhines, Rob Andrews, Samaresh Nair, and Tim Winters.

13. References

13.1. Normative References

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

13.2. Informative References

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, DOI 10.17487/RFC2616, June 1999, <https://www.rfc-editor.org/info/rfc2616>.

[RFC2647] Newman, D., "Benchmarking Terminology for Firewall Performance", RFC 2647, DOI 10.17487/RFC2647, August 1999, <https://www.rfc-editor.org/info/rfc2647>.

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

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, DOI 10.17487/RFC5681, September 2009, <https://www.rfc-editor.org/info/rfc5681>.

Appendix A. NetSecOPEN Basic Traffic Mix

A traffic mix for testing performance of next generation firewalls MUST scale to stress the DUT based on real-world conditions. In order to achieve this the following MUST be included:

- o Clients connecting to multiple different server FQDNs per application
- o Clients loading apps and pages with connections and objects in specific orders
- o Multiple unique certificates for HTTPS/TLS
- o A wide variety of different object sizes
- o Different URL paths
- o Mix of HTTP and HTTPS

A traffic mix for testing performance of next generation firewalls MUST also facilitate application identification using different detection methods with and without decryption of the traffic. Such as:

- o HTTP HOST based application detection

- o HTTPS/TLS Server Name Indication (SNI)
- o Certificate Subject Common Name (CN)

The mix MUST be of sufficient complexity and volume to render differences in individual apps as statistically insignificant. For example, changes in like to like apps - such as one type of video service vs. another both consist of larger objects whereas one news site vs. another both typically have more connections then other apps because of trackers and embedded advertising content. To achieve sufficient complexity, a mix MUST have:

- o Thousands of URLs each client walks thru
- o Hundreds of FQDNs each client connects to
- o Hundreds of unique certificates for HTTPS/TLS
- o Thousands of different object sizes per client in orders matching applications

The following is a description of what a popular application in an enterprise traffic mix contains.

Table 5 lists the FQDNs, number of transactions and bytes transferred as an example, client interactions with Office 365 Outlook, Word, Excel, PowerPoint, SharePoint and Skype.

Office365 FQDN	Bytes	Transaction
r1.res.office365.com	14,056,960	192
s1-word-edit-15.cdn.office.net	6,731,019	22
company1-my.sharepoint.com	6,269,492	42
swx.cdn.skype.com	6,100,027	12
static.sharepointonline.com	6,036,947	41
spoprod-a.akamaihd.net	3,904,250	25
s1-excel-15.cdn.office.net	2,767,941	16
outlook.office365.com	2,047,301	86
shellprod.msocdn.com	1,008,370	11

word-edit.officeapps.live.com	932,080	25
res.delve.office.com	760,146	2
s1-powerpoint-15.cdn.office.net	557,604	3
appsforoffice.microsoft.com	511,171	5
powerpoint.officeapps.live.com	471,625	14
excel.officeapps.live.com	342,040	14
s1-officeapps-15.cdn.office.net	331,343	5
webdir0a.online.lync.com	66,930	15
portal.office.com	13,956	1
config.edge.skype.com	6,911	2
clientlog.portal.office.com	6,608	8
webdir.online.lync.com	4,343	5
graph.microsoft.com	2,289	2
nam.loki.delve.office.com	1,812	5
login.microsoftonline.com	464	2
login.windows.net	232	1

Table 5: Office365

Clients MUST connect to multiple server FQDNs in the same order as real applications. Connections MUST be made when the client is interacting with the application and MUST NOT first setup up all connections. Connections SHOULD stay open per client for subsequent transactions to the same FQDN similar to how a web browser behaves. Clients MUST use different URL Paths and Object sizes in orders as they are observed in real Applications. Clients MAY also setup multiple connections per FQDN to process multiple transactions in a sequence at the same time. Table 6 has a partial example sequence of the Office 365 Word application transactions.

FQDN	URL Path	Object size
company1-my.sharepoint.com	/personal...	23,132
word-edit.officeapps.live.com	/we/WsaUpload.ashx	2
static.sharepointonline.com	/bld/.../blank.js	454
static.sharepointonline.com	/bld/.../ initstrings.js	23,254
static.sharepointonline.com	/bld/.../init.js	292,740
company1-my.sharepoint.com	/ScriptResource...	102,774
company1-my.sharepoint.com	/ScriptResource...	40,329
company1-my.sharepoint.com	/WebResource...	23,063
word-edit.officeapps.live.com	/we/wordeditorframe. aspx...	60,657
static.sharepointonline.com	/bld/_layouts/.../ blank.js	454
s1-word-edit-15.cdn.office.net	/we/s/.../ EditSurface.css	19,201
s1-word-edit-15.cdn.office.net	/we/s/.../ WordEditor.css	221,397
s1-officeapps-15.cdn.office.net	/we/s/.../ Microsoft Ajax.js	107,571
s1-word-edit-15.cdn.office.net	/we/s/.../ wacbootwe.js	39,981
s1-officeapps-15.cdn.office.net	/we/s/.../ CommonIntl.js	51,749
s1-word-edit-15.cdn.office.net	/we/s/.../ Compat.js	6,050
s1-word-edit-15.cdn.office.net	/we/s/.../ Box4Intl.js	54,158

s1-word-edit-15.cdn.office.net	/we/s/.../ WoncaIntl.js	24,946
s1-word-edit-15.cdn.office.net	/we/s/.../ WordEditorIntl.js	53,515
s1-word-edit-15.cdn.office.net	/we/s/.../ WordEditorExp.js	1,978,712
s1-word-edit-15.cdn.office.net	/we/s/.../jSanity.js	10,912
word-edit.officeapps.live.com	/we/OneNote.ashx	145,708

Table 6: Office365 Word Transactions

For application identification the HTTPS/TLS traffic MUST include realistic Certificate Subject Common Name (CN) data as well as Server Name Indications (SNI). For example, a DUT MAY detect Facebook Chat traffic by inspecting the certificate and detecting *.facebook.com in the certificate subject CN and subsequently detect the word chat in the FQDN 5-edge-chat.facebook.com and identify traffic on the connection to be Facebook Chat.

Table 7 includes further examples in SNI and CN pairs for several FQDNs of Office 365.

Server Name Indication (SNI)	Certificate Subject Common Name (CN)
r1.res.office365.com	*.res.outlook.com
login.windows.net	graph.windows.net
webdir0a.online.lync.com	*.online.lync.com
login.microsoftonline.com	stamp2.login.microsoftonline.com
webdir.online.lync.com	*.online.lync.com
graph.microsoft.com	graph.microsoft.com
outlook.office365.com	outlook.com
appsforoffice.microsoft.com	appsforoffice.microsoft.com

Table 7: Office365 SNI and CN Pairs Examples

NetSecOPEN has provided a reference enterprise perimeter traffic mix with dozens of applications, hundreds of connections, and thousands of transactions.

The enterprise perimeter traffic mix consists of 70% HTTPS and 30% HTTP by Bytes, 58% HTTPS and 42% HTTP by Transactions. By connections with a single connection per FQDN the mix consists of 43% HTTPS and 57% HTTP. With multiple connections per FQDN the HTTPS percentage is higher.

Table 8 is a summary of the NetSecOPEN enterprise perimeter traffic mix sorted by bytes with unique FQDNs and transactions per applications.

Application	FQDNs	Transactions	Bytes
Office365	26	558	52,931,947
Box	4	90	23,276,089
Salesforce	6	365	23,137,548
Gmail	13	139	16,399,289

Linkedin	10	206	15,040,918
DailyMotion	8	77	14,751,514
GoogleDocs	2	71	14,205,476
Wikia	15	159	13,909,777
Foxnews	82	499	13,758,899
Yahoo Finance	33	254	13,134,011
Youtube	8	97	13,056,216
Facebook	4	207	12,726,231
CNBC	77	275	11,939,566
Lightreading	27	304	11,200,864
BusinessInsider	16	142	11,001,575
Alexa	5	153	10,475,151
CNN	41	206	10,423,740
Twitter Video	2	72	10,112,820
Cisco Webex	1	213	9,988,417
Slack	3	40	9,938,686
Google Maps	5	191	8,771,873
SpectrumIEEE	7	145	8,682,629
Yelp	9	146	8,607,645
Vimeo	12	74	8,555,960
Wikihow	11	140	8,042,314
Netflix	3	31	7,839,256
Instagram	3	114	7,230,883
Morningstar	30	150	7,220,121

Docusign	5	68	6,972,738
Twitter	1	100	6,939,150
Tumblr	11	70	6,877,200
Whatsapp	3	46	6,829,848
Imdb	16	251	6,505,227
NOAAgov	1	44	6,316,283
IndustryWeek	23	192	6,242,403
Spotify	18	119	6,231,013
AutoNews	16	165	6,115,354
Evernote	3	47	6,063,168
NatGeo	34	104	6,026,344
BBC News	18	156	5,898,572
Investopedia	38	241	5,792,038
Pinterest	8	102	5,658,994
Succesfactors	2	112	5,049,001
AbaJournal	6	93	4,985,626
Pbworks	4	78	4,670,980
NetworkWorld	42	153	4,651,354
WebMD	24	280	4,416,736
OilGasJournal	14	105	4,095,255
Trello	5	39	4,080,182
BusinessWire	5	109	4,055,331
Dropbox	5	17	4,023,469
Nejm	20	190	4,003,657

OilGasDaily	7	199	3,970,498
Chase	6	52	3,719,232
MedicalNews	6	117	3,634,187
Marketwatch	25	142	3,291,226
Imgur	5	48	3,189,919
NPR	9	83	3,184,303
Onelogin	2	31	3,132,707
Concur	2	50	3,066,326
Service-now	1	37	2,985,329
Apple itunes	14	80	2,843,744
BerkeleyEdu	3	69	2,622,009
MSN	39	203	2,532,972
Indeed	3	47	2,325,197
MayoClinic	6	56	2,269,085
Ebay	9	164	2,219,223
UCLAedu	3	42	1,991,311
ConstructionDive	5	125	1,828,428
EducationNews	4	78	1,605,427
BofA	12	68	1,584,851
ScienceDirect	7	26	1,463,951
Reddit	8	55	1,441,909
FoodBusinessNews	5	49	1,378,298
Amex	8	42	1,270,696
Weather	4	50	1,243,826

Wikipedia	3	27	958,935
Bing	1	52	697,514
ADP	1	30	508,654
Grand Total	983	10021	569,819,095

Table 8: Summary of NetSecOPEN Enterprise Perimeter Traffic Mix

Authors' Addresses

Balamuhunthan Balarajah

Email: bm.balarajah@gmail.com

Carsten Rossenhoevel
EANTC AG
Salzufer 14
Berlin 10587
Germany

Email: cross@eantc.de

Brian Monkman
NetSecOPEN
417 Independence Court
Mechanicsburg, PA 17050
USA

Email: bmonkman@netsecopen.org

Internet Engineering Task Force S. Jacob, Ed.
Internet-Draft K. Tiruveedhula
Intended status: Informational Juniper Networks
Expires: November 7, 2020 May 6, 2020

Benchmarking Methodology for EVPN VPWS
draft-kishjac-bmwg-evpnvpwstest-04

Abstract

This document defines methodologies for benchmarking EVPN-VPWS performance. EVPN-VPWS is defined in RFC 8214, and is being deployed in Service Provider networks. Specifically this document defines the methodologies for benchmarking EVPN-VPWS Scale convergence, Fail over,Core isolation,high availability and longevity.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

EVPN-VPWS is defined in RFC 8214,discusses how VPWS can be combined with EVPNs to provide a new/combined solution. This draft defines methodologies that can be used to benchmark RFC 8214 solutions. Further, this draft provides methodologies for benchmarking the performance of EVPN VPWS Scale,Scale Convergence, Core isolation, longevity, high availability.

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

All-Active Redundancy Mode: When all PEs attached to an Ethernet segment are allowed to forward known unicast traffic to/from that

Ethernet segment for a given VLAN, then the Ethernet segment is defined to be operating in All-Active redundancy mode.

AA: All Active mode

AC: Attachment Circuits

CE: Customer Router/Devices/Switch.

DF: Designated Forwarder

DUT: Device under test.

Ethernet Segment (ES): When a customer site (device or network) is connected to one or more PEs via a set of Ethernet links, then that set of links is referred to as an 'Ethernet segment'.

EVI: An EVPN instance spanning the Provider Edge (PE) devices participating in that EVPN.

Ethernet Segment Identifier (ESI): A unique non-zero identifier that identifies an Ethernet segment is called an 'Ethernet Segment Identifier'.

Ethernet Tag: An Ethernet tag identifies a particular broadcast domain, e.g., a VLAN. An EVPN instance consists of one or more broadcast domains.

Interface: Physical interface of a router/switch.

IRB: Integrated routing and bridging interface

MAC: Media Access Control addresses on a PE.

MHPE2: Multi homed Provider Edge router 2.

MHPE1: Multi homed Provider Edge router 1.

SHPE3: Single homed Provider Edge Router 3.

PE: Provider Edge device.

P: Provider Router.

RR: Route Reflector.

RT: Traffic Generator.

Sub Interface: Each physical Interfaces is subdivided into Logical units.

SA: Single Active

Single-Active Redundancy Mode: When only a single PE, among all the PEs attached to an Ethernet segment, is allowed to forward traffic to/from that Ethernet segment for a given VLAN, then the Ethernet segment is defined to be operating in Single-Active redundancy mode.

VPWS: Virtual private wire service.

2. Test Topology

There are five routers in the Test setup. SHPE3, RR/P, MHPE1 and MHPE2 emulating a service provider network. CE is a customer device connected to MHPE1 and MHPE2, it is configured with bridge domains in multiple vlans. The traffic generator is connected to CE and SHPE3.The MHPE1 acts as DUT.The traffic generator will be used as sender and receiver of traffic.The DUT will be the reference point for all the test cases. MHPE1 and MHPE2 are multi home routers connected to CE running single active mode. The traffic generator will be generating traffic at 10% of the line rate.

Topology Diagram

```
             +-----------------+                  +--------------------------+
             |                 |                  |                          |
             |                 |                  |Traffic Generator sender/ |
             |  SHPE3          |                  |receiver of layer 2 traffic|
             |                 +------------------+  with multiple Vlans     |
             |                 |                  +--------------------------+
             +---------+------+
                       |  Core Link
                       |
              +--------+-----+
              |              |
              |   RR/P       |
              |              |     Core link
              |              +-----------------+
              +--+-----------+                 |
                 |                             |
                 |    core link                |
                 |                             |
   +-------------+---+                        ++-------------------+
   |                 |                        |                    |
   |                 |                        |   MHPE2            |
   |MHPE1(DUT)       |                        |                    |
   |                 |                        |                    |
   |                 |                        |                    |
   +-----------------+------+           +-----+--------------------+
                            |           |
           PE-CE link       |           | PE-CE link
                            |           |
                            |           |
                            |           |
                            |           |
                      +-----+-----------+----+         +-----------------------
----+
                      | CE/Layer 2 bridge    +-----------| Traffic Generator  send
er/ |
                      |                      |         |receiver  of layer 2 tra
ffic|
                      |                      |         | with multiple Vlans
    |
                      |                      |         +-----------------------
----+
                      |                      |
                      +----------------------+
```

Topology 1

Topology Diagram

Figure 1

Test Setup Configurations:

SHPE3 is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support. This router must be configured with N EVPN-VPWS instances for testing. Traffic generator is connected to this router for sending and receiving traffic.

RR is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support. This router acts as a provider router and as a route reflector.

MHPE1 is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support.This router must be configured with N EVPN-VPWS instances for testing.This router is configured with ESI per vlan or ESI per interface. It is functioning as multi homing PE working on Single Active EVPN mode.This router serves as the DUT and it is connected to CE. MHPE1 is acting as DUT for all the test cases.

MHPE2 is configured with Interior Gateway protocols like OPSF or IS-IS for underlay, LDP for MPLS support,Interior Border Gateway with EVPN address family for overlay support.This router must be configured with N EVPN-VPWS instances for testing.This router is configured with ESI per vlan or ESI per interface. It is functioning as multi homing PE working on Single Active EVPN mode.It is connected to CE.

CE is acting as bridge configured with multiple vlans,the same vlans are configured on MHPE1,MHPE2,SHPE3. traffic generator is connected to CE. The traffic generator acts as sender or receiver of traffic.

Depending up on the test scenarios the traffic generators will be used to generate uni directional or bi directional flows.

The above configuration will be serving as the base configuration for all test cases.

3. Test Cases

The following tests are conducted to measure the packet loss during the local link and core failure in DUT with Scaled AC's.

3.1. Local Failure Scenario 1

Objective:

Measure the time taken to switch from primary to backup during local link failure.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VPWS. The AC must be up and running. "N" AC's in MHPE1,MHPE2, working in SA mode.Ensure DUT is active and MHPE2 is backup PE.Send unicast packets to CE from traffic generator. The traffic is uni directional and it flows from CE to DUT working as Active router. Then shut the DUT-CE link, so that traffic from CE switches to MHPE2.Traffic must be tested with various line rate that from 10% to 98%.

Measurement :

Measure the time taken by the traffic to switch from Active router to the backup. The test is repeated for "N" times and the values are collected. The AC's local switch over time is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts. Fail over time must be measured for various line rate.

AC's switch over from primary to backup PE in sec = (T1+T2+..Tn/N)

3.2. Local Failure Scenario 2

Objective:

Measure time taken by remote PE to switch traffic from primary to backup during CE link failure.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VPWS. The AC must be up and running. "N" AC's in MHPE1,MHPE2, working in SA mode.Ensure DUT is active and MHPE2 is backup PE.Send unicast packets to SHPE3 from traffic generator. The traffic is uni directional and it flows from

SHPE3 to DUT working as Active router. Then shut the DUT-CE link, so the remote traffic flow switches from DUT to MHPE2.Traffic must be tested with various line rate that from 10% to 98%.

Measurement :

Measure the time taken by the traffic to switch from Active router to the backup.The test is repeated for "N" times and the values are collected. The AC's switch over time for the remote traffic is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts. Fail over time must be measured for various line rate.

AC's switch over from primary to backup PE in sec = (T1+T2+..Tn/N)

3.3. Core Failure

Objective:

Measure the time taken by remote PE to switch traffic from primary to backup during core link failure.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VPWS. The AC must be up and running. "N" AC's in MHPE1,MHPE2, working in SA mode.Ensure DUT is active and MHPE2 is backup PE.Send unicast packets to SHPE3 from traffic generator. The traffic is uni directional and it flows from SHPE3 to DUT working as Active router. Then shut the DUT core link, so the remote traffic flow switches from DUT to MHPE2. Traffic must be tested with various line rate that from 10% to 98%.

Measurement :

Measure the time taken by the traffic to switch from Active router to the backup. The test is repeated for "N" times and the values are collected. The AC's switch over time for the remote traffic is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts. Fail over time must be measured for various line rate.

AC's core Failure fail over time = (T1+T2+..Tn/N)

3.4. Link Flap

Objective:

Measure time taken by primary PE to regain control after the local PE-CE link flap.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VPWS. The AC must be up and running. "N" AC's in MHPE1,MHPE2, working in SA mode.Ensure DUT is active and MHPE2 is backup PE.Send unicast packets to CE from traffic generator. The traffic is uni directional and it flows from CE to DUT working as Active router. Then shut the DUT core link, so the local traffic flow switches from DUT to MHPE2. Once the fail over is performed. Bring the link up. Now the DUT becomes the Active router. Measure time taken by the DUT to regain the traffic.Traffic must be tested with various line rate that from 10% to 98%.

Measurement :

Measure the time taken by the traffic to switch back to the DUT. The test is repeated for "N" times and the values are collected. The AC's switch over time for the remote traffic is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts. Fail over time must be measured for various line rate.

Time taken to switch back to primary(DUT) once the link is restored = (T1+T2+..Tn/N)

4. Scale Convergence

4.1. To measure the packet loss during the core link failure.

Objective:

Measure the convergence at a higher number of AC's

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VPWS. The AC must be up and running. "N*100" AC's in MHPE1,MHPE2, working in SA mode.Ensure DUT is active and MHPE2 is backup PE.Send unicast packets to CE from traffic generator and send traffic from traffic generator to SHPE3.The traffic is directional and it flows from CE to DUT and from DUT to CE, working as Active router. Then shut the DUT core link, so the traffic flow switches from DUT to MHPE2. Measure traffic switching time.Traffic must be tested with various line rate that from 10% to 98%.

Measurement :

Measure the time taken by the traffic to switch from DUT to MHPE2. The test is repeated for "N" times and the values are collected. The AC's switch over time for the traffic is calculated by averaging the values obtained by "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts. Fail over time must be measured for various line rate.

Packet loss in sec = (T1+T2+..Tn/N)

5. High Availability

5.1. To Record the whether there is traffic loss due to routing engine failover for redundancy test.

Objective:

Measure the traffic loss during routing engine fail over.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VPWS. The AC must be up and running. "N*100" AC's in MHPE1,MHPE2, working in SA mode.Ensure DUT is active and MHPE2 is backup PE.Send unicast packets to CE and SHPE3 from traffic generator. The traffic is directional and it flows from CE to DUT and from DUT to CE, working as Active router. Do a routing engine fail over once the traffic is stabilized in DUT. Traffic must be tested with various line rate that from 10% to 98%. The expectation is 0 packet loss, no role change in AC's.

Measurement :

The expectation of the test is 0 traffic loss with no change in the DF role. DUT should not withdraw any routes.But in cases where the DUT is not property synchronized between master and standby,due to that packet loss are observed. In that scenario the packet loss is measured.The test is repeated for "N" times and the values are collected.The packet loss is calculated by averaging the values obtained by "N" samples.

Packet loss in sec = (T1+T2+..Tn/N)

6. SOAK Test

This test is carried out to measure the stability of the DUT in a scaled environment with traffic over a period of time "T'". In each interval "t1" the DUT CPU usage, memory usage are measured. The DUT is checked for any crashes during this time period.

6.1. To Measure the stability of the DUT with scale and traffic.

Objective:

To measure the stability of the DUT in a scaled environment with traffic.

Topology : Topology 1

Procedure:

Scale N AC's in DUT,SHPE3 and MHPE2.Send F frames to DUT from CE using traffic generator with different X SA and DA for N EVI's. Send F frames from traffic generator to SHPE3 with X different SA and DA. There is a bi directional traffic flow with F pps in each direction. The DUT must run with traffic for 24 hours, every hour check for memory leak, crash.

Measurement :

Take the hourly reading of CPU, process memory.There should not be any leak, crashes, CPU spikes. Th CPU spike is determined as the CPU usage which shoots at 40 to 50 percent of the average usage. The average value vary from device to device. Memory leak is determined by increase usage of the memory for EVPN-VPWS process. The expectation is under steady state the memory usage for EVPN-VPWS process should not increase.

7. Acknowledgements

We would like to thank Al and Sarah for the support.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

The benchmarking tests described in this document are limited to the performance characterization of controllers in a lab environment with isolated networks. 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.

10. References

10.1. Normative References

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

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

[RFC2899] Ginoza, S., "Request for Comments Summary RFC Numbers 2800-2899", RFC 2899, DOI 10.17487/RFC2899, May 2001, <https://www.rfc-editor.org/info/rfc2899>.

10.2. Informative References

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RFC8214] Boutros, S., Sajassi, A., Salam, S., Drake, J., and J. Rabadan, "Virtual Private Wire Service Support in Ethernet VPN", RFC 8214, DOI 10.17487/RFC8214, August 2017, <https://www.rfc-editor.org/info/rfc8214>.

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

Benchmarking Methodology Working Group
Internet-Draft
Intended status: Informational
Expires: November 21, 2020
G. Lencse
BUTE
K. Shima
IIJ-II
May 20, 2020

An Upgrade to Benchmarking Methodology for Network Interconnect Devices

draft-lencse-bmwg-rfc2544-bis-00

Abstract

RFC 2544 has defined a benchmarking methodology for network interconnect devices. We recommend a few upgrades to it for producing more reasonable results. The recommended upgrades can be classified into two categories: the application of the novelties of RFC 8219 for the legacy RFC 2544 use cases and the following new ones. Checking a reasonably small timeout individually for every single frame in the throughput and frame loss rate benchmarking procedures. Performing a statistically relevant number of tests for all benchmarking procedures. Addition of an optional non-zero frame loss acceptance criterion for the throughput measurement procedure and defining its reporting format.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

[RFC2544] has defined a benchmarking methodology for network interconnect devices. [RFC5180] addressed IPv6 specificities and also added technology updates, but declared IPv6 transition technologies out of its scope. [RFC8219] addressed the IPv6 transition technologies, and it added further measurement procedures (e.g. for packet delay variation (PDV) and inter packet delay variation (IPDV)). It has also recommended to perform multiple tests (at least 20), and it proposed median as summarizing function and 1st and 99th percentiles as the measure of variation of the results of the multiple tests. This is a significant change compared to [RFC2544], which always used only average as summarizing function. [RFC8219] also redefined the latency measurement procedure with the requirement of marking at least 500 frames with identifying tags for latency measurements, instead of using only a single one. However, all these improvements apply only for the IPv6 transition technologies, and no update was made to [RFC2544] / [RFC5180], which we believe to be desirable.

Moreover, [RFC8219] has reused the throughput and frame loss rate benchmarking procedures from [RFC2544] with no changes. When we tested their feasibility with a few SIIT [RFC7915] implementations, we have pointed out three possible improvements in [LEN2020A]:

- o Checking a reasonably small timeout individually for every single frame with the throughput and frame loss rate benchmarking procedures.

- o Performing a statistically relevant number of tests for these two benchmarking procedures.

- o Addition of an optional non-zero frame loss acceptance criterion for the throughput benchmarking procedure and defining its reporting format.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Recommendation to Backport the Novelties of RFC8219

Besides addressing IPv6 transition technologies, [RFC8219] has also made several technological upgrades reflecting the current state of the art of networking technologies and benchmarking. But all the novelties mentioned in Section 1 of this document currently apply only for the benchmarking of IPv6 transition technologies. We contend that they could be simply backported to the benchmarking of network interconnect devices. For example, siitperf [SIITPERF], our [RFC8219] compliant DPDK-based software Tester was designed for benchmarking different SIIT [RFC7915] (also called stateless NAT64) implementations, but if it is configured to have the same IP version on both sides, it can be used to test IPv4 or IPv6 (or dual stack) routers [LEN2020B]. We highly recommend the backporting of the latency, PDV and IPDV benchmarking measurement procedures of [RFC8219].

3. Improved Throughput and Frame Loss Rate Measurement Procedures using Individual Frame Timeout

The throughput measurement procedure defined in [RFC2544] only counts the number of the sent and received test frames, but it does not identify the test frames individually. On the one hand, this approach allows the Tester to send always the very same test frame to the DUT, which was very likely an important advantage in 1999. However, on the other hand, thus the Tester cannot check if the order of the frames is kept, or if the frames arrive back to the Tester within a given timeout time. (Perhaps none of them was an issue of hardware based network interconnect devices in 1999. But today network packet forwarding and manipulation is often implemented in software having larger buffers and producing potentially higher latencies.)

Whereas real-time applications are obviously time sensitive, other applications like HTTP or FTP are often considered throughput hungry and time insensitive. However, we have demonstrated that when we applied 100ms delay to 1% of the test frames, the throughput of HTTP download dropped by more that 50% [LEN2020C]. Therefore, an advanced throughput measurement procedure that checks the timeout time for every single test frame may produce more reasonable results. We have shown that this measurement is now feasible [LEN2020B]. In this case, we used 64-bit integers to identify the test frames and measured the latency of the frames as required by the PDV measurement procedure in Section 7.3.1. of [RFC8219]. In our particular test, we used 10ms as frame timeout, which could be a suitable value, but we recommend further studies do determine the recommended timeout value.

We recommend that the reported results of the improved throughput and frame loss rate measurements SHOULD include the applied timeout value.

4. Requirement of Statistically Relevant Number of Tests

Section 4 of [RFC2544] says that: "Furthermore, selection of the tests to be run and evaluation of the test data must be done with an understanding of generally accepted testing practices regarding repeatability, variance and statistical significance of small numbers of trials." It is made a stronger requirement (by using a "MUST") in Section 3 of [RFC5180] stating that: "Test execution and results analysis MUST be performed while observing generally accepted testing practices regarding repeatability, variance, and statistical significance of small numbers of trials." But no practical guidelines are provided concerning the minimally necessary number of tests.

[RFC8219] mentions at four different places that the tests must be repeated at least 20 times. These places are the benchmarking procedures for:

- latency (Section 7.2)
- packet delay variation (Section 7.3.1)

- o inter packet delay variation (Section 7.3.2)
- o DNS64 performance (Section 9.2).

We believe that a similar guideline for the minimal number of tests would be helpful for the throughput and frame loss rate benchmarking procedures. We consider 20 as an affordable number of minimum repetitions of the frame loss rate measurements. However, as for throughput measurements, we contend that the binary search may require rather high number of steps in certain situations (e.g. tens of millions of frames per second rate and high resolution) that the requirement of at least 20 repetitions of the binary search would result in unreasonably high measurement execution times. Therefore, we recommend to use an algorithm that checks the statistical properties of the results of the tests and it may stop before 20 repetitions, if the results are consistent, but it may require more than 20 repetitions, if the results are scattered. (The algorithm is yet to be developed.)

5. An Optional Non-zero Frame Loss Acceptance Criterion for the Throughput Measurement Procedure

When we defined the measurement procedure for DNS64 performance in Section 9.2 of [RFC8219], we followed both spirit and wording of the [RFC2544] throughput measurement procedure including the requirement for absolutely zero packet loss. We have elaborated our underlying considerations in our research paper [LEN2017] as follows:

1. Our goal is a well-defined performance metric, which can be measured simply and efficiently. Allowing any packet loss would result in a need for scanning/trying a large range of rates to discover the highest rate of successfully processed DNS queries.

2. Even if users may tolerate a low loss rate (please note the DNS uses UDP with no guarantee for delivery), it cannot be arbitrarily high, thus, we could not avoid defining a limit. However, any other limits than zero percent would be hardly defensible.

3. Other benchmarking procedures use the same criteria of zero packet loss and this is the standard in IETF Benchmarking Methodology Working Group.

On the one hand, we still consider our arguments valid, however, on the other hand, we are aware of different arguments for the justification of an optional non-zero frame loss acceptance criterion, too:

- Frame loss is present in our networks from the very beginning and our applications are well prepared to handle frame loss. They can definitely tolerate some low frame loss rates like 0.01% (1 frame from 10,000 frames).

- It is a wide-spread practice among benchmarking professionals to allow a certain low rate of frame loss for a long time [TOL2001] and commercially available network performance testers allow to specify a parameter usually called as "Loss Tolerance" to express a zero or non-zero acceptance criterion for throughput measurements.

- Today network packet forwarding and manipulation is often implemented in software. They do not work the same as the hardware-based forwarding devices, and may be affected by other processes running in the same host hardware. So it is not feasible to require 0% of frame loss in such forwarding devices.

- Forwarding devices (especially but not necessarily only the software-based ones) may today also have larger buffers and thus they may produce potentially higher latencies. As we have shown in Section 3, late packets are not really useful for the applications, and thus they are to be considered as lost ones. For being strict with the latency during throughput measurements (e.g. 10ms timeout), we should make up with the loss tolerance to provide meaningful benchmarking results.

- Likely due to the high frame loss rates can be experienced in WiFi networks, the latest development direction of TCP congestion control algorithms considers loss no more a sign of congestion (e.g. TCP BBR).

So we felt the necessity of having options to allow frame loss. Therefore, we recommend that throughput measurement with some low tolerated frame loss rates like 0.001% or 0.01% be a recognized optional test for network interconnect devices. To avoid the possibility of gaming, our recommendation is that the results of such tests MUST clearly state the applied loss tolerance rate.

6. Acknowledgements

The authors would like to thank ... (TBD)

7. IANA Considerations

This document does not make any request to IANA.

8. Security Considerations

We have no further security considerations beyond that of [RFC8219]. Perhaps they should be cited here so that they be applied not only for the benchmarking of IPv6 transition technologies, but also for the benchmarking of all network interconnect devices.

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, <https://www.rfc-editor.org/info/rfc2119>.

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

[RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, DOI 10.17487/RFC5180, May 2008, <https://www.rfc-editor.org/info/rfc5180>.

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8219] Georgescu, M., Pislaru, L., and G. Lencse, "Benchmarking Methodology for IPv6 Transition Technologies", RFC 8219, DOI 10.17487/RFC8219, August 2017, <https://www.rfc-editor.org/info/rfc8219>.

9.2. Informative References

[LEN2017] Lencse, G., Georgescu, M., and Y. Kadobayashi, "Benchmarking Methodology for DNS64 Servers", Computer Communications, vol. 109, no. 1, pp. 162-175, DOI: 10.1016/j.comcom.2017.06.004, Sep 2017, <http://www.hit.bme.hu/~lencse/publications/ECC-2017-B-M-DNS64-revised.pdf>.

[LEN2020A]
Lencse, G. and K. Shima, "Performance analysis of SIIT implementations: Testing and improving the methodology", Computer Communications, vol. 156, no. 1, pp. 54-67, DOI: 10.1016/j.comcom.2020.03.034, Apr 2020, <http://www.hit.bme.hu/~lencse/publications/ECC-2020-SIIT-Performance-published.pdf>.

[LEN2020B]
Lencse, G., "Design and Implementation of a Software Tester for Benchmarking Stateless NAT64 Gateways", under second review in IEICE Transactions on Communications, <http://www.hit.bme.hu/~lencse/publications/IEICE-2020-siitperf-revised.pdf>.

[LEN2020C]
Lencse, G., Shima, K., and A. Kovacs, "Gaming with the Throughput and the Latency Benchmarking Measurement Procedures of RFC 2544", under review in International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems, <http://www.hit.bme.hu/~lencse/publications/IJATES2-2020-Gaming-RFC2544-for-review.pdf>.

[SIITPERF]
Lencse, G. and Y. Kadobayashi, "Siitperf: An RFC 8219 compliant SIIT (stateless NAT64) tester written in C++ using DPDK", source code, available from GitHub, 2019, <https://github.com/lencsegabor/siitperf>.

[TOL2001] Tolly, K., "The real meaning of zero-loss testing", IT World Canada, 2001, <https://www.itworldcanada.com/article/kevin-tolly-the-real-meaning-of-zero-loss-testing/33066>.

Appendix A. Change Log

A.1. 00

Initial version.

Authors' Addresses

Gabor Lencse
Budapest University of Technology and Economics
Magyar Tudosok korutja 2.
Budapest H-1117
Hungary

Email: lencse@hit.bme.hu

Keiichi Shima
IIJ Innovation Institute
Iidabashi Grand Bloom, 2-10-2 Fujimi
Chiyoda-ku, Tokyo 102-0071
Japan

Email: keiichi@iijlab.net

Benchmarking Working Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2020

M. Konstantynowicz, Ed.
P. Mikus, Ed.
Cisco Systems

July 08, 2019

NFV Service Density Benchmarking
draft-mkonstan-nf-service-density-01

Abstract

Network Function Virtualization (NFV) system designers and operators continuously grapple with the problem of qualifying performance of network services realised with software Network Functions (NF) running on Commercial-Off-The-Shelf (COTS) servers. One of the main challenges is getting repeatable and portable benchmarking results and using them to derive deterministic operating range that is production deployment worthy.

This document specifies benchmarking methodology for NFV services that aims to address this problem space. It defines a way for measuring performance of multiple NFV service instances, each composed of multiple software NFs, and running them at a varied service "packing" density on a single server.

The aim is to discover deterministic usage range of NFV system. In addition specified methodology can be used to compare and contrast different NFV virtualization technologies.

Status of This Memo

Copyright Notice

Table of Contents

1. Terminology

- NFV: Network Function Virtualization, a general industry term describing network functionality implemented in software.

- NFV service: a software based network service realized by a topology of interconnected constituent software network function applications.

- NFV service instance: a single instantiation of NFV service.

- Data-plane optimized software: any software with dedicated threads handling data-plane packet processing e.g. FD.io VPP (Vector Packet Processor), OVS-DPDK.

- Packet Loss Ratio (PLR): ratio of packets received relative to packets transmitted over the test trial duration, calculated using formula: PLR = (pkts_transmitted - pkts_received) / pkts_transmitted. For bi-directional throughput tests aggregate PLR is calculated based on the aggregate number of packets transmitted and received.

- Packet Throughput Rate: maximum packet offered load DUT/SUT forwards within the specified Packet Loss Ratio (PLR). In many cases the rate depends on the frame size processed by DUT/SUT. Hence packet throughput rate MUST be quoted with specific frame size as received by DUT/SUT during the measurement. For bi-directional tests, packet throughput rate should be reported as aggregate for both directions. Measured in packets-per-second (pps) or frames-per-second (fps), equivalent metrics.

- Non Drop Rate (NDR): maximum packet/bandwith throughput rate sustained by DUT/SUT at PLR equal zero (zero packet loss) specific to tested frame size(s). MUST be quoted with specific packet size as received by DUT/SUT during the measurement. Packet NDR measured in packets-per-second (or fps), bandwidth NDR expressed in bits-per-second (bps).

- Partial Drop Rate (PDR): maximum packet/bandwith throughput rate sustained by DUT/SUT at PLR greater than zero (non-zero packet loss) specific to tested frame size(s). MUST be quoted with specific packet size as received by DUT/SUT during the measurement. Packet PDR measured in packets-per-second (or fps), bandwidth PDR expressed in bits-per-second (bps).

- Maximum Receive Rate (MRR): packet/bandwidth rate regardless of PLR sustained by DUT/SUT under specified Maximum Transmit Rate (MTR) packet load offered by traffic generator. MUST be quoted with both specific packet size and MTR as received by DUT/SUT during the measurement. Packet MRR measured in packets-per-second (or fps), bandwidth MRR expressed in bits-per-second (bps).

2. Motivation

2.1. Problem Description

Network Function Virtualization (NFV) system designers and operators continuously grapple with the problem of qualifying performance of network services realised with software Network Functions (NF) running on Commercial-Off-The-Shelf (COTS) servers. One of the main challenges is getting repeatable and portable benchmarking results and using them to derive deterministic operating range that is production deployment worthy.

Lack of well defined and standardised NFV centric performance methodology and metrics makes it hard to address fundamental questions that underpin NFV production deployments:

1. What NFV service and how many instances can run on a single compute node?

2. How to choose the best compute resource allocation scheme to maximise service yield per node?

3. How do different NF applications compare from the service density perspective?

4. How do the virtualisation technologies compare e.g. Virtual Machines, Containers?

Getting answers to these points should allow designers to make data based decisions about the NFV technology and service design best suited to meet requirements of their use cases. Thereby obtained benchmarking data would aid in selection of the most appropriate NFV infrastructure design and platform and enable more accurate capacity planning, an important element for commercial viability of the NFV service.

2.2. Proposed Solution

The primary goal of the proposed benchmarking methodology is to focus on NFV technologies used to construct NFV services. More specifically to i) measure packet data-plane performance of multiple NFV service instances while running them at varied service "packing" densities on a single server and ii) quantify the impact of using multiple NFs to construct each NFV service instance and introducing multiple packet processing hops and links on each packet path.

The overarching aim is to discover a set of deterministic usage ranges that are of interest to NFV system designers and operators. In addition, specified methodology can be used to compare and contrast different NFV virtualisation technologies.

In order to ensure wide applicability of the benchmarking methodology, the approach is to separate NFV service packet processing from the shared virtualisation infrastructure by decomposing the software technology stack into three building blocks:

```
                 +-------------------------------+
                 |          NFV Service          |
                 +-------------------------------+
                 |   Virtualization Technology   |
                 +-------------------------------+
                 |        Host Networking        |
                 +-------------------------------+
```

Figure 1. NFV software technology stack.

Proposed methodology is complementary to existing NFV benchmarking industry efforts focusing on vSwitch benchmarking [RFC8204], [TST009] and extends the benchmarking scope to NFV services.

This document does not describe a complete benchmarking methodology, instead it is focusing on the system under test configuration. Each of the compute node configurations identified in this document is to be evaluated for NFV service data-plane performance using existing and/or emerging network benchmarking standards. This may include methodologies specified in [RFC2544], [TST009], [draft-vpolak-mkonstan-bmwg-mlrsearch] and/or [draft-vpolak-bmwg-plrsearch].

3. NFV Service

It is assumed that each NFV service instance is built of one or more constituent NFs and is described by: topology, configuration and resulting packet path(s).

Each set of NFs forms an independent NFV service instance, with multiple sets present in the host.

3.1. Topology

NFV topology describes the number of network functions per service instance, and their inter-connections over packet interfaces. It includes all point-to-point virtual packet links within the compute node, Layer-2 Ethernet or Layer-3 IP, including the ones to host networking data-plane.

Theoretically, a large set of possible NFV topologies can be realised using software virtualisation topologies, e.g. ring, partial -/full-mesh, star, line, tree, ladder. In practice however, only a few topologies are in the actual use as NFV services mostly perform either bumps-in-a-wire packet operations (e.g. security filtering/inspection, monitoring/telemetry) and/or inter-site forwarding decisions (e.g. routing, switching).

Two main NFV topologies have been identified so far for NFV service density benchmarking:

1. Chain topology: a set of NFs connect to host data-plane with minimum of two virtual interfaces each, enabling host data-plane to facilitate NF to NF service chain forwarding and provide connectivity with external network.

2. Pipeline topology: a set of NFs connect to each other in a line fashion with edge NFs homed to host data-plane. Host data-plane provides connectivity with external network.

In both cases multiple NFV service topologies are running in parallel. Both topologies are shown in figures 2. and 3. below.

NF chain topology:

```
   +-----------------------------------------------------------+
   |                     Host Compute Node                     |
   |                                                           |
   |    SmNF1       SmNF2                   SmNFn   Service-m  |
   |     ...         ...                     ...       ...     |
   |    S2NF1       S2NF2                   S2NFn   Service-2  |
   | +--------+  +--------+              +--------+            |
   | |  S1NF1 |  |  S1NF2 |              |  S1NFn |            |
   | |        |  |        |     ....     |        | Service-1  |
   | |        |  |        |              |        |            |
   | +-+----+-+  +-+----+-+    +    +    +-+----+-+            |
   |   |    |      |    |      |    |      |    |   Virtual    |
   |   |    |<-CS->|    |<-CS->|    |<-CS->|    |   Interfaces |
   | +-+----+------+----+------+----+------+----+-+            |
   | |                                            | CS: Chain  |
   | |                                            |   Segment  |
   | |             Host Data-Plane                |            |
   | +-+--+----------------------------------+--+-+            |
   |   |  |                                  |  |              |
   +-----------------------------------------------------------+
       |  |                                  |  |   Physical
       |  |                                  |  |   Interfaces
   +---+--+----------------------------------+--+--------------+
   |                                                           |
   |                    Traffic Generator                      |
   |                                                           |
   +-----------------------------------------------------------+
```

Figure 2. NF chain topology forming a service instance.

NF pipeline topology:

```
   +-----------------------------------------------------------+
   |                     Host Compute Node                     |
   |                                                           |
   |    SmNF1       SmNF2                   SmNFn   Service-m  |
   |     ...         ...                     ...       ...     |
   |    S2NF1       S2NF2                   S2NFn   Service-2  |
   | +--------+  +--------+              +--------+            |
   | |  S1NF1 |  |  S1NF2 |              |  S1NFn |            |
   | |        +--+        +--+  ....  +--+        |  Service1  |
   | |        |  |        |              |        |            |
   | +-+------+  +--------+              +------+-+            |
   |   |                                        |   Virtual    |
   |   |<-Pipeline Edge          Pipeline Edge->|   Interfaces |
   | +-+----------------------------------------+-+            |
   | |                                            |            |
   | |                                            |            |
   | |               Host Data-Plane              |            |
   | +-+--+----------------------------------+--+-+            |
   |   |  |                                  |  |              |
   +---+--+----------------------------------+--+--------------+
       |  |                                  |  |   Physical
       |  |                                  |  |   Interfaces
   +---+--+----------------------------------+--+--------------+
   |                                                           |
   |                    Traffic Generator                      |
   |                                                           |
   +-----------------------------------------------------------+
```

Figure 3. NF pipeline topology forming a service instance.

3.2. Configuration

NFV configuration includes all packet processing functions in NFs including Layer-2, Layer-3 and/or Layer-4-to-7 processing as appropriate to specific NF and NFV service design. L2 sub- interface encapsulations (e.g. 802.1q, 802.1ad) and IP overlay encapsulation (e.g. VXLAN, IPSec, GRE) may be represented here too as appropriate, although in most cases they are used as external encapsulation and handled by host networking data-plane.

NFV configuration determines logical network connectivity that is Layer-2 and/or IPv4/IPv6 switching/routing modes, as well as NFV service specific aspects. In the context of NFV density benchmarking methodology the initial focus is on logical network connectivity between the NFs, and no NFV service specific configurations. NF specific functionality is emulated using IPv4/IPv6 routing.

Building on the two identified NFV topologies, two common NFV configurations are considered:

1. Chain configuration:

   * Relies on chain topology to form NFV service chains.

   * NF packet forwarding designs:

     + IPv4/IPv6 routing.

   * Requirements for host data-plane:

     + L2 switching with L2 forwarding context per each NF chain segment, or

     + IPv4/IPv6 routing with IP forwarding context per each NF chain segment or per NF chain.

2. Pipeline configuration:

   * Relies on pipeline topology to form NFV service pipelines.

   * Packet forwarding designs:

     + IPv4/IPv6 routing.

   * Requirements for host data-plane:

     + L2 switching with L2 forwarding context per each NF pipeline edge link, or

     + IPv4/IPv6 routing with IP forwarding context per each NF pipeline edge link or per NF pipeline.

3.3. Packet Path(s)

NFV packet path(s) describe the actual packet forwarding path(s) used for benchmarking, resulting from NFV topology and configuration. They are aimed to resemble true packet forwarding actions during the NFV service lifecycle.

Based on the specified NFV topologies and configurations two NFV packet paths are taken for benchmarking:

1. Snake packet path

   * Requires chain topology and configuration.

* Packets enter the NFV chain through one edge NF and progress to the other edge NF of the chain.

* Within the chain, packets follow a zigzagging "snake" path entering and leaving host data-plane as they progress through the NF chain.

* Host data-plane is involved in packet forwarding operations between NIC interfaces and edge NFs, as well as between NFs in the chain.

2. Pipeline packet path

    * Requires pipeline topology and configuration.

    * Packets enter the NFV chain through one edge NF and progress to the other edge NF of the pipeline.

    * Within the chain, packets follow a straight path entering and leaving subsequent NFs as they progress through the NF pipeline.

    * Host data-plane is involved in packet forwarding operations between NIC interfaces and edge NFs only.

Both packet paths are shown in figures below.

Snake packet path:

```
   +-----------------------------------------------------------+
   |                     Host Compute Node                     |
   |                                                           |
   |    SmNF1       SmNF2                   SmNFn   Service-m  |
   |     ...         ...                     ...       ...     |
   |    S2NF1       S2NF2                   S2NFn   Service-2  |
   | +--------+  +--------+              +--------+            |
   | |  S1NF1 |  |  S1NF2 |              |  S1NFn |            |
   | |        |  |        |     ....     |        | Service1   |
   | |  XXXX  |  |  XXXX  |              |  XXXX  |            |
   | +-+X--X+-+  +-+X--X+-+    +X  X+    +-+X--X+-+            |
   |   |X  X|      |X  X|      |X  X|      |X  X|   Virtual    |
   |   |X  X|      |X  X|      |X  X|      |X  X|   Interfaces |
   | +-+X--X+------+X--X+------+X--X+------+X--X+-+            |
   | |  X  XXXXXXXXXX  XXXXXXXXXX  XXXXXXXXXX  X  |            |
   | |  X                                      X  |            |
   | |  X          Host Data-Plane             X  |            |
   | +-+X-+----------------------------------+-X+-+            |
   |   |X |                                  | X|              |
   +----X--------------------------------------X---------------+
       |X |                                  | X|   Physical
       |X |                                  | X|   Interfaces
   +---+X-+----------------------------------+-X+--------------+
   |                                                           |
   |                    Traffic Generator                      |
   |                                                           |
   +-----------------------------------------------------------+
```

Figure 4. Snake packet path thru NF chain topology.

Pipeline packet path:

```
   +----------------------------------------------------------+
   |                     Host Compute Node                    |
   |                                                          |
   |    SmNF1       SmNF2                 SmNFn   Service-m   |
   |     ...         ...                   ...       ...      |
   |    S2NF1       S2NF2                 S2NFn   Service-2   |
   |  +--------+  +--------+             +--------+           |
   |  |  S1NF1 |  |  S1NF2 |             |  S1NFn |           |
   |  |        +--+        +--+  ....  +--+        |  Service1 |
   |  |  XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXX  |           |
   |  +--X-----+  +--------+             +-----X--+           |
   |    |X                                     X|   Virtual   |
   |    |X                                     X|   Interfaces|
   |  +-+X-------------------------------------X+-+           |
   |  |  X                                     X  |           |
   |  |  X                                     X  |           |
   |  |  X          Host Data-Plane            X  |           |
   |  +-+X-+-------------------------------+-X+-+           |
   |    |X |                               | X|               |
   +----X--------------------------------------X---------------+
       |X |                               | X|   Physical
       |X |                               | X|   Interfaces
   +---+X-+-------------------------------+-X+--------------+
   |                                                          |
   |                     Traffic Generator                    |
   |                                                          |
   +----------------------------------------------------------+
```

Figure 5. Pipeline packet path thru NF pipeline topology.

In all cases packets enter NFV system via shared physical NIC interfaces controlled by shared host data-plane, are then associated with specific NFV service (based on service discriminator) and subsequently are cross- connected/switched/routed by host data-plane to and through NF topologies per one of the above listed schemes.

4. Virtualization Technology

NFV services are built of composite isolated NFs, with virtualisation technology providing the workload isolation. Following virtualisation technology types are considered for NFV service density benchmarking:

1. Virtual Machines (VMs)

    * Relying on host hypervisor technology e.g. KVM, ESXi, Xen.

    * NFs running in VMs are referred to as VNFs.

2. Containers

   * Relying on Linux container technology e.g. LXC, Docker.

   * NFs running in Containers are referred to as CNFs.

Different virtual interface types are available to VNFs and CNFs:

1. VNF

   * virtio-vhostuser: fully user-mode based virtual interface.

   * virtio-vhostnet: involves kernel-mode based backend.

2. CNF

   * memif: fully user-mode based virtual interface.

   * af_packet: involves kernel-mode based backend.

   * (add more common ones)

5. Host Networking

Host networking data-plane is the central shared resource that underpins creation of NFV services. It handles all of the connectivity to external physical network devices through physical network connections using NICs, through which the benchmarking is done.

Assuming that NIC interface resources are shared, here is the list of widely available host data-plane options for providing packet connectivity to/from NICs and constructing NFV chain and pipeline topologies and configurations:

- Linux Kernel-Mode Networking.

- Linux User-Mode vSwitch.

- Virtual Machine vSwitch.

- Linux Container vSwitch.

- SRIOV NIC Virtual Function - note: restricted support for chain and pipeline topologies, as it requires hair-pinning through the NIC and oftentimes also through external physical switch.

Analysing properties of each of these options and their Pros/Cons for specified NFV topologies and configurations is outside the scope of this document.

From all listed options, performance optimised Linux user-mode vswitch deserves special attention. Linux user-mode switch decouples NFV service from the underlying NIC hardware, offers rich multi-tenant functionality and most flexibility for supporting NFV services. But in the same time it is consuming compute resources and is harder to benchmark in NFV service density scenarios.

Following sections focus on using Linux user-mode vSwitch, focusing on its performance benchmarking at increasing levels of NFV service density.

6. NFV Service Density Matrix

In order to evaluate performance of multiple NFV services running on a compute node, NFV service instances are benchmarked at increasing density, allowing to construct an NFV Service Density Matrix. Table below shows an example of such a matrix, capturing number of NFV service instances (row indices), number of NFs per service instance (column indices) and resulting total number of NFs (values).

NFV Service Density - NF Count View

SVC	001	002	004	006	008	00N
001	1	2	4	6	8	1*N
002	2	4	8	12	16	2*N
004	4	8	16	24	32	4*N
006	6	12	24	36	48	6*N
008	8	16	32	48	64	8*N
00M	M*1	M*2	M*4	M*6	M*8	M*N

RowIndex: Number of NFV Service Instances, 1..M.
ColumnIndex: Number of NFs per NFV Service Instance, 1..N.
Value: Total number of NFs running in the system.

In order to deliver good and repeatable network data-plane performance, NFs and host data-plane software require direct access to critical compute resources. Due to a shared nature of all resources on a compute node, a clearly defined resource allocation scheme is defined in the next section to address this.

In each tested configuration host data-plane is a gateway between the external network and the internal NFV network topologies. Offered packet load is generated and received by an external traffic generator per usual benchmarking practice.

It is proposed that benchmarks are done with the offered packet load distributed equally across all configured NFV service instances. This approach should provide representative benchmarking data for each tested topology and configuraiton, and a good guesstimate of maximum performance required for capacity planning.

Following sections specify compute resource allocation, followed by examples of applying NFV service density methodology to VNF and CNF benchmarking use cases.

7. Compute Resource Allocation

Performance optimized NF and host data-plane software threads require timely execution of packet processing instructions and are very sensitive to any interruptions (or stalls) to this execution e.g. cpu core context switching, or cpu jitter. To that end, NFV service density methodology treats controlled mapping ratios of data plane software threads to physical processor cores with directly allocated cache hierarchies as the first order requirement.

Other compute resources including memory bandwidth and PCIe bandwidth have lesser impact and as such are subject for further study. For more detail and deep-dive analysis of software data plane performance and impact on different shared compute resources is available in [BSDP].

It is assumed that NFs as well as host data-plane (e.g. vswitch) are performance optimized, with their tasks executed in two types of software threads:

- o data-plane - handling data-plane packet processing and forwarding, time critical, requires dedicated cores. To scale data-plane performance, most NF apps use multiple data-plane threads and rely on NIC RSS (Receive Side Scaling), virtual interface multi-queue and/or integrated software hashing to distribute packets across the data threads.

- o main-control - handling application management, statistics and control-planes, less time critical, allows for core sharing. For most NF apps this is a single main thread, but often statistics (counters) and various control protocol software are run in separate threads.

Core mapping scheme described below allocates cores for all threads of specified type belonging to each NF app instance, and separately lists number of threads to a number of logical/physical core mappings for processor configurations with enabled/disabled Symmetric Multi-Threading (SMT) (e.g. AMD SMT, Intel Hyper-Threading).

If NFV service density benchmarking is run on server nodes with Symmetric Multi-Threading (SMT) (e.g. AMD SMT, Intel Hyper-Threading) for higher performance and efficiency, logical cores allocated to data- plane threads should be allocated as pairs of sibling logical cores corresponding to the hyper-threads running on the same physical core.

Separate core ratios are defined for mapping threads of vSwitch and NFs. In order to get consistent benchmarking results, the mapping ratios are enforced using Linux core pinning.

application	thread type	app:core ratio	threads/pcores (SMT disabled)	threads/lcores map (SMT enabled)
vSwitch-1c	data	1:1	1DT/1PC	2DT/2LC
	main	1:S2	1MT/S2PC	1MT/1LC
vSwitch-2c	data	1:2	2DT/2PC	4DT/4LC
	main	1:S2	1MT/S2PC	1MT/1LC
vSwitch-4c	data	1:4	4DT/4PC	8DT/8LC
	main	1:S2	1MT/S2PC	1MT/1LC
NF-0.5c	data	1:S2	1DT/S2PC	1DT/1LC
	main	1:S2	1MT/S2PC	1MT/1LC
NF-1c	data	1:1	1DT/1PC	2DT/2LC
	main	1:S2	1MT/S2PC	1MT/1LC
NF-2c	data	1:2	2DT/2PC	4DT/4LC
	main	1:S2	1MT/S2PC	1MT/1LC

o Legend to table

   * Header row

      + application - network application with optimized data-plane, a vSwitch or Network Function (NF) application.

  + thread type - either "data", short for data-plane; or "main", short for all main-control threads.

  + app:core ratio - ratio of per application instance threads of specific thread type to physical cores.

  + threads/pcores (SMT disabled) - number of threads of specific type (DT for data-plane thread, MT for main thread) running on a number of physical cores, with SMT disabled.

  + threads/lcores map (SMT enabled) - number of threads of specific type (DT, MT) running on a number of logical cores, with SMT enabled. Two logical cores per one physical core.

* Content rows

  + vSwitch-(1c|2c|4c) - vSwitch with 1 physical core (or 2, or 4) allocated to its data-plane software worker threads.

  + NF-(0.5c|1c|2c) - NF application with half of a physical core (or 1, or 2) allocated to its data-plane software worker threads.

  + Sn - shared core, sharing ratio of (n).

  + DT - data-plane thread.

  + MT - main-control thread.

  + PC - physical core, with SMT/HT enabled has many (mostly 2 today) logical cores associated with it.

  + LC - logical core, if more than one lc get allocated in sets of two sibling logical cores running on the same physical core.

  + SnPC - shared physical core, sharing ratio of (n).

  + SnLC - shared logical core, sharing ratio of (n).

Maximum benchmarked NFV service densities are limited by a number of physical cores on a compute node.

A sample physical core usage view is shown in the matrix below.

NFV Service Density - Core Usage View
vSwitch-1c, NF-1c

SVC	001	002	004	006	008	010
001	2	3	6	9	12	15
002	3	6	12	18	24	30
004	6	12	24	36	48	60
006	9	18	36	54	72	90
008	12	24	48	72	96	120
010	15	30	60	90	120	150

RowIndex: Number of NFV Service Instances, 1..10.
ColumnIndex: Number of NFs per NFV Service Instance, 1..10.
Value: Total number of physical processor cores used for NFs.

8. NFV Service Data-Plane Benchmarking

NF service density scenarios should have their data-plane performance benchmarked using existing and/or emerging network benchmarking standards as noted earlier.

Following metrics should be measured (or calculated) and reported:

- Packet throughput rate (packets-per-second)
  - Specific to tested packet size or packet sequence (e.g. some type of packet size mix sent in recurrent sequence).
  - Applicable types of throughput rate: NDR, PDR, MRR.
- (Calculated) Bandwidth throughput rate (bits-per-second) corresponding to the measured packet throughput rate.
- Packet one-way latency (seconds)
  - Measured at different packet throughput rates load e.g. light, medium, heavy.

Listed metrics should be itemized per service instance and per direction (e.g. forward/reverse) for latency.

9. Sample NFV Service Density Benchmarks

To illustrate defined NFV service density applicability, following sections describe three sets of NFV service topologies and configurations that have been benchmarked in open-source: i) in [LFN-FDio-CSIT], a continuous testing and data-plane benchmarking project, ii) as part of CNCF CNF Testbed initiative [CNCF-CNF-Testbed] and iii) in OPNFV NFVbench project.

In the first two cases each NFV service instance definition is based on the same set of NF applications, and varies only by network addressing configuration to emulate multi-tenant operating environment.

OPNFV NFVbench project is focusing on benchmarking the actual production deployments that are aligned with OPNFV specifications.

9.1. Intrepreting the Sample Results

TODO How to interpret and avoid misreading included results? And how to avoid falling into the trap of using these results to draw generilized conclusions about performance of different virtualization technologies, e.g. VM and Containers, irrespective of deployment scenarios and what VNFs and CNFs are in the actual use.

9.2. Benchmarking MRR Throughput

Initial NFV density throughput benchmarks have been performed using Maximum Receive Rate (MRR) test methodology defined and used in FD.io CSIT.

MRR tests measure the packet forwarding rate under specified Maximum Transmit Rate (MTR) packet load offered by traffic generator over a set trial duration, regardless of packet loss ratio (PLR). MTR for specified Ethernet frame size was set to the bi-directional link rate, 2x 10GbE in referred results.

Tests were conducted with two traffic profiles: i) continuous stream of 64B frames, ii) continuous stream of IMIX sequence of (7x 64B, 4x 570B, 1x 1518B), all sizes are L2 untagged Ethernet.

NFV service topologies tested include: VNF service chains, CNF service chains and CNF service pipelines.

9.3. VNF Service Chain

VNF Service Chain (VSC) topology is tested with KVM hypervisor (Ubuntu 18.04-LTS), with NFV service instances consisting of NFs running in VMs (VNFs). Host data-plane is provided by FD.io VPP vswitch. Virtual interfaces are virtio-vhostuser. Snake forwarding packet path is tested using [TRex] traffic generator, see figure.

```
   +-----------------------------------------------------------+
   |                     Host Compute Node                     |
   |                                                           |
   | +--------+  +--------+              +--------+            |
   | | S1VNF1 |  | S1VNF2 |              | S1VNFn |            |
   | |        |  |        |     ....     |        | Service1   |
   | |  XXXX  |  |  XXXX  |              |  XXXX  |            |
   | +-+X--X+-+  +-+X--X+-+              +-+X--X+-+            |
   |   |X  X|      |X  X|                  |X  X|   Virtual    |
   |   |X  X|      |X  X|      |X  X|      |X  X|   Interfaces |
   | +-+X--X+------+X--X+------+X--X+------+X--X+-+            |
   | |  X  XXXXXXXXXX  XXXXXXXXXX  XXXXXXXXXX  X  |            |
   | |  X                                      X  |            |
   | |  X          FD.io VPP vSwitch           X  |            |
   | +-+X-+----------------------------------+-X+-+            |
   |   |X |                                  | X|              |
   +----X--------------------------------------X---------------+
       |X |                                  | X|   Physical
       |X |                                  | X|   Interfaces
   +---+X-+----------------------------------+-X+--------------+
   |                                                           |
   |                  Traffic Generator (TRex)                 |
   |                                                           |
   +-----------------------------------------------------------+
```

Figure 6. VNF service chain test setup.

9.4. CNF Service Chain

CNF Service Chain (CSC) topology is tested with Docker containers (Ubuntu 18.04-LTS), with NFV service instances consisting of NFs running in Containers (CNFs). Host data-plane is provided by FD.io VPP vswitch. Virtual interfaces are memif. Snake forwarding packet path is tested using [TRex] traffic generator, see figure.

```
+---------------------------------------------------------+
|                     Host Compute Node                   |
|                                                         |
| +--------+  +--------+              +--------+          |
| | S1CNF1 |  | S1CNF2 |              | S1CNFn |          |
| |        |  |        |     ....     |        | Service1 |
| |  XXXX  |  |  XXXX  |              |  XXXX  |          |
| +-+X--X+-+  +-+X--X+-+              +-+X--X+-+          |
|   |X  X|      |X  X|                  |X  X|   Virtual  |
|   |X  X|      |X  X|      |X  X|      |X  X|   Interfaces |
| +-+X--X+------+X--X+------+X--X+------+X--X+-+          |
| |  X  XXXXXXXXXX  XXXXXXXXXX  XXXXXXXXXX  X  |          |
| |  X                                      X  |          |
| |  X          FD.io VPP vSwitch           X  |          |
| +-+X-+----------------------------------+-X+-+          |
|   |X |                                  | X|            |
+----X--------------------------------------X-------------+
    |X |                                  | X|   Physical
    |X |                                  | X|   Interfaces
+---+X-+----------------------------------+-X+------------+
|                                                         |
|                Traffic Generator (TRex)                 |
|                                                         |
+---------------------------------------------------------+
```

Figure 7. CNF service chain test setup.

9.5. CNF Service Pipeline

CNF Service Pipeline (CSP) topology is tested with Docker containers (Ubuntu 18.04-LTS), with NFV service instances consisting of NFs running in Containers (CNFs). Host data-plane is provided by FD.io VPP vswitch. Virtual interfaces are memif. Pipeline forwarding packet path is tested using [TRex] traffic generator, see figure.

```
+------------------------------------------------------------+
|                     Host Compute Node                      |
|                                                            |
| +--------+  +--------+              +--------+             |
| | S1NF1  |  | S1NF2  |              | S1NFn  |             |
| |        +--+        +--+  ....  +--+        |  Service1   |
| |  XXXXXXXXXXXXXXXXXXXXXXXX    XXXXXXXXXXXX  |             |
| +--X-----+  +--------+              +-----X--+             |
|    |X                                     X|   Virtual     |
|    |X                                     X|   Interfaces  |
| +-+X--------------------------------------X+-+             |
| |  X                                      X  |             |
| |  X                                      X  |             |
| |  X          FD.io VPP vSwitch           X  |             |
| +-+X-+----------------------------------+-X+-+             |
|    |X |                                  | X|              |
+----X--------------------------------------X----------------+
     |X |                                  | X|   Physical
     |X |                                  | X|   Interfaces
+---+X-+----------------------------------+-X+---------------+
|                                                            |
|                  Traffic Generator (TRex)                  |
|                                                            |
+------------------------------------------------------------+
```

Figure 8. CNF service chain test setup.

9.6. Sample Results: FD.io CSIT

FD.io CSIT project introduced NFV density benchmarking in release CSIT-1904 and published results for the following NFV service topologies and configurations:

1. VNF Service Chains

    * VNF: DPDK-L3FWD v19.02

        + IPv4 forwarding

        + NF-1c

    * vSwitch: VPP v19.04-release

        + L2 MAC switching

        + vSwitch-1c, vSwitch-2c

    * frame sizes: 64B, IMIX

2. CNF Service Chains

   * CNF: VPP v19.04-release

      + IPv4 routing

      + NF-1c

   * vSwitch: VPP v19.04-release

      + L2 MAC switching

      + vSwitch-1c, vSwitch-2c

   * frame sizes: 64B, IMIX

3. CNF Service Pipelines

   * CNF: VPP v19.04-release

      + IPv4 routing

      + NF-1c

   * vSwitch: VPP v19.04-release

      + L2 MAC switching

      + vSwitch-1c, vSwitch-2c

   * frame sizes: 64B, IMIX

More information is available in FD.io CSIT-1904 report, with specific references listed below:

o Testbed: [CSIT-1904-testbed-2n-skx]

o Test environment: [CSIT-1904-test-enviroment]

o Methodology: [CSIT-1904-nfv-density-methodology]

o Results: [CSIT-1904-nfv-density-results]

9.7. Sample Results: CNCF/CNFs

CNCF CI team introduced a CNF testbed initiative focusing on benchmaring NFV density with open-source network applications running as VNFs and CNFs. Following NFV service topologies and configurations have been tested to date:

1. VNF Service Chains

   * VNF: VPP v18.10-release

     + IPv4 routing

     + NF-1c

   * vSwitch: VPP v18.10-release

     + L2 MAC switching

     + vSwitch-1c, vSwitch-2c

   * frame sizes: 64B, IMIX

2. CNF Service Chains

   * CNF: VPP v18.10-release

     + IPv4 routing

     + NF-1c

   * vSwitch: VPP v18.10-release

     + L2 MAC switching

     + vSwitch-1c, vSwitch-2c

   * frame sizes: 64B, IMIX

3. CNF Service Pipelines

   * CNF: VPP v18.10-release

     + IPv4 routing

     + NF-1c

   * vSwitch: VPP v18.10-release

     + L2 MAC switching

     + vSwitch-1c, vSwitch-2c

* frame sizes: 64B, IMIX

More information is available in CNCF CNF Testbed github, with summary test results presented in summary markdown file, references listed below:

o Results: [CNCF-CNF-Testbed-Results]

9.8. Sample Results: OPNFV NFVbench

TODO Add short NFVbench based test description, and NFVbench sweep chart with single VM per service instance: Y-axis packet throughput rate or bandwidth throughput rate, X-axis number of concurrent service instances.

10. IANA Considerations

No requests of IANA.

11. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization of a 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.

12. Acknowledgements

Thanks to Vratko Polak of FD.io CSIT project and Michael Pedersen of the CNCF Testbed initiative for their contributions and useful suggestions. Extended thanks to Alec Hothan of OPNFV NFVbench project for numerous comments, suggestions and references to his/team work in the OPNFV/NVFbench project.

13. References

13.1. Normative References

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

13.2. Informative References

[BSDP] "Benchmarking Software Data Planes Intel(R) Xeon(R) Skylake vs. Broadwell", March 2019, <https://fd.io/wp-content/uploads/sites/34/2019/03/benchmarking_sw_data_planes_skx_bdx_mar07_2019.pdf>.

[CNCF-CNF-Testbed] "Cloud native Network Function (CNF) Testbed", July 2019, <https://github.com/cncf/cnf-testbed/>.

[CNCF-CNF-Testbed-Results] "CNCF CNF Testbed: NFV Service Density Benchmarking", December 2018, <https://github.com/cncf/cnf-testbed/blob/master/comparison/doc/cncf-cnfs-results-summary.md>.

[CSIT-1904-nfv-density-methodology] "FD.io CSIT Test Methodology: NFV Service Density", June 2019, <https://docs.fd.io/csit/rls1904/report/introduction/methodology_nfv_service_density.html>.

[CSIT-1904-nfv-density-results] "FD.io CSIT Test Results: NFV Service Density", June 2019, <https://docs.fd.io/csit/rls1904/report/vpp_performance_tests/nf_service_density/index.html>.

[CSIT-1904-test-enviroment] "FD.io CSIT Test Environment", June 2019, <https://docs.fd.io/csit/rls1904/report/vpp_performance_tests/test_environment.html>.

[CSIT-1904-testbed-2n-skx]
"FD.io CSIT Test Bed", June 2019, <https://docs.fd.io/csit/rls1904/report/introduction/physical_testbeds.html#node-xeon-skylake-2n-skx>.

[draft-vpolak-bmwg-plrsearch]
"Probabilistic Loss Ratio Search for Packet Throughput (PLRsearch)", July 2019, <https://tools.ietf.org/html/draft-vpolak-bmwg-plrsearch>.

[draft-vpolak-mkonstan-bmwg-mlrsearch]
"Multiple Loss Ratio Search for Packet Throughput (MLRsearch)", July 2019, <https://tools.ietf.org/html/draft-vpolak-mkonstan-bmwg-mlrsearch>.

[LFN-FDio-CSIT]
"Fast Data io, Continuous System Integration and Testing Project", July 2019, <https://wiki.fd.io/view/CSIT>.

[NFVbench]
"NFVbench Data Plane Performance Measurement Features", July 2019, <https://opnfv-nfvbench.readthedocs.io/en/latest/testing/user/userguide/readme.html>.

[RFC8204] Tahhan, M., O'Mahony, B., and A. Morton, "Benchmarking Virtual Switches in the Open Platform for NFV (OPNFV)", RFC 8204, DOI 10.17487/RFC8204, September 2017, <https://www.rfc-editor.org/info/rfc8204>.

[TRex] "TRex Low-Cost, High-Speed Stateful Traffic Generator", July 2019, <https://github.com/cisco-system-traffic-generator/trex-core>.

[TST009] "ETSI GS NFV-TST 009 V3.1.1 (2018-10), Network Functions Virtualisation (NFV) Release 3; Testing; Specification of Networking Benchmarks and Measurement Methods for NFVI", October 2018, <https://www.etsi.org/deliver/etsi_gs/NFV-TST/001_099/009/03.01.01_60/gs_NFV-TST009v030101p.pdf>.

Authors' Addresses

Maciek Konstantynowicz (editor)
Cisco Systems

Email: mkonstan@cisco.com

Peter Mikus (editor)
Cisco Systems

Email: pmikus@cisco.com

Internet Engineering Task Force S. Jacob, Ed.
Internet-Draft V. Nagarajan
Intended status: Informational Juniper Networks
Expires: November 5, 2020 May 4, 2020

Benchmarking Methodology for EVPN Multicasting
draft-vikjac-bmwg-evpnmultest-04

Abstract

This document defines methodologies for benchmarking IGMP proxy performance over EVPN-VXLAN.IGMP proxy over EVPN is defined in draft-ietf-bess-evpn-IGMP-mld-proxy-02, and is being deployed in data center networks. Specifically this document defines the methodologies for benchmarking IGMP proxy convergence, leave latency Scale,Core isolation, high availability and longevity.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

IGMP proxy over EVPN-VXLAN is defined in draft-ietf-bess-evpn-IGMP-mld-proxy-02,and is being deployed in data center networks. Specifically this document defines the methodologies for benchmarking IGMP proxy convergence,leave latency Scale,Core isolation, high availability and longevity.

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

All-Active Redundancy Mode: When all PEs attached to an Ethernet segment are allowed to forward known unicast traffic to/from that Ethernet segment for a given VLAN, then the Ethernet segment is defined to be operating in All-Active redundancy mode.

AA: All Active mode

CE: Customer Router/Devices/Switch.

DF: Designated Forwarder

DUT: Device under test.

EBGP: Exterior Border Gateway Protocol.

Ethernet Segment (ES): When a customer site (device or network) is connected to one or more PEs via a set of Ethernet links, then that set of links is referred to as an 'Ethernet segment'.

EVI: An EVPN instance spanning the leaf,spine devices participating in that EVPN.

EVPN: Ethernet Virtual Private Network

Ethernet Segment Identifier (ESI): A unique non-zero identifier that identifies an Ethernet segment is called an 'Ethernet Segment Identifier'.

Ethernet Tag: An Ethernet tag identifies a particular broadcast domain, e.g., a VLAN. An EVPN instance consists of one or more broadcast domains.

Interface: Physical interface of a router/switch.

IGMP: Internet Group Management Protocol

IBGP: Interior Border Gateway Protocol

IRB: Integrated routing and bridging interface

MAC: Media Access Control addresses on a PE.

MLD: Multicast Listener Discovery

NVO: Network Visualization Overlay

RT Traffic Generator.

Sub Interface Each physical Interfaces is subdivided into Logical units.

VXLAN: Virtual Extensible LAN

2. Test Topology

There are six routers in the topology. Leaf1,leaf2, leaf3,spine1,spine2 emulating a data center network. CE is a customer device connected to leaf1 and leaf2,it is configured with bridge domains in different vlans. The traffic generator is connected to CE,leaf1,leaf2,leaf3,spine1 and spine 2 to emulate multicast source and host generating IGMP join/leave.

Topology Diagram

```
    RT
                   +--------------+            +--------------+
            RT     |              |            |              +------------------
-+
         +---------+       spine1 +-----------------------------------+
                   |              |            | spine2       |       |
                   |              |            |              |       |
                   +----+---------+---------+  +----+----+----+------------------
+
                        |                   |       |    |            |
|
                        |                   |       |    |            |
|
                        |                   |       |    |            |
|
                        |                   |       |    |            |
|
                        |                   |       |    |            |
|
RT            +---------+-----+---------+---+-------+----++       +--+-----------
+----+
+-------------+               |         |                 |       |      leaf3
     |
             ++     leaf1     |         |     leaf2       |       |
     |
              |     DUT       |         |                 |       |
     |
              +----+----------+         +---+--------+----+       +--------+-----
-----+
                   |                        |        |                     |
                   |                        |        |                     |
                   |                        |        |                     |
                 +-+-----------------+------+        |                     |
                 |                   |               +--+   RT             ++  RT
                 |        CE         |  RT
                 |                   +-------+
                 +-------------------+
                  CE connected to leaf1 and leaf2 in EVPN-VXLAN Active-Active mod
e.
```

Topology 1

Topology Diagram

Figure 1

Test Setup Configurations:

Leaf1, Leaf2,Leaf3 are configured with Exterior Border Gateway protocol as the underlay protocol. The routes are advertised over it. The EVPN signaling is enabled on it in order to have the overlay reachability. Leaves are configured with "N" EVPN-VXLAN EVI’s. CE is multi homed to leaf1 and leaf2. The Interface connecting to the CE is configured with ESI per interface or ESI per vlan. Leaf1 and leaf2 are running EVPN-VXLAN AA mode to CE.

Spine1,spine2 are configured with Exterior Border Gateway protocol as the underlay protocol. The routes are advertised over it. The EVPN signaling is enabled over it to have the overlay reachability. Spines are configured with "N" EVPN-VXLAN EVI's. Traffic generators are connected spine1,spine2. Spine1 and Spine2 work as single home EVPN-VXLAN EVI's.

CE is acting as bridge configured with multiple vlans,the same vlans are configured on leaf1 and leaf2. traffic generator is connected to CE. The traffic generator acts as sender or receiver of traffic.

Depending up on the test scenarios the traffic generators will be used to generate igmp membership report or multicast traffic.

The above configuration will be serving as the base configuration for all test cases.

3. Test Cases

The following tests are conducted to measure the learning rate,leave rate,leave latency of IGMP messages which propagates in leaf and spine.

3.1. Learning Rate

Objective:

Measure the time taken to learn X1...Xn IGMP join generated by host/ hosts.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN.Traffic generator connected to leaf1 must send IGMP membership report for groups X1... Xn to a vlan present in leaf1,leaf2 which is a part of EVPN-VLXAN EVI.Measure the time taken to learn X1..Xn (*,G) entries in the DUT.

Measurement :

Measure the time taken by the DUT to learn the "X" IGMP membership report. The test is repeated for "N" times and the values are collected. The IGMP membership report learning rate is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Learning Rate = (T1+T2+..Tn)/N

3.2. Flush Rate

Objective:

Measure the time taken to Flush the X1... Xn (*,G) entries in DUT.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN.Traffic generator connected to the leaf1 must send IGMP membership report for groups X1... Xn to a vlan present in leaf1 which is a part of EVPN-VLXAN EVI. Stop the membership report from traffic generator. Measure the time taken to Flush X1..Xn (*,G) entries in the DUT.

Measurement :

Measure the time taken by the DUT to flush the "X" (*,G) entries The test is repeated for "N" times and the values are collected. The flush rate is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample. The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Flush Rate = (T1+T2+..Tn)/N

3.3. Leave Latency

Objective:

Measure the time taken by the DUT to stop forwarding the multicast traffic during the receipt of IGMP leave from RT.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN.Traffic generator connected to the leaf1 must send IGMP membership report for groups

X1... Xn to a vlan present in leaf1,leaf2 which is a part of EVPN-VLXAN EVI. Send multicast traffic from the RT port connected to spine1 to these groups requested by the leaf1. The leaf1 must receives multicast traffic.Send the IGMP leave message from the traffic generator to the leaf1. Measure the time taken by leaf1 to Flush X1..Xn (*,G) entries and stop forwarding the multicast traffic to RT.

Measurement :

Measure the time taken by the DUT to stop forwarding the multicast traffic.The test is repeated for "N" times and the values are collected. The leave latency is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Leave Latency = (T1+T2+..Tn)/N

3.4. Join Latency

Objective:

Measure the time taken by the DUT to create IGMP entries for N vlans.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to leaf1 acts as receiver of multicast traffic. Send IGMP membership report for groups X1...Xn from RT port connected to leaf1. The leaf1 has N vlans subscribed to these groups. Send multicast traffic from source.Measure the time taken to forward the multicast traffic to the receiver.

Measurement :

Measure the time taken by the DUT to forward the multicast traffic to these "N" vlans. The test is repeated for "N" times and the values are collected. The join latency is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Join Latency = (T1+T2+..Tn)/N

3.5. Leave Latency of N Vlans in DUT

Objective:

Measure the time taken by the DUT to stop forwarding the multicast traffic to N vlans during the receipt of IGMP leave messages from RT.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to leaf1 acts as receiver of multicast traffic. Send IGMP membership report for groups X1...Xn from RT port connected to leaf1. The leaf1 has N vlans subscribed to these groups. Send multicast traffic from source. Once the traffic is in steady state, send IGMP leave message to these groups.Once the leaf1 receiver the leave messages. it will flush the entries and stop forwarding the traffic to the receiver.

Measurement :

Measure the time taken by the DUT to stop forwaring the multicast traffic to these "N" vlans. The test is repeated for "N" times and the values are collected. The join latency is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Leave Latency = (T1+T2+..Tn)/N

3.6. Join Latency of N vlans in DUT working EVPN AA mode

Objective:

Measure the time taken to learn X1...Xn IGMP join generated by host/ hosts located in N vlans in DUT operating in EVPN AA mode.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. Measure time taken by the EVPN DF to forward the multicast traffic to the CE.

Measurement :

Measure the time taken by the EVPN DF to forward the multicast traffic for "N" vlans. The test is repeated for "N" times and the values are collected. The join latency is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Join Latency = (T1+T2+..Tn)/N

3.7. Leave Latency of DUT operating in EVPN AA

Objective:

Measure the time taken by the DUT to stop forwarding the multicast traffic to N vlans during the receipt of IGMP leave messages from RT.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability.The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. Once traffic reaches steady state, send IGMP leave from RT connected to CE. Measure the time taken by the EVPN DF to stop forward the multicast traffic to the CE.

Measurement :

Measure the time taken by the EVPN DF to stop forward the multicast traffic for "N" vlans. The test is repeated for "N" times and the values are collected.The leave latency is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.The measurement is carried out using external server which polls the DUT using automated scripts.

Leave Latency = (T1+T2+..Tn/N)

3.8. Join Latency with reception of Type 6 route

Objective:

Measure the time takes to forward the traffic by DUT after the receipt of type 6 join from peer MHPE in same ESI.

Topology : Topology 1

Procedure:

Configure "N" EVPN-VXLAN in leaf1,leaf2,leaf3,spine1 and spine2.Leaf1 and leaf2 are connected to CE which are working in EVPN AA mode. Configure N vlans in RT which are present in leaf1, then send IGMP join messages from RT connected to CE for groups ranging from X1...Xn to these vlans. The CE in turn forwards the IGMP messages to leaf2 operating in EVPN AA mode. leaf2 and leaf1 are working EVPN AA mode. Leaf 2 will send the type 6 join to the DUT(leaf 1).Then send traffic to these groups from spine1. Traffic flows from spine1 to CE. Measure the time taken by DUT to forward the traffic after the receipt of type 6 join from leaf1.

Measurement :

Measure the time taken by DUT to forward the multicast traffic flowing towards RT.

Repeat these test and plot the data. The test is repeated for "N" times and the values are collected. The time is calculated by averaging the values obtained from "N" samples.

Time taken by DUT to forward the traffic towards RT in sec = (T1+T2+..Tn/N)

4. Link Flap

4.1. Packet Loss measurement in DUT due to CE link Failure

Objective:

Measure the packet loss during the CE to DF(DUT) link failure.

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. The DF is the leaf1(DUT).Disable the link between DF and CE. Traffic switch to the new DF. Measure the loss of the traffic.

Measurement :

Measure the packet loss duration during the link disable. The test is repeated for "N" times and the values are collected. The packet loss duration is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.

Packet loss in sec = (T1+T2+..Tn)/N

4.2. Core Link Failure in EVPN AA

Objective:

Measure the packet loss during the DF core failure

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability.The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. The DF is the leaf1(DUT).Disable all the core links of DUT. Traffic switch to the new DF. Measure the loss of the traffic.

Measurement :

Measure the packet loss duration during the core link disable. The test is repeated for "N" times and the values are collected. The packet loss duration is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.

Packet loss in sec = (T1+T2+..Tn)/N

4.3. Routing Failure in DUT operating in EVPN-VXLAN AA

Objective:

Measure the packet loss during the DF routing failure

Topology : Topology 1

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. Leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. The DF is the leaf1(DUT). Perform restart routing DUT. Traffic switch to the new DF. Measure the loss of the traffic.

Measurement :

Measure the packet loss duration during the routing failure in DUT. The test is repeated for "N" times and the values are collected. The packet loss duration is calculated by averaging the values obtained from "N" samples. "N" is an arbitrary number to get a sufficient sample.The time measured for each sample is denoted by T1,T2...Tn.

Packet loss in sec = (T1+T2+..Tn)/N

5. High Availability

5.1. Routing Engine Fail over.

Objective:

Measure traffic loss during routing engine failover.

Topology : Topology 3

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. The DF is the leaf1(DUT). Perform routing engine failover in DUT. Traffic switch to the new DF. Measure the loss of the traffic.

Measurement :

The expectation of the test is 0 traffic loss with no change in the DF role. DUT should not withdraw any routes.But in cases where the DUT is not property synchronized between master and standby,due to that packet loss are observed. In that scenario the packet loss is measured.The test is repeated for "N" times and the values are collected.The packet loss is calculated by averaging the values obtained by "N" samples.

Packet loss in sec = (T1+T2+..Tn)/N

6. SOAK Test

This is measuring the performance of DUT running with scaled configuration with traffic over a peroid of time "T'". In each interval "t1" the parameters measured are CPU usage, memory usage, crashes.

6.1. Stability of the DUT with traffic.

Objective:

Measure the stability of the DUT in a scaled environment with traffic.

Topology : Topology 3

Procedure:

Confirm the DUT is up and running with EVPN-VXLAN. Ensure the route reachability. The RT port connected to spine1 acts the source of the multicast traffic. The RT port connected to CE acts as receiver of multicast traffic. leaf1 and leaf2 are multi homed EVPN-VXLAN EVI's running AA mode.The leaf1 and leaf2 have "N" vlans configured in EVPN-VXLAN EVI's, these vlans subscribe to multicast groups ranging from X1...Xn. Send IGMP membership report to these groups from RT connected to CE for these "N" vlans. Send multicast traffic from source to these groups. The DF is the leaf1(DUT). Traffic will be forwarded to the CE by the DF. Run the traffic for "T" time interval.

Measurement :

Take the hourly reading of CPU, process memory.There should not be any leak, crashes, CPU spikes. Th CPU spike is determined as the CPU usage which shoots at 40 to 50 percent of the average usage. The average value vary from device to device. Memory leak is determined by increase usage of the memory for EVPN-VPWS process. The expectation is under steady state the memory usage for EVPN-VXLAN,IGMP processes should not increase.

7. Acknowledgments

We would like to thank Al and Sarah for the support.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

The benchmarking tests described in this document are limited to the performance characterization of controllers in a lab environment with isolated networks. 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.

10. References

10.1. Normative References

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

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

[RFC2899] Ginoza, S., "Request for Comments Summary RFC Numbers 2800-2899", RFC 2899, DOI 10.17487/RFC2899, May 2001, <https://www.rfc-editor.org/info/rfc2899>.

10.2. Informative References

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

Appendix A. Appendix

Authors' Addresses

Sudhin Jacob (editor)
Juniper Networks
Bangalore, Karnataka 560103
India

Phone: +91 8061212543
Email: sjacob@juniper.net

Vikram Nagarajan
Juniper Networks
Bangalore, Karnataka 560103
India

Phone: +91 8061212543
Email: vikramna@juniper.net

Probabilistic Loss Ratio Search for Packet Throughput (PLRsearch)

draft-vpolak-bmwg-plrsearch-03

Abstract

This document addresses challenges while applying methodologies described in [RFC2544] to benchmarking software based NFV (Network Function Virtualization) data planes over an extended period of time, sometimes referred to as "soak testing". Packet throughput search approach proposed by this document assumes that system under test is probabilistic in nature, and not deterministic.

Status of This Memo

Copyright Notice

Table of Contents

1. Motivation

Network providers are interested in throughput a networking system can sustain.

[RFC2544] assumes loss ratio is given by a deterministic function of offered load. But NFV software systems are not deterministic enough. This makes deterministic algorithms (such as Binary Search per [RFC2544] and [draft-vpolak-mkonstan-bmwg-mlrsearch] with single trial) to return results, which when repeated show relatively high standard deviation, thus making it harder to tell what "the throughput" actually is.

We need another algorithm, which takes this indeterminism into account.

2. Relation To RFC2544

The aim of this document is to become an extension of [RFC2544] suitable for benchmarking networking setups such as software based NFV systems.

3. Terms And Assumptions

Due to the indeterministic nature of certain NFV systems that are the targetted by PLRsearch algorithm, existing network benchmarking terms are explicated and a number of new terms and assumptions are introduced.

3.1. Device Under Test

In software networking, "device" denotes a specific piece of software tasked with packet processing. Such device is surrounded with other software components (such as operating system kernel). It is not possible to run devices without also running the other components, and hardware resources are shared between both.

For purposes of testing, the whole set of hardware and software components is called "system under test" (SUT). As SUT is the part of the whole test setup performance of which can be measured by [RFC2544] methods, this document uses SUT instead of [RFC2544] DUT.

Device under test (DUT) can be re-introduced when analysing test results using whitebox techniques, but that is outside the scope of this document.

3.2. System Under Test

System under test (SUT) is a part of the whole test setup whose performance is to be benchmarked. The complete methodology contains other parts, whose performance is either already established, or not affecting the benchmarking result.

3.3. SUT Configuration

Usually, system under test allows different configurations, affecting its performance. The rest of this document assumes a single configuration has been chosen.

3.4. SUT Setup

Similarly to [RFC2544], it is assumed that the system under test has been updated with all the packet forwarding information it needs, before the trial measurements (see below) start.

3.5. Network Traffic

Network traffic is a type of interaction between system under test and the rest of the system (traffic generator), used to gather information about the system under test performance. PLRsearch is applicable only to areas where network traffic consists of packets.

3.6. Packet

Unit of interaction between traffic generator and the system under test. Term "packet" is used also as an abstraction of Ethernet frames.

3.6.1. Packet Offered

Packet can be offered, which means it is sent from traffic generator to the system under test.

Each offered packet is assumed to become received or lost in a short time.

3.6.2. Packet Received

Packet can be received, which means the traffic generator verifies it has been processed. Typically, when it is succesfully sent from the system under test to traffic generator.

It is assumed that each received packet has been caused by an offered packet, so the number of packets received cannot be larger than the number of packets offered.

3.6.3. Packet Lost

Packet can be lost, which means sent but not received in a timely manner.

It is assumed that each lost packet has been caused by an offered packet, so the number of packets lost cannot be larger than the number of packets offered.

Usually, the number of packets lost is computed as the number of packets offered, minus the number of packets received.

3.6.4. Other Packets

PLRsearch is not considering other packet behaviors known from networking (duplicated, reordered, greatly delayed), assuming the test specification reclassifies those behaviors to fit into the first three categories.

3.7. Traffic Profile

Usually, the performance of the system under test depends on a "type" of a particular packet (for example size), and "composition" if the network traffic consists of a mixture of different packet types.

Also, some systems under test contain multiple "ports" packets can be offered to and received from.

All such qualities together (but not including properties of trial measurements) are called traffic profile.

Similarly to system under test configuration, this document assumes only one traffic profile has been chosen for a particular test.

3.8. Traffic Generator

Traffic generator is the part of the whole test setup, distinct from the system under test, responsible both for offering packets in a highly predictable manner (so the number of packets offered is known), and for counting received packets in a precise enough way (to distinguish lost packets from tolerably delayed packets).

Traffic generator must offer only packets compatible with the traffic profile, and only count similarly compatible packets as received.

Criteria defining which received packets are compatible are left for test specification to decide.

3.9. Offered Load

Offered load is an aggregate rate (measured in packets per second) of network traffic offered to the system under test, the rate is kept constant for the duration of trial measurement.

3.10. Trial Measurement

Trial measurement is a process of stressing (previously setup) system under test by offering traffic of a particular offered load, for a particular duration.

After that, the system has a short time to become idle, while the traffic generator decides how many packets were lost.

After that, another trial measurement (possibly with different offered load and duration) can be immediately performed. Traffic generator should ignore received packets caused by packets offered in previous trial measurements.

3.11. Trial Duration

Duration for which the traffic generator was offering packets at constant offered load.

In theory, care has to be taken to ensure the offered load and trial duration predict integer number of packets to offer, and that the traffic generator really sends appropriate number of packets within precisely enough timed duration. In practice, such consideration do not change PLRsearch result in any significant way.

3.12. Packet Loss

Packet loss is any quantity describing a result of trial measurement.

It can be loss count, loss rate or loss ratio. Packet loss is zero (or non-zero) if either of the three quantities are zero (or non-zero, respecively).

3.12.1. Loss Count

Number of packets lost (or delayed too much) at a trial measurement by the system under test as determined by packet generator. Measured in packets.

3.12.2. Loss Rate

Loss rate is computed as loss count divided by trial duration. Measured in packets per second.

3.12.3. Loss Ratio

Loss ratio is computed as loss count divided by number of packets offered. Measured as a real (in practice rational) number between zero or one (including).

3.13. Trial Order Independent System

Trial order independent system is a system under test, proven (or just assumed) to produce trial measurement results that display trial order independence.

That means when a pair of consequent trial measurements are performed, the probability to observe a pair of specific results is the same, as the probability to observe the reversed pair of results whe performing the reversed pair of consequent measurements.

PLRsearch assumes the system under test is trial order independent.

In practice, most system under test are not entirely trial order independent, but it is not easy to devise an algorithm taking that into account.

3.14. Trial Measurement Result Distribution

When a trial order independent system is subjected to repeated trial measurements of constant duration and offered load, Law of Large Numbers implies the observed loss count frequencies will converge to a specific probability distribution over possible loss counts.

This probability distribution is called trial measurement result distribution, and it depends on all properties fixed when defining it. That includes the system under test, its chosen configuration, the chosen traffic profile, the offered load and the trial duration.

As the system is trial order independent, trial measurement result distribution does not depend on results of few initial trial measurements, of any offered load or (finite) duration.

3.15. Average Loss Ratio

Probability distribution over some (finite) set of states enables computation of probability-weighted average of any quantity evaluated on the states (called the expected value of the quantity).

Average loss ratio is simply the expected value of loss ratio for a given trial measurement result distribution.

3.16. Duration Independent System

Duration independent system is a trial order independent system, whose trial measurement result distribution is proven (or just assumed) to display practical independence from trial duration. See definition of trial duration for discussion on practical versus theoretical.

The only requirement is for average loss ratio to be independent of trial duration.

In theory, that would necessitate each trial measurement result distribution to be a binomial distribution. In practice, more distributions are allowed.

PLRsearch assumes the system under test is duration independent, at least for trial durations typically chosen for trial measurements initiated by PLRsearch.

3.17. Load Regions

For a duration independent system, trial measurement result distribution depends only on offered load.

It is convenient to name some areas of offered load space by possible trial results.

3.17.1. Zero Loss Region

A particular offered load value is said to belong to zero loss region, if the probability of seeing non-zero loss trial measurement result is exactly zero, or at least practically indistinguishable from zero.

3.17.2. Guaranteed Loss Region

A particular offered load value is said to belong to guaranteed loss region, if the probability of seeing zero loss trial measurement result (for non-negligible count of packets offered) is exactly zero, or at least practically indistinguishable from zero.

3.17.3. Non-Deterministic Region

A particular offered load value is said to belong to non-deterministic region, if the probability of seeing zero loss trial measurement result (for non-negligible count of packets offered) is practically distinguishable from both zero and one.

3.17.4. Normal Region Ordering

Although theoretically the three regions can be arbitrary sets, this document assumes they are intervals, where zero loss region contains values smaller than non-deterministic region, which in turn contains values smaller than guaranteed loss region.

3.18. Deterministic System

A hypothetical duration independent system with normal region ordering, whose non-deterministic region is extremely narrow (only present due to "practical distinguishibility" and cases when the expected number of packets offered is not and integer).

A duration independent system which is not deterministic is called non- deterministic system.

3.19. Througphput

Throughput is the highest offered load provably causing zero packet loss for trial measurements of duration at least 60 seconds.

For duration independent systems with normal region ordering, the throughput is the highest value within the zero loss region.

3.20. Deterministic Search

Any algorithm that assumes each measurement is a proof of the offered load belonging to zero loss region (or not) is called deterministic search.

This definition includes algorithms based on "composite measurements" which perform multiple trial measurements, somehow re-classifying results pointing at non-deterministic region.

Binary Search is an example of deterministic search.

Single run of a deterministic search launched against a deterministic system is guaranteed to find the throughput with any prescribed precision (not better than non-deterministic region width).

Multiple runs of a deterministic search launched against a non-deterministic system can return varied results within non-deterministic region. The exact distribution of deterministic search results depends on the algorithm used.

3.21. Probabilistic Search

Any algorithm which performs probabilistic computations based on observed results of trial measurements, and which does not assume that non-deterministic region is practically absent, is called probabilistic search.

A probabilistic search algorithm, which would assume that non-deterministic region is practically absent, does not really need to perform probabilistic computations, so it would become a deterministic search.

While probabilistic search for estimating throughput is possible, it would need a careful model for boundary between zero loss region and non-deterministic region, and it would need a lot of measurements of almost surely zero loss to reach good precision.

3.22. Loss Ratio Function

For any duration independent system, the average loss ratio depends only on offered load (for a particular test setup).

Loss ratio function is the name used for the function mapping offered load to average loss ratio.

This function is initially unknown.

3.23. Target Loss Ratio

Input parameter of PLRsearch. The average loss ratio the output of PLRsearch aims to achieve.

3.24. Critical Load

Aggregate rate of network traffic, which would lead to average loss ratio exactly matching target loss ratio, if used as the offered load for infinite many trial measurement.

3.25. Critical Load Estimate

Any quantitative description of the possible critical load PLRsearch is able to give after observing finite amount of trial measurements.

3.26. Fitting Function

Any function PLRsearch uses internally instead of the unknown loss ratio function. Typically chosen from small set of formulas (shapes) with few parameters to tweak.

3.27. Shape of Fitting Function

Any formula with few undetermined parameters.

3.28. Parameter Space

A subset of Real Coordinate Space. A point of parameter space is a vector of real numbers. Fitting function is defined by shape (a formula with parameters) and point of parameter space (specifying values for the parameters).

4. Abstract Algorithm

4.1. High level description

PLRsearch accepts some input arguments, then iteratively performs trial measurements at varying offered loads (and durations), and returns some estimates of critical load.

PLRsearch input arguments form three groups.

First group has a single argument: measurer. This is a callback (function) accepting offered load and duration, and returning the measured loss count.

Second group consists of load related arguments required for measurer to work correctly, typically minimal and maximal load to offer. Also, target loss ratio (if not hardcoded) is a required argument.

Third group consists of time related arguments. Typically the duration for the first trial measurement, duration increment per subsequent trial measurement, and total time for search. Some PLRsearch implementation may use estimation accuracy parameters as an exit condition instead of total search time.

The returned quantities should describe the final (or best) estimate of critical load. Implementers can chose any description that suits their users, typically it is average and standard deviation, or lower and upper boundary.

4.2. Main Ideas

The search tries to perform measurements at offered load close to the critical load, because measurement results at offered loads far from the critical load give less information on precise location of the critical load. As virtually every trial measurement result alters the estimate of the critical load, offered loads vary as they approach the critical load.

The only quantity of trial measurement result affecting the computation is loss count. No latency (or other information) is taken into account.

PLRsearch uses Bayesian Inference, computed using numerical integration, which takes long time to get reliable enough results. Therefore it takes some time before the most recent measurement result starts affecting subsequent offered loads and critical rate estimates.

During the search, PLRsearch spawns few processes that perform numerical computations, the main process is calling the measurer to perform trial measurements, without any significant delays between them. The durations of the trial measurements are increasing linearly, as higher number of trial measurement results take longer to process.

4.2.1. Trial Durations

[RFC2544] motivates the usage of at least 60 second duration by the idea of the system under test slowly running out of resources (such as memory buffers).

Practical results when measuring NFV software systems show that relative change of trial duration has negligible effects on average loss ratio, compared to relative change in offered load.

While the standard deviation of loss ratio usually shows some effects of trial duration, they are hard to model. So PLRsearch assumes SUT is duration independent, and chooses trial durations only based on numeric integration requirements.

4.2.2. Target Loss Ratio

(TODO: Link to why we think 1e-7 is acceptable loss ratio.)

4.3. PLRsearch Building Blocks

Here we define notions used by PLRsearch which are not applicable to other search methods, nor probabilistic systems under test in general.

4.3.1. Bayesian Inference

PLRsearch uses a fixed set of fitting function shapes, and uses Bayesian inference to track posterior distribution on each fitting function parameter space.

Specifically, the few parameters describing a fitting function become the model space. Given a prior over the model space, and trial duration results, a posterior distribution is computed, together with quantities describing the critical load estimate.

Likelihood of a particular loss count is computed using Poisson distribution of average loss rate given by the fitting function (at specific point of parameter space).

Side note: Binomial Distribution is a better fit compared to Poisson distribution (acknowledging that the number of packets lost cannot be higher than the number of packets offered), but the difference tends to be relevant only in high loss region. Using Poisson distribution lowers the impact of measurements in high loss region, thus helping the algorithm to converge towards critical load faster.

4.3.2. Iterative Search

The idea PLRsearch is to iterate trial measurements, using Bayesian inference to compute both the current estimate of the critical load and the next offered load to measure at.

The required numerical computations are done in parallel with the trial measurements.

This means the result of measurement "n" comes as an (additional) input to the computation running in parallel with measurement "n+1", and the outputs of the computation are used for determining the offered load for measurement "n+2".

Other schemes are possible, aimed to increase the number of measurements (by decreasing their duration), which would have even higher number of measurements run before a result of a measurement affects offered load.

4.3.3. Fitting Functions

To make the space of possible loss ratio functions more tractable the algorithm uses only few fitting function shapes for its predicitons. As the search algorithm needs to evaluate the function also far away from the critical load, the fitting function have to be reasonably behaved for every positive offered load, specifically cannot cannot predict non-positive packet loss ratio.

4.3.4. Measurement Impact

Results from trials far from the critical load are likely to affect the critical load estimate negatively, as the fitting functions do not need to be good approximations there. This is true mainly for guaranteed loss region, as in zero loss region even badly behaved fitting function predicts loss count to be "almost zero", so seeing a measurement confirming the loss has been zero indeed has small impact.

Discarding some results, or "suppressing" their impact with ad-hoc methods (other than using Poisson distribution instead of binomial) is not used, as such methods tend to make the overall search unstable. We rely on most of measurements being done (eventually) near the critical load, and overweighting far-off measurements (eventually) for well-behaved fitting functions.

4.3.5. Fitting Function Coefficients Distribution

To accomodate systems with different behaviours, a fitting function is expected to have few numeric parameters affecting its shape (mainly affecting the linear approximation in the critical region).

The general search algorithm can use whatever increasing fitting function, some specific functions are described later.

It is up to implementer to chose a fitting function and prior distribution of its parameters. The rest of this document assumes each parameter is independently and uniformly distributed over a common interval. Implementers are to add non-linear transformations into their fitting functions if their prior is different.

4.3.6. Exit Condition

Exit condition for the search is either the standard deviation of the critical load estimate becoming small enough (or similar), or overal search time becoming long enough.

The algorithm should report both average and standard deviation for its critical load posterior.

4.3.7. Integration

The posterior distributions for fitting function parameters are not be integrable in general.

The search algorithm utilises the fact that trial measurement takes some time, so this time can be used for numeric integration (using suitable method, such as Monte Carlo) to achieve sufficient precision.

4.3.8. Optimizations

After enough trials, the posterior distribution will be concentrated in a narrow area of the parameter space. The integration method should take advantage of that.

Even in the concentrated area, the likelihood can be quite small, so the integration algorithm should avoid underflow errors by some means, for example by tracking the logarithm of the likelihood.

4.3.9. Offered Load Selection

The simplest rule is to set offered load for next trial measurememnt equal to the current average (both over posterio and over fitting function shapes) of the critical load estimate.

Contrary to critical load estimate computation, heuristic algorithms affecting offered load selection do not introduce instability, and can help with convergence speed.

4.3.10. Trend Analysis

If the reported averages follow a trend (maybe without reaching equilibrium), average and standard deviation COULD refer to the equilibrium estimates based on the trend, not to immediate posterior values.

But such post-processing is discouraged, unless a clear reason for the trend is known. Frequently, presence of such a trend is a sign of some of PLRsearch assumption being violated (usually trial order independence or duration independence).

It is RECOMMENDED to report any trend quantification together with direct critical load estimate, so users can draw their own conclusion. Alternatively, trend analysis may be a part of exit conditions, requiring longer searches for systems displaying trends.

5. Known Implementations

The only known working implementation of PLRsearch is in Linux Foundation FD.io CSIT open-source project [FDio-CSIT-PLRsearch].

5.1. FD.io CSIT Implementation Specifics

The search receives min_rate and max_rate values, to avoid measurements at offered loads not supporeted by the traffic generator.

The implemented tests cases use bidirectional traffic. The algorithm stores each rate as bidirectional rate (internally, the algorithm is agnostic to flows and directions, it only cares about overall counts of packets sent and packets lost), but debug output from traffic generator lists unidirectional values.

5.1.1. Measurement Delay

In a sample implemenation in FD.io CSIT project, there is roughly 0.5 second delay between trials due to restrictons imposed by packet traffic generator in use (T-Rex).

As measurements results come in, posterior distribution computation takes more time (per sample), although there is a considerable constant part (mostly for inverting the fitting functions).

Also, the integrator needs a fair amount of samples to reach the region the posterior distribution is concentrated at.

And of course, speed of the integrator depends on computing power of the CPUs the algorithm is able to use.

All those timing related effects are addressed by arithmetically increasing trial durations with configurable coefficients (currently 5.1 seconds for the first trial, each subsequent trial being 0.1 second longer).

5.1.2. Rounding Errors and Underflows

In order to avoid them, the current implementation tracks natural logarithm (instead of the original quantity) for any quantity which is never negative. Logarithm of zero is minus infinity (not supported by Python), so special value "None" is used instead. Specific functions for frequent operations (such as "logarithm of sum of exponentials") are defined to handle None correctly.

5.1.3. Fitting Functions

Current implementation uses two fitting functions. In general, their estimates for critical rate differ, which adds a simple source of systematic error, on top of posterior dispersion reported by integrator. Otherwise the reported stdev of critical rate estimate is unrealistically low.

Both functions are not only increasing, but also convex (meaning the rate of increase is also increasing).

As Primitive Function to any positive function is an increasing function, and Primitive Function to any increasing function is convex function; both fitting functions were constructed as double Primitive Function to a positive function (even though the intermediate increasing function is easier to describe).

As not any function is integrable, some more realistic functions (especially with respect to behavior at very small offered loads) are not easily available.

Both fitting functions have a "central point" and a "spread", varied by simply shifting and scaling (in x-axis, the offered load direction) the function to be doubly integrated. Scaling in y-axis (the loss rate direction) is fixed by the requirement of transfer rate staying nearly constant in very high offered loads.

In both fitting functions (as they are a double Primitive Function to a symmetric function), the "central point" turns out to be equal to the aforementioned limiting transfer rate, so the fitting function parameter is named "mrr", the same quantity CSIT Maximum Receive Rate tests are designed to measure.

Both fitting functions return logarithm of loss rate, to avoid rounding errors and underflows. Parameters and offered load are not given as logarithms, as they are not expected to be extreme, and the formulas are simpler that way.

Both fitting functions have several mathematically equivalent formulas, each can lead to an overflow or underflow in different places. Overflows can be eliminated by using different exact formulas for different argument ranges. Underflows can be avoided by using approximate formulas in affected argument ranges, such ranges have their own formulas to compute. At the end, both fitting function implementations contain multiple "if" branches, discontinuities are a possibility at range boundaries.

5.1.3.1. Stretch Function

The original function (before applying logarithm) is Primitive Function to Logistic Function. The name "stretch" is used for related a function in context of neural networks with sigmoid activation function.

Formula for stretch fitting function: average loss rate (r) computed from offered load (b), mrr parameter (m) and spread parameter (a), given as InputForm of Wolfram language:

```
r = (a*(1 + E^(m/a))*Log[(E^(b/a) + E^(m/a))/(1 + E^(m/a))])/E^(m/a)
```

5.1.3.2. Erf Function

The original function is double Primitive Function to Gaussian Function. The name "erf" comes from error function, the first primitive to Gaussian.

Formula for erf fitting function: average loss rate (r) computed from offered load (b), mrr parameter (m) and spread parameter (a), given as InputForm of Wolfram language:

```
r = ((a*(E^(-((b - m)^2/a^2)) - E^(-(m^2/a^2))))/Sqrt[Pi] + m*Erfc[m/a]
   + (b - m)*Erfc[(-b + m)/a])/(1 + Erf[m/a])
```

5.1.4. Prior Distributions

The numeric integrator expects all the parameters to be distributed (independently and) uniformly on an interval (-1, 1).

As both "mrr" and "spread" parameters are positive and not dimensionless, a transformation is needed. Dimentionality is inherited from max_rate value.

The "mrr" parameter follows a Lomax Distribution with alpha equal to one, but shifted so that mrr is always greater than 1 packet per second.

The "stretch" parameter is generated simply as the "mrr" value raised to a random power between zero and one; thus it follows a Reciprocal Distribution.

5.1.5. Integrator

After few measurements, the posterior distribution of fitting function arguments gets quite concentrated into a small area. The integrator is using Monte Carlo with Importance Sampling where the biased distribution is Bivariate Gaussian distribution, with deliberately larger variance. If the generated sample falls outside (-1, 1) interval, another sample is generated.

The the center and the covariance matrix for the biased distribution is based on the first and second moments of samples seen so far (within the computation), with the following additional features designed to avoid hyper-focused distributions.

Each computation starts with the biased distribution inherited from the previous computation (zero point and unit covariance matrix is used in the first computation), but the overal weight of the data is set to the weight of the first sample of the computation. Also, the center is set to the first sample point. When additional samples come, their weight (including the importance correction) is compared to the weight of data seen so far (within the computation). If the new sample is more than one e-fold more impactful, both weight values (for data so far and for the new sample) are set to (geometric) average if the two weights. Finally, the actual sample generator uses covariance matrix scaled up by a configurable factor (8.0 by default).

This combination showed the best behavior, as the integrator usually follows two phases. First phase (where inherited biased distribution or single big sasmples are dominating) is mainly important for locating the new area the posterior distribution is concentrated at. The second phase (dominated by whole sample population) is actually relevant for the critical rate estimation.

5.1.6. Offered Load Selection

First two measurements are hardcoded to happen at the middle of rate interval and at max_rate. Next two measurements follow MRR-like logic, offered load is decreased so that it would reach target loss ratio if offered load decrease lead to equal decrease of loss rate.

Basis for offered load for next trial measurements is the integrated average of current critical rate estimate, averaged over fitting function.

There is one workaround implemented, aimed at reducing the number of consequent zero loss measurements. The workaround first stores every measurement result which loss ratio was the targed loss ratio or higher. Sorted list (called lossy loads) of such results is maintained.

When a sequence of one or more zero loss measurement results is encountered, a smallest of lossy loads is drained from the list. If the estimate average is smaller than the drained value, a weighted average of this estimate and the drained value is used as the next offered load. The weight of the drained value doubles with each additional consecutive zero loss results.

This behavior helps the algorithm with convergence speed, as it does not need so many zero loss result to get near critical load. Using the smallest (not drained yet) of lossy loads makes it sure the new offered load is unlikely to result in big loss region. Draining even if the estimate is large enough helps to discard early measurements when loss hapened at too low offered load. Current implementation adds 4 copies of lossy loads and drains 3 of them, which leads to fairly stable behavior even for somewhat inconsistent SUTs.

6. IANA Considerations

No requests of IANA.

7. Security Considerations

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

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

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

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

8. Acknowledgements

To be added.

9. References

9.1. Normative References

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[draft-vpolak-mkonstan-bmwg-mlrsearch] "Multiple Loss Ratio Search for Packet Throughput (MLRsearch)", February 2020, <https://tools.ietf.org/html/draft-vpolak-mkonstan-bmwg-mlrsearch>.

[FDio-CSIT-PLRsearch]
"FD.io CSIT Test Methodology - PLRsearch", February 2020, <https://docs.fd.io/csit/rls2001/report/introduction/methodology_data_plane_throughput/methodology_plrsearch.html>.

Authors' Addresses

Maciek Konstantynowicz (editor)
Cisco Systems

Email: mkonstan@cisco.com

Vratko Polak (editor)
Cisco Systems

Email: vrpolak@cisco.com

Benchmarking Working Group M. Konstantynowicz, Ed.
Internet-Draft V. Polak, Ed.
Intended status: Informational Cisco Systems
Expires: September 7, 2020 March 06, 2020

Multiple Loss Ratio Search for Packet Throughput (MLRsearch)
draft-vpolak-mkonstan-bmwg-mlrsearch-03

Abstract

This document proposes changes to [RFC2544], specifically to packet throughput search methodology, by defining a new search algorithm referred to as Multiple Loss Ratio search (MLRsearch for short). Instead of relying on binary search with pre-set starting offered load, it proposes a novel approach discovering the starting point in the initial phase, and then searching for packet throughput based on defined packet loss ratio (PLR) input criteria and defined final trial duration time. One of the key design principles behind MLRsearch is minimizing the total test duration and searching for multiple packet throughput rates (each with a corresponding PLR) concurrently, instead of doing it sequentially.

The main motivation behind MLRsearch is the new set of challenges and requirements posed by NFV (Network Function Virtualization), specifically software based implementations of NFV data planes. Using [RFC2544] in the experience of the authors yields often not repetitive and not replicable end results due to a large number of factors that are out of scope for this draft. MLRsearch aims to address this challenge in a simple way of getting the same result sooner, so more repetitions can be done to describe the replicability.

This Internet-Draft will expire on September 7, 2020.

Copyright Notice

Table of Contents

1. Terminology

- o Frame size: size of an Ethernet Layer-2 frame on the wire, including any VLAN tags (dot1q, dot1ad) and Ethernet FCS, but excluding Ethernet preamble and inter-frame gap. Measured in bytes.

- o Packet size: same as frame size, both terms used interchangeably.

- Device Under Test (DUT): In software networking, "device" denotes a specific piece of software tasked with packet processing. Such device is surrounded with other software components (such as operating system kernel). It is not possible to run devices without also running the other components, and hardware resources are shared between both. For purposes of testing, the whole set of hardware and software components is called "system under test" (SUT). As SUT is the part of the whole test setup performance of which can be measured by [RFC2544] methods, this document uses SUT instead of [RFC2544] DUT. Device under test (DUT) can be re-introduced when analysing test results using whitebox techniques, but this document sticks to blackbox testing.

- System Under Test (SUT): System under test (SUT) is a part of the whole test setup whose performance is to be benchmarked. The complete test setup contains other parts, whose performance is either already established, or not affecting the benchmarking result.

- Bi-directional throughput tests: involve packets/frames flowing in both transmit and receive directions over every tested interface of SUT/DUT. Packet flow metrics are measured per direction, and can be reported as aggregate for both directions and/or separately for each measured direction. In most cases bi-directional tests use the same (symmetric) load in both directions.

- Uni-directional throughput tests: involve packets/frames flowing in only one direction, i.e. either transmit or receive direction, over every tested interface of SUT/DUT. Packet flow metrics are measured and are reported for measured direction.

- Packet Loss Ratio (PLR): ratio of packets received relative to packets transmitted over the test trial duration, calculated using formula: PLR = (pkts_transmitted - pkts_received) / pkts_transmitted. For bi-directional throughput tests aggregate PLR is calculated based on the aggregate number of packets transmitted and received.

- Packet Throughput Rate: maximum packet offered load DUT/SUT forwards within the specified Packet Loss Ratio (PLR). In many cases the rate depends on the frame size processed by DUT/SUT. Hence packet throughput rate MUST be quoted with specific frame size as received by DUT/SUT during the measurement. For bi-directional tests, packet throughput rate should be reported as aggregate for both directions. Measured in packets-per-second (pps) or frames-per-second (fps), equivalent metrics.

- o Bandwidth Throughput Rate: a secondary metric calculated from packet throughput rate using formula: bw_rate = pkt_rate * (frame_size + L1_overhead) * 8, where L1_overhead for Ethernet includes preamble (8 Bytes) and inter-frame gap (12 Bytes). For bi-directional tests, bandwidth throughput rate should be reported as aggregate for both directions. Expressed in bits-per-second (bps).

- o Non Drop Rate (NDR): maximum packet/bandwith throughput rate sustained by DUT/SUT at PLR equal zero (zero packet loss) specific to tested frame size(s). MUST be quoted with specific packet size as received by DUT/SUT during the measurement. Packet NDR measured in packets-per-second (or fps), bandwidth NDR expressed in bits-per-second (bps).

- o Partial Drop Rate (PDR): maximum packet/bandwith throughput rate sustained by DUT/SUT at PLR greater than zero (non-zero packet loss) specific to tested frame size(s). MUST be quoted with specific packet size as received by DUT/SUT during the measurement. Packet PDR measured in packets-per-second (or fps), bandwidth PDR expressed in bits-per-second (bps).

- o Maximum Receive Rate (MRR): packet/bandwidth rate regardless of PLR sustained by DUT/SUT under specified Maximum Transmit Rate (MTR) packet load offered by traffic generator. MUST be quoted with both specific packet size and MTR as received by DUT/SUT during the measurement. Packet MRR measured in packets-per-second (or fps), bandwidth MRR expressed in bits-per-second (bps).

- o Trial: a single measurement step. See [RFC2544] section 23.

- o Trial duration: amount of time over which packets are transmitted in a single measurement step.

2. MLRsearch Background

Multiple Loss Ratio search (MLRsearch) is a packet throughput search algorithm suitable for deterministic systems (as opposed to probabilistic systems). MLRsearch discovers multiple packet throughput rates in a single search, with each rate associated with a distinct Packet Loss Ratio (PLR) criteria.

For cases when multiple rates need to be found, this property makes MLRsearch more efficient in terms of time execution, compared to traditional throughput search algorithms that discover a single packet rate per defined search criteria (e.g. a binary search specified by [RFC2544]). MLRsearch reduces execution time even further by relying on shorter trial durations of intermediate steps, with only the final measurements conducted at the specified final trial duration. This results in the shorter overall search execution time when compared to a traditional binary search, while guaranteeing the same results for deterministic systems.

In practice two rates with distinct PLRs are commonly used for packet throughput measurements of NFV systems: Non Drop Rate (NDR) with PLR=0 and Partial Drop Rate (PDR) with PLR>0. The rest of this document describes MLRsearch for NDR and PDR. If needed, MLRsearch can be adapted to discover more throughput rates with different pre-defined PLRs.

Similarly to other throughput search approaches like binary search, MLRsearch is effective for SUTs/DUTs with PLR curve that is continuously flat or increasing with growing offered load. It may not be as effective for SUTs/DUTs with abnormal PLR curves.

MLRsearch relies on traffic generator to qualify the received packet stream as error-free, and invalidate the results if any disqualifying errors are present e.g. out-of-sequence frames.

MLRsearch can be applied to both uni-directional and bi-directional throughput tests.

For bi-directional tests, MLRsearch rates and ratios are aggregates of both directions, based on the following assumptions:

- o Traffic transmitted by traffic generator and received by SUT/DUT has the same packet rate in each direction, in other words the offered load is symmetric.

- o SUT/DUT packet processing capacity is the same in both directions, resulting in the same packet loss under load.

3. MLRsearch Overview

The main properties of MLRsearch:

- o MLRsearch is a duration aware multi-phase multi-rate search algorithm:

  * Initial Phase determines promising starting interval for the search.

  * Intermediate Phases progress towards defined final search criteria.

      * Final Phase executes measurements according to the final search criteria.

      * Final search criteria are defined by following inputs:

         + PLRs associated with NDR and PDR.

         + Final trial duration.

         + Measurement resolution.

- Initial Phase:

   * Measure MRR over initial trial duration.

   * Measured MRR is used as an input to the first intermediate phase.

- Multiple Intermediate Phases:

   * Trial duration:

      + Start with initial trial duration in the first intermediate phase.

      + Converge geometrically towards the final trial duration.

   * Track two values for NDR and two for PDR:

      + The values are called lower_bound and upper_bound.

      + Each value comes from a specific trial measurement:

         - Most recent for that transmit rate.

         - As such the value is associated with that measurement's duration and loss.

      + A bound can be valid or invalid:

         - Valid lower_bound must conform with PLR search criteria.

         - Valid upper_bound must not conform with PLR search criteria.

         - Example of invalid NDR lower_bound is if it has been measured with non-zero loss.

- Invalid bounds are not real boundaries for the searched value:

  o They are needed to track interval widths.

- Valid bounds are real boundaries for the searched value.

- Each non-initial phase ends with all bounds valid.

- Bound can become invalid if it re-measured at a longer trial duration in a sub-sequent phase.

* Search:

  + Start with a large (lower_bound, upper_bound) interval width, that determines measurement resolution.

  + Geometrically converge towards the width goal of the phase.

  + Each phase halves the previous width goal.

    - First measurement of the next phase will be internal search which always gives a valid bound and brings the width to the new goal.

    - Only one bound then needs to be re-measured with new duration.

* Use of internal and external searches:

  + External search:

    - Measures at transmit rates outside the (lower_bound, upper_bound) interval.

    - Activated when a bound is invalid, to search for a new valid bound by multiplying (for example doubling) the interval width.

    - It is a variant of "exponential search".

  + Internal search:

    - A "binary search" that measures at transmit rates within the (lower_bound, upper_bound) valid interval, halving the interval width.

o Final Phase:

* Executed with the final test trial duration, and the final width goal that determines resolution of the overall search.

- Intermediate Phases together with the Final Phase are called Non-Initial Phases.

The main benefits of MLRsearch vs. binary search include:

- In general MLRsearch is likely to execute more trials overall, but likely less trials at a set final trial duration.

- In well behaving cases, e.g. when results do not depend on trial duration, it greatly reduces (>50%) the overall duration compared to a single PDR (or NDR) binary search over duration, while finding multiple drop rates.

- In all cases MLRsearch yields the same or similar results to binary search.

- Note: both binary search and MLRsearch are susceptible to reporting non-repeatable results across multiple runs for very bad behaving cases.

Caveats:

- Worst case MLRsearch can take longer than a binary search e.g. in case of drastic changes in behaviour for trials at varying durations.

4. Sample Implementation

Following is a brief description of a sample MLRsearch implementation, which is a simlified version of the existing implementation.

4.1. Input Parameters

1. *maximum_transmit_rate* - Maximum Transmit Rate (MTR) of packets to be used by external traffic generator implementing MLRsearch, limited by the actual Ethernet link(s) rate, NIC model or traffic generator capabilities.

2. *minimum_transmit_rate* - minimum packet transmit rate to be used for measurements. MLRsearch fails if lower transmit rate needs to be used to meet search criteria.

3. *final_trial_duration* - required trial duration for final rate measurements.

4. *initial_trial_duration* - trial duration for initial MLRsearch phase.

5. *final_relative_width* - required measurement resolution expressed as (lower_bound, upper_bound) interval width relative to upper_bound.

6. *packet_loss_ratio* - maximum acceptable PLR search criterion for PDR measurements.

7. *number_of_intermediate_phases* - number of phases between the initial phase and the final phase. Impacts the overall MLRsearch duration. Less phases are required for well behaving cases, more phases may be needed to reduce the overall search duration for worse behaving cases.

4.2. Initial Phase

1. First trial measures at configured maximum transmit rate (MTR) and discovers maximum receive rate (MRR).

   * IN: trial_duration = initial_trial_duration.

   * IN: offered_transmit_rate = maximum_transmit_rate.

   * DO: single trial.

   * OUT: measured loss ratio.

   * OUT: MRR = measured receive rate. If loss ratio is zero, MRR is set below MTR so that interval width is equal to the width goal of the first intermediate phase.

2. Second trial measures at MRR and discovers MRR2.

   * IN: trial_duration = initial_trial_duration.

   * IN: offered_transmit_rate = MRR.

   * DO: single trial.

   * OUT: measured loss ratio.

   * OUT: MRR2 = measured receive rate. If loss ratio is zero, MRR2 is set above MRR so that interval width is equal to the width goal of the first intermediate phase. MRR2 could end up being equal to MTR (for example if both measurements so far

had zero loss), which was already measured, step 3 is skipped in that case.

3. Third trial measures at MRR2.

   * IN: trial_duration = initial_trial_duration.

   * IN: offered_transmit_rate = MRR2.

   * DO: single trial.

   * OUT: measured loss ratio.

4.3. Non-Initial Phases

1. Main loop:

   1. IN: trial_duration for the current phase. Set to initial_trial_duration for the first intermediate phase; to final_trial_duration for the final phase; or to the element of interpolating geometric sequence for other intermediate phases. For example with two intermediate phases, trial_duration of the second intermediate phase is the geometric average of initial_trial_duration and final_trial_duration.

   2. IN: relative_width_goal for the current phase. Set to final_relative_width for the final phase; doubled for each preceding phase. For example with two intermediate phases, the first intermediate phase uses quadruple of final_relative_width and the second intermediate phase uses double of final_relative_width.

   3. IN: ndr_interval, pdr_interval from the previous main loop iteration or the previous phase. If the previous phase is the initial phase, both intervals are formed by a (correctly ordered) pair of MRR2 and MRR. Note that the initial phase is likely to create intervals with invalid bounds.

   4. DO: According to the procedure described in point 2., either exit the phase (by jumping to 1.7.), or calculate new transmit rate to measure with.

   5. DO: Perform the trial measurement at the new transmit rate and trial_duration, compute its loss ratio.

   6. DO: Update the bounds of both intervals, based on the new measurement. The actual update rules are numerous, as NDR

external search can affect PDR interval and vice versa, but the result agrees with rules of both internal and external search. For example, any new measurement below an invalid lower_bound becomes the new lower_bound, while the old measurement (previously acting as the invalid lower_bound) becomes a new and valid upper_bound. Go to next iteration (1.3.), taking the updated intervals as new input.

7. OUT: current ndr_interval and pdr_interval. In the final phase this is also considered to be the result of the whole search. For other phases, the next phase loop is started with the current results as an input.

2. New transmit rate (or exit) calculation (for point 1.4.):

   1. If there is an invalid bound then prepare for external search:

      + IF the most recent measurement at NDR lower_bound transmit rate had the loss higher than zero, then the new transmit rate is NDR lower_bound decreased by two NDR interval widths.

      + Else, IF the most recent measurement at PDR lower_bound transmit rate had the loss higher than PLR, then the new transmit rate is PDR lower_bound decreased by two PDR interval widths.

      + Else, IF the most recent measurement at NDR upper_bound transmit rate had no loss, then the new transmit rate is NDR upper_bound increased by two NDR interval widths.

      + Else, IF the most recent measurement at PDR upper_bound transmit rate had the loss lower or equal to PLR, then the new transmit rate is PDR upper_bound increased by two PDR interval widths.

   2. Else, if interval width is higher than the current phase goal:

      + IF NDR interval does not meet the current phase width goal, prepare for internal search. The new transmit rate is a in the middle of NDR lower_bound and NDR upper_bound.

      + IF PDR interval does not meet the current phase width goal, prepare for internal search. The new transmit rate is a in the middle of PDR lower_bound and PDR upper_bound.

3. Else, if some bound has still only been measured at a lower duration, prepare to re-measure at the current duration (and the same transmit rate). The order of priorities is:

   + NDR lower_bound,

   + PDR lower_bound,

   + NDR upper_bound,

   + PDR upper_bound.

4. Else, do not prepare any new rate, to exit the phase. This ensures that at the end of each non-initial phase all intervals are valid, narrow enough, and measured at current phase trial duration.

5. FD.io CSIT Implementation

The only known working implementation of MLRsearch is in the open-source code running in Linux Foundation FD.io CSIT project [FDio-CSIT-MLRsearch] as part of a Continuous Integration / Continuous Development (CI/CD) framework.

MLRsearch is also available as a Python package in [PyPI-MLRsearch].

5.1. Additional details

This document so far has been describing a simplified version of MLRsearch algorithm. The full algorithm as implemented in CSIT contains additional logic, which makes some of the details (but not general ideas) above incorrect. Here is a short description of the additional logic as a list of principles, explaining their main differences from (or additions to) the simplified description, but without detailing their mutual interaction.

1. Logarithmic transmit rate.

   * In order to better fit the relative width goal, the interval doubling and halving is done differently.

   * For example, the middle of 2 and 8 is 4, not 5.

2. Optimistic maximum rate.

   * The increased rate is never higher than the maximum rate.

   * Upper bound at that rate is always considered valid.

3. Pessimistic minimum rate.

   * The decreased rate is never lower than the minimum rate.

   * If a lower bound at that rate is invalid, a phase stops refining the interval further (until it gets re-measured).

4. Conservative interval updates.

   * Measurements above the current upper bound never update a valid upper bound, even if drop ratio is low.

   * Measurements below the current lower bound always update any lower bound if drop ratio is high.

5. Ensure sufficient interval width.

   * Narrow intervals make external search take more time to find a valid bound.

   * If the new transmit increased or decreased rate would result in width less than the current goal, increase/decrease more.

   * This can happen if the measurement for the other interval makes the current interval too narrow.

   * Similarly, take care the measurements in the initial phase create wide enough interval.

6. Timeout for bad cases.

   * The worst case for MLRsearch is when each phase converges to intervals way different than the results of the previous phase.

   * Rather than suffer total search time several times larger than pure binary search, the implemented tests fail themselves when the search takes too long (given by argument _timeout_).

7. Pessimistic external search.

   * Valid bound becoming invalid on re-measurement with higher duration is frequently a sign of SUT behaving in non-deterministic way (from blackbox point of view). If the final width interval goal is too narrow compared to width of rate region where SUT is non-deterministic, it is quite likely that there will be multiple invalid bounds before the external search finds a valid one.

* In this case, external search can be sped up by increasing interval width more rapidly. As only powers of two ensure the subsequent internal search will not result in needlessly narrow interval, a parameter _doublings_ is introduced to control the pessimism of external search. For example three doublings result in interval width being multiplied by eight in each external search iteration.

5.1.1. FD.io CSIT Input Parameters

1. *maximum_transmit_rate* - Typical values: 2 * 14.88 Mpps for 64B 10GE link rate, 2 * 18.75 Mpps for 64B 40GE NIC (specific model).

2. *minimum_transmit_rate* - Value: 2 * 10 kpps (traffic generator limitation).

3. *final_trial_duration* - Value: 30 seconds.

4. *initial_trial_duration* - Value: 1 second.

5. *final_relative_width* - Value: 0.005 (0.5%).

6. *packet_loss_ratio* - Value: 0.005 (0.5%).

7. *number_of_intermediate_phases* - Value: 2. The value has been chosen based on limited experimentation to date. More experimentation needed to arrive to clearer guidelines.

8. *timeout* - Limit for the overall search duration (for one search). If MLRsearch oversteps this limit, it immediatelly declares the test failed, to avoid wasting even more time on a misbehaving SUT. Value: 600 (seconds).

9. *doublings* - Number of dublings when computing new interval width in external search. Value: 2 (interval width is quadroupled). Value of 1 is best for well-behaved SUTs, but value of 2 has been found to decrease overall search time for worse-behaved SUT configurations, contributing more to the overall set of different SUT configurations tested.

5.2. Example MLRsearch Run

The following table shows data from a real test run in CSIT (using the default input values as above). The first column is the phase, the second is the trial measurement performed (aggregate bidirectional offered load in megapackets per second, and trial duration in seconds). Each of last four columns show one bound as updated after the measurement (duration truncated to save space).

Loss ratio is not shown, but invalid bounds are marked with a plus sign.

Phase	Trial	NDR lower	NDR upper	PDR lower	PDR upper
init.	37.50 1.00	N/A	37.50 1.	N/A	37.50 1.
init.	10.55 1.00	+10.55 1.	37.50 1.	+10.55 1.	37.50 1.
init.	9.437 1.00	+9.437 1.	10.55 1.	+9.437 1.	10.55 1.
int 1	6.053 1.00	6.053 1.	9.437 1.	6.053 1.	9.437 1.
int 1	7.558 1.00	7.558 1.	9.437 1.	7.558 1.	9.437 1.
int 1	8.446 1.00	8.446 1.	9.437 1.	8.446 1.	9.437 1.
int 1	8.928 1.00	8.928 1.	9.437 1.	8.928 1.	9.437 1.
int 1	9.179 1.00	8.928 1.	9.179 1.	9.179 1.	9.437 1.
int 1	9.052 1.00	9.052 1.	9.179 1.	9.179 1.	9.437 1.
int 1	9.307 1.00	9.052 1.	9.179 1.	9.179 1.	9.307 1.
int 2	9.115 5.48	9.115 5.	9.179 1.	9.179 1.	9.307 1.
int 2	9.243 5.48	9.115 5.	9.179 1.	9.243 5.	9.307 1.
int 2	9.179 5.48	9.115 5.	9.179 5.	9.243 5.	9.307 1.
int 2	9.307 5.48	9.115 5.	9.179 5.	9.243 5.	+9.307 5.

int 2	9.687 5.48	9.115 5.	9.179 5.	9.307 5.	9.687 5.
int 2	9.495 5.48	9.115 5.	9.179 5.	9.307 5.	9.495 5.
int 2	9.401 5.48	9.115 5.	9.179 5.	9.307 5.	9.401 5.
final	9.147 30.0	9.115 5.	9.147 30	9.307 5.	9.401 5.
final	9.354 30.0	9.115 5.	9.147 30	9.307 5.	9.354 30
final	9.115 30.0	+9.115 30	9.147 30	9.307 5.	9.354 30
final	8.935 30.0	8.935 30	9.115 30	9.307 5.	9.354 30
final	9.025 30.0	9.025 30	9.115 30	9.307 5.	9.354 30
final	9.070 30.0	9.070 30	9.115 30	9.307 5.	9.354 30
final	9.307 30.0	9.070 30	9.115 30	9.307 30	9.354 30

6. IANA Considerations

No requests of IANA.

7. Security Considerations

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

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

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

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

8. Acknowledgements

Many thanks to Alec Hothan of OPNFV NFVbench project for thorough review and numerous useful comments and suggestions.

9. References

9.1. Normative References

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[FDio-CSIT-MLRsearch]
"FD.io CSIT Test Methodology - MLRsearch", February 2020, <https://docs.fd.io/csit/rls2001/report/introduction/methodology_data_plane_throughput/methodology_mlrsearch_tests.html>.

[PyPI-MLRsearch]
"MLRsearch 0.3.0, Python Package Index", February 2020, <https://pypi.org/project/MLRsearch/0.3.0/>.

Authors' Addresses

Maciek Konstantynowicz (editor)
Cisco Systems

Email: mkonstan@cisco.com

Vratko Polak (editor)
Cisco Systems

Email: vrpolak@cisco.com