

Benchmarking Methodology WG
Internet Draft
Intended status: Informational
Expires: April 7, 2015

Sarah Banks
VSS Monitoring
Fernando Calabria
Cisco
Gery Czirjak
Ramdas Machat
Juniper
October 8, 2014

ISSU Benchmarking Methodology
draft-banks-bmwg-issu-meth-05

Abstract

Modern forwarding devices attempt to minimize any control and data plane disruptions while performing planned software changes, by implementing a technique commonly known as an In Service Software Upgrade (ISSU).

This document specifies a set of common methodologies and procedures designed to characterize the overall behavior of a Device Under Test (DUT), subject to an ISSU event.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to publish it as an RFC and to translate it into languages other than English.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

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

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

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

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

This Internet-Draft will expire on April 7, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction.....	3
2. Conventions used in this document.....	4
3. Generic ISSU Process, phased approach.....	5
3.1. Software Download.....	6
3.2. Software Staging.....	6
3.3. Upgrade Run.....	7
3.4. Upgrade Acceptance.....	7
4. Test Methodology.....	8
4.1. Test Topology.....	8
4.2. Load Model.....	9
5. ISSU Test Methodology.....	10
5.1. Pre-ISSU recommended verifications.....	10

5.2. Software Staging.....	10
5.3. Upgrade Run.....	11
5.4. Post ISSU verifications.....	12
5.5. ISSU under negative stimuli.....	13
6. ISSU Abort and Rollback.....	14
7. Final Report - Data Presentation - Analysis.....	14
7.1. Data collection considerations.....	16
8. Security Considerations.....	16
9. IANA Considerations.....	17
10. Conclusions.....	17
11. References.....	17
11.1. Normative References.....	17
11.2. Informative References.....	17
12. Acknowledgments.....	17

1. Introduction

As required by most Service Provider (SP) network operators, ISSU functionality has been implemented by modern forwarding devices to upgrade or downgrade from one software version to another with a goal of eliminating the downtime of the router and/or the outage of service. However, It is noted that while most operators expect that whiledesire such behavior as a elimination is the goal, minimal downtime and/or degradation of service is often expected.

The ISSU operation may apply in terms of an atomic version change of the entire system software or it may be applied in a more modular sense such as for a patch or maintenance upgrade. The procedure described herein may be used to verify either approach, as may be supported by the vendor hardware and software.

In support of this document, a set of expectations for an ISSU operation can be summarized as follows:

- The software is successfully migrated, from one version to a successive version or vice versa.
- There are no control plane interruptions throughout the process. That is, the upgrade/downgrade could be accomplished while the device remains "in service". It is noted however, that most service providers will still undertake such actions in a maintenance window (even in redundant environments) to minimize any risk.

- Interruptions to the forwarding plane are expected to be minimal to none.

- The total time to accomplish the upgrade is minimized, again to reduce potential network outage exposure (e.g. an external failure event might impact the network as it operates with reduced redundancy).

This document provides a set of procedures to characterize a given forwarding device's ISSU behavior quantitatively, from the perspective of meeting the above expectations.

Different hardware configurations may be expected to be benchmarked, but a typical configuration for a forwarding device that supports ISSU consists of at least one pair of Routing Processors (RP's) that operate in a redundant fashion, and single or multiple Forwarding Engines (Line Cards) that may or may not be redundant, as well as fabric cards or other components as applicable. However, this does not preclude the possibility that a device in question can perform ISSU functions through the operation of independent process components, which may be upgraded without impact to the overall operation of the device. As an example, perhaps the software module involved in SNMP functions can be upgraded without impacting other operations.

The concept of a multi-chassis deployment may also be characterized by the current set of proposed methodologies, but the implementation specific details (i.e. process placement and others) are beyond the scope of the current document.

Since most modern forwarding devices, where ISSU would be applicable, do consist of redundant RP's and hardware-separated control plane and data plane functionality, this document will focus on methodologies which would be directly applicable to those platforms. It is anticipated that the concepts and approaches described herein may be readily extended to accommodate other device architectures as well.

2. Conventions used in this document

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

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

In this document, the characters ">>" preceding an indented line(s) indicates a compliance requirement statement using the key words listed above. This convention aids reviewers in quickly identifying or finding the explicit compliance requirements of this RFC.

3. Generic ISSU Process, phased approach

ISSU may be viewed as the behavior of a device when exposed to a planned change in its software functionality. This may mean changes to the core operating system, separate processes or daemons or even of firmware logic in programmable hardware devices (e.g. CPLD/FPGA). The goal of an ISSU implementation is to permit such actions with minimal or no disruption to the primary operation of the device in question.

ISSU may be user initiated through direct interaction with the device or activated through some automated process on a management system or even on the device itself. For the purposes of this document, we will focus on the model where the ISSU action is initiated by direct user intervention.

The ISSU process can be viewed as a series of different phases or activities, as defined below. For each of these phases, the test operator MUST record the outcome as well as any relevant observations (defined further in the present document). Note that, a given vendor implementation may or may not permit the abortion of the in-progress ISSU at particular stages. There may also be certain restrictions as to ISSU availability given certain functional configurations (for example, ISSU in the presence of Bidirectional Failure Detection (BFD) [RFC 5880] may not be supported. It is incumbent upon the test operator to ensure that the DUT is appropriately configured to provide the appropriate test environment as needed. As with any properly orchestrated test effort, the test plan document should reflect these and other relevant details and SHOULD be written with close attention to the expected production-operating environment. The combined analysis of the results of each phase will characterize the overall ISSU process with the main goal of being able to identify and quantify any disruption in service (from the data and control plane perspective) allowing operators to plan their maintenance activities with greater precision.

The generic ISSU process can be viewed as a series of the following phases:

3.1. Software Download

In this first phase, the requested software package may be downloaded to the router and is typically stored onto a device. The downloading of software may be performed automatically by the device as part of the upgrade process, or it may be initiated separately. Such separation allows an administrator to download the new code inside or outside of a maintenance window; it is anticipated that downloading new code and saving it to disk on the router will not impact operations. In the case where the software can be downloaded outside of the actual upgrade process, the administrator SHOULD do so; downloading software can skew timing results based on factors that are often not comparative in nature. Internal compatibility verification may be performed by the software running on the DUT, to verify the checksum of the files downloaded as well as any other pertinent checks. Depending upon vendor implementation, these mechanisms may extend to include verification that the downloaded module(s) meet a set of identified pre-requisites such as hardware or firmware compatibility or minimum software requirements. Where such mechanisms are made available by the product, they should be verified, by the tester, with the perspective of avoiding operational issues in production. Verification should include both positive verification (ensuring that an ISSU action should be permitted) as well as negative tests (creation of scenarios where the verification mechanisms would report exceptions).

3.2. Software Staging

In this second phase, the requested software package is loaded into the pertinent components of a given forwarding device (typically the RP in standby state). Internal compatibility verification may be performed by the software running on the DUT, as part of the upgrade process itself, to verify the checksum of the files downloaded as well as any other pertinent checks. Depending upon vendor implementation, these mechanisms may extend to include verification that the downloaded module(s) meet a set of identified pre-requisites such as hardware or firmware compatibility or minimum software requirements. Where such mechanisms are made available by the product, they should be verified, by the tester, with the perspective of avoiding operational issues in production. In this case, the

execution of these checks is within scope of the upgrade time, and SHOULD be included in the testing results. Once the new software is downloaded to the pertinent components of the DUT, the upgrade begins and the DUT begins to prepare itself for upgrade. Depending on the vendor implementation, it is expected that redundant hardware pieces within the DUT are upgraded, including the backup or secondary RP.

3.3. Upgrade Run

In this phase, a switchover of RPs may take place, where one RP is now upgraded with the new version of software. More importantly, the "Upgrade Run" phase is where the internal changes made to information and state stored on the router, on disk and in memory, are either migrated to the "new" version of code, or transformed/rebuilt to meet the standards of the new version of code, and pushed onto the appropriate pieces of hardware. It is within this phase that any outage(s) on the control or forwarding plane MAY be expected to be observed.

This is the critical phase of the ISSU, where the control plane should not be impacted and any interruptions to the forwarding plane should be minimal to none.

For some implementations, the above two steps may be concatenated into one monolithic operation. In such case, the calculation of the respective ISSU time intervals may need to be adapted accordingly. If any control or data plane interruptions occur, it is expected to be observed and recorded within this stage.

3.4. Upgrade Acceptance

In this phase, the new version of software MUST be running in all the physical nodes of the logical forwarding device. (RP's and LC's as applicable). At this point, configuration control is returned to the operator and normal device operation i.e. outside of ISSU-oriented operation, is resumed.

4. Test Methodology

As stated by <http://tools.ietf.org/wg/bmwg/draft-ietf-bmwg-2544-as/> (when it becomes an RFC) The Test Topology Setup must be part of an ITE (Isolated Test Environment)

The reporting of results MUST take into account the repeatability considerations from Section 4 of [RFC2544]. It is RECOMMENDED to perform multiple trials and report average results. The results are reported in a simple statement including the measured frame loss and ISSU impact times.

4.1. Test Topology

The hardware configuration of the DUT (Device Under test) MUST be identical to the one expected to be or currently deployed in production in order for the benchmark to have relevance. This would include the number of RP's, hardware version, memory and initial software release, any common chassis components, such as fabric hardware in the case of a fabric-switching platform and the specific LC's (version, memory, interfaces type, rate etc.)

For the Control and Data plane, differing configuration approaches MAY be utilized. The recommended approach relies on "mimicking" the existing production data and control plane information, in order to emulate all the necessary Layer1 through Layer3 and, if appropriate, upper layer characteristics of the network, as well as end to end traffic/communication pairs. In other words, design a representative load model of the production environment and deploy a collapsed topology utilizing test tools and/or external devices, where the DUT will be tested. Note that, the negative impact of ISSU operations is likely to impact scaled, dynamic topologies to a greater extent than simpler, static environments. As such, this methodology is advised for most test scenarios.

The second, more simplistic approach is to deploy an ITE "Isolated Testing Environment" as described in some of the existing standards for benchmarking methodologies (e.g. RFC2544/RFC6815) in which endpoints are "directly" connected to the DUT. In this manner control plane information is kept to a minimum (only connected interfaces) and only a basic data plane of sources and destinations is applied. If this methodology is selected, care must be taken to understand that the systemic behavior of the ITE may not be identical to that experienced by a device in a production network role. That is, control plane validation may be minimal to none if this methodology

is employed. It may be possible to perform some degree of data plane validation with this approach.

4.2. Load Model

In consideration of the defined test topology, a load model must be developed to exercise the DUT while the ISSU event is introduced. This applied load should be defined in such a manner as to provide a granular, repeatable verification of the ISSU impact on transit traffic. Sufficient traffic load (rate) should be applied to permit timing extrapolations at a minimum granularity of 100 milliseconds e.g. 100Mbps for a 10Gbps interface. The use of steady traffic streams rather than bursty loads is preferred to simplify analysis. The traffic should be patterned to provide a broad range of source and destination pairs, which resolve to a variety of FIB (forwarding information base) prefix lengths. If the production network environment includes multicast traffic or VPN's (L2, L3 or IPSec) it is critical to include these in the model.

For mixed protocol environments (e.g. IPv4 and IPv6), frames SHOULD be distributed between the different protocols. The distribution SHOULD approximate the network conditions of deployment. In all cases, the details of the mixed protocol distribution MUST be included in the reporting.

The feature, protocol timing and other relevant configurations should be matched to the expected production environment. Deviations from the production templates may be deemed necessary by the test operator (for example, certain features may not support ISSU or the test bed may not be able to accommodate such). However, the impact of any such divergence should be clearly understood and the differences MUST be recorded in the results documentation.

It is recommended that an NMS system be deployed, preferably similar to that utilized in production. This will allow for monitoring of the DUT while it is being tested both in terms of supporting the system resource impact analysis as well as from the perspective of detecting interference with non-transit (management) traffic as a result of the ISSU operation. Additionally, a DUT management session other than snmp-based, typical of usage in production, should be established to the DUT and monitored for any disruption.

It is suggested that the actual test exercise be managed utilizing direct console access to the DUT, if at all possible to avoid the

possibility that a network interruption impairs execution of the test exercise.

All in all, the load model should attempt to simulate the production network environment to the greatest extent possible in order to maximize the applicability of the results generated.

5. ISSU Test Methodology

As previously described, for the purposes of this test document, the ISSU process is divided into three main phases. The following methodology assumes that a suitable test topology has been constructed per section 4. A description of the methodology to be applied for each of the above phases follows:

5.1. Pre-ISSU recommended verifications

Verify that enough hardware and software resources are available to complete the Load operation (enough disk space).

Verify that the redundancy states between RPs and other nodes are as expected (e.g. redundancy on, RP's synchronized).

Verify that the device, if running NSR capable routing protocols, is in a "ready" state; that is, that the sync between RPs is complete and the system is ready for failover, if necessary.

Gather a configuration snapshot of the device and all of its applicable components.

Verify that the node is operating in a "steady" state (that is, no critical or maintenance function is being currently performed).

Note any other operational characteristics that the tester may deem applicable to the specific implementation deployed.

5.2. Software Staging

Establish all relevant protocol adjacencies and stabilize routing within the test topology. In particular, ensure that the scaled

levels of the dynamic protocols are dimensioned as specified by the test topology plan.

Clear relevant logs and interface counters to simplify analysis. If possible, set logging timestamps to a highly granular mode. If the topology includes management systems, ensure that the appropriate polling levels have been applied, sessions established and that the responses are per expectation.

Apply the traffic loads as specified in the load model previously developed for this exercise.

Document an operational baseline for the test bed with relevant data supporting the above steps (include all relevant load characteristics of interest in the topology e.g. routing load, traffic volumes, memory and CPU utilization)

Note the start time (T0) and begin the code change process utilizing the appropriate mechanisms as expected to be used in production (e.g. active download with TFTP/FTP/SCP/etc. or direct install from local or external storage facility). In order to ensure that ISSU process timings are not skewed by the lack of a network wide synchronization source, the use of a network NTP source is encouraged.

Take note of any logging information and command line interface (CLI) prompts as needed (this detail will be vendor-specific). Respond to any DUT prompts in a timely manner.

Monitor the DUT for the reload of secondary RP to the new software level. Once the secondary has stabilized on the new code, note the completion time. The duration of these steps will be logged as "T1".

Review system logs for any anomalies, check that relevant dynamic protocols have remained stable and note traffic loss if any. Verify that deployed management systems have not identified any unexpected behavior.

5.3. Upgrade Run

The following assumes that the software load step and upgrade step are discretely controllable. If not, maintain the afore-mentioned timer and monitor for completion of the ISSU as described below.

Note the start time and initiate the actual upgrade procedure. Monitor the operation of the secondary route processor while it initializes with the new software and assumes mastership of the DUT.

At this point, pay particular attention to any indications of control plane disruption, traffic impact or other anomalous behavior. Once the DUT has converged upon the new code and returned to normal operation note the completion time and log the duration of this step as T2.

Review the syslog data in the DUT and neighboring devices for any behavior, which would be disruptive in a production environment (linecard reloads, control plane flaps etc.). Examine the traffic generators for any indication of traffic loss over this interval. If the Test Set reported any traffic loss, note the number of frames lost as "TP_frames". If the test set also provides outage duration, note this as TP_time (alternatively this may be calculated as TP/offered pps (packets per second) load).

Verify the DUT status observations as per any NMS systems managing the DUT and its neighboring devices. Document the observed CPU and memory statistics both during the ISSU upgrade event and after and ensure that memory and CPU have returned to an expected (previously baselined) level.

5.4. Post ISSU verifications

The following describes a set of post-ISSU verification tasks that are not directly part of the ISSU process, but are recommended for execution in order to validate a successful upgrade:

- . Configuration delta analysis

- o Examine the post-ISSU configurations to determine if any changes have occurred either through process error or due to differences in the implementation of the upgraded code.

- . Exhaustive control plane analysis

- o Review the details of the RIB and FIB to assess whether any unexpected changes have been introduced in the forwarding paths.

- . Verify that both RPs are up and that the redundancy mechanism for the control plane is enabled and fully synchronized.

- . Verify that no control plane (protocol) events or flaps were detected.

- . Verify that no L1 and or L2 interface flaps were observed.

- . Document the hitless operation or presence of an outage based upon the counter values provided by the Test Set.

5.5. ISSU under negative stimuli

As an OPTIONAL Test Case, the operator may want to perform an ISSU test while the DUT is under stress by introducing route churn to any or all of the involved phases of the ISSU process.

One approach relies on the operator to gather statistical information from the production environment and determine a specific number of routes to flap every 'fixed' or 'variable' interval. Alternatively, the operator may wish to simply pre-select a fixed number of prefixes to flap. As an example, an operator may decide to flap 1% of all the BGP routes every minute and restore them 1 minute afterwards. The tester may wish to apply this negative stimulus throughout the entire ISSU process or most importantly, during the run phase.

It is important to ensure that these routes, which are introduced solely for stress proposes, MUST not overlap the ones (per the Load Model) specifically leveraged to calculate the TP (recorded outage). Furthermore, there SHOULD NOT be 'operator induced' control plane - protocol adjacency flaps for the duration of the test process as it may adversely affect the characterization of the entire test exercise. For example, triggering IGP adjacency events may force re-computation of underlying routing tables with attendant impact to the perceived ISSU timings. While not recommended, if such trigger events are desired by the test operator, care should be taken to avoid the introduction of unexpected anomalies within the test harness.

6. ISSU Abort and Rollback

Where a vendor provides such support, the ISSU process could be aborted for any reason by the operator. However, the end results and behavior may depend on the specific phase where the process was aborted. While this is implementation dependent, as a general recommendation, if the process is aborted during the "Software Download" or "Software Staging" phases, no impact to service or device functionality should be observed. In contrast, if the process is aborted during the "Upgrade Run" or "Upgrade Accept" phases, the system may reload and revert back to the previous software release and as such, this operation may be service affecting.

Where vendor support is available, the abort/rollback functionality should be verified and the impact, if any, quantified generally following the procedures provided above.

7. Final Report - Data Presentation - Analysis

All ISSU impact results are summarized in a simple statement describing the "ISSU Disruption Impact" including the measured frame loss and impact time, where impact time is defined as the time frame determined per the TP reported outage. These are considered to be the primary data points of interest. However, the entire ISSU operational impact should also be considered in support of planning for maintenance and as such, additional reporting points are included.

Software download/secondary update	T1
Upgrade/Run	T2
ISSU Traffic Disruption (Frame Loss)	TP_frames
ISSU Traffic Impact Time (milliseconds)	TP Time
ISSU Housekeeping Interval	T3
(Time for both RP's up on new code and fully synced - Redundancy restored)	

Total ISSU Maintenance Window

T4 (sum of T1+T2+T3)

The results reporting MUST provide the following information:

- . DUT hardware and software detail
- . Test Topology definition and diagram (especially as related to the ISSU operation)
- . Load Model description including protocol mixes and any divergence from the production environment
- . Time Results as per above
- . Anomalies Observed during ISSU
- . Anomalies Observed in post-ISSU analysis

It is RECOMMENDED that the following parameters be reported in these units:

Parameter	Units or Examples
Traffic Load	Frames per second and bits per Second
Disruption (average)	Frames
Impact Time (average)	Milliseconds
Number of trials	Integer count
Protocols	IPv4, IPv6, MPLS, etc.
Frame Size	Octets
Port Media	Ethernet, Gigabit Ethernet (GbE), Packet over SONET (POS), etc.
Port Speed	10 Gbps, 1 Gbps, 100 Mbps, etc.

Interface Encap. Ethernet, Ethernet VLAN,
 PPP, High-Level Data Link Control
 (HDLC), etc.

Number of Prefixes
flapped (ON Interval)
(Optional) # of prefixes / Time (minutes)

Number of Prefixes
flapped (OFF Interval) # of prefixes / Time (minutes)
(Optional)

Document any configuration deltas, which are observed after the ISSU upgrade has taken effect. Note differences, which are driven by changes in the patch or release level as well as items, which are aberrant changes due to software faults. In either of these cases, any unexpected behavioral changes should be analyzed and a determination made as to the impact of the change (be it functional variances or operational impacts to existing scripts or management mechanisms).

7.1. Data collection considerations

When a DUT is undergoing an ISSU operation, it's worth noting that the DUT's data collection and reporting of data, such as counters, interface statistics, log messages, etc., might not be accurate. As such, one SHOULD NOT rely on the DUTs data collection methods, but rather, SHOULD use the test tools and equipment to collect data used for reporting in Section 7. Care and consideration should be paid in testing or adding new test cases, such that the desired data can be collected from the test tools themselves, or other external equipment, outside of the DUT itself.

8. Security Considerations

None at this time.

9. IANA Considerations

None at this time.

10. Conclusions

None at this time.

11. References

11.1. Normative References

- [1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [2] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

11.2. Informative References

- [3] Faber, T., Touch, J. and W. Yue, "The TIME-WAIT state in TCP and Its Effect on Busy Servers", Proc. Infocom 1999 pp. 1573-1583.
- [Fab1999] Faber, T., Touch, J. and W. Yue, "The TIME-WAIT state in TCP and Its Effect on Busy Servers", Proc. Infocom 1999 pp. 1573-1583.
- [RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

12. Acknowledgments

The authors wish to thanks Vibin Thomas for his valued review and feedback.

Copyright (c) 2014 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- o Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- o Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- o Neither the name of Internet Society, IETF or IETF Trust, nor the names of specific contributors, may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Authors' Addresses

Sarah Banks
VSS Monitoring
Email: sbanks@encrypted.net

Fernando Calabria
Cisco Systems
Email: fcalabri@cisco.com

Gery Czirjak
Juniper Networks
Email: gczirjak@juniper.net

Ramdas Machat
Juniper Networks
Email: rmachat@juniper.net

Internet-Draft
Network Working Group
Intended Status: Informational
Expires: March 2015

Bhuvaneswaran Vengainathan
Anton Basil
Veryx Technologies
Vishwas Manral
Ionos Corp
Mark Tassinari
Hewlett-Packard
September 26, 2014

Benchmarking Methodology for SDN Controller Performance
draft-bhuvan-bmwg-of-controller-benchmarking-01

Abstract

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

Status of this Memo

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

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

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

This Internet-Draft will expire on March 26, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Scope	4
4. Test Setup	5
4.1 SDN Network - Controller working in Standalone Mode	5
4.2 SDN Network - Controller working in Cluster Mode	5
4.3 SDN Network with TE - Controller working in Standalone Mode	6
4.4 SDN Network with TE - Controller working in Cluster Mode	6
4.5 SDN Node with TE - Controller working in Standalone Mode	7
4.6 SDN Node with TE - Controller working in Cluster Mode	8
5. Test Considerations	8
5.1 Network Topology	8
5.2 Test Traffic	8
5.3 Connection Setup	9
5.4 Measurement Accuracy	9
5.5 Real World Scenario	9
6. Test Reporting	9
7. Benchmarking Tests	10
7.1 Performance	10
7.1.1 Network Topology Discovery Time	10
7.1.2 Synchronous Message Processing Time	12
7.1.3 Synchronous Message Processing Rate	13
7.1.4 Path Provisioning Time	15
7.1.5 Path Provisioning Rate	17
7.1.6 Network Topology Change Detection Time	19
7.2 Scalability	21
7.2.1 Network Discovery Size	21
7.2.2 Flow Scalable Limit	22
7.3 Security	23
7.3.1 Exception Handling	23
7.3.2 Denial of Service Handling	24
7.4 Reliability	25
7.4.1 Controller Failover Time	25
7.4.2 Network Re-Provisioning Time	26
8. Test Coverage	28

9. References	28
9.1 Normative References	28
9.2 Informative References	29
10. IANA Considerations	29
11. Security Considerations	29
12. Acknowledgements	29
13. Authors' Addresses	30

1. Introduction

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

Conventions used in this document

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

2. Terminology

SDN Node:

An SDN node is a physical or virtual entity that forwards data in a software defined environment.

Flow:

A flow is a traffic stream having same source and destination address. The address could be MAC or IP or combination of both.

Learning Rate:

The rate at which the controller learns the new source addresses from the received traffic without dropping.

Controller Forwarding Table:

A controller forwarding table contains flow records for the flows configured in the data path.

Northbound Interface:

Northbound interface is the application programming interface provided by the SDN controller for communication with SDN services and applications.

Southbound Interface:

Southbound interface is the application programming interface provided by the SDN controller for communication with the SDN nodes.

Proactive Flow Provisioning:

Proactive flow provisioning is the pre-provisioning of flow entries into the controller's forwarding table through controller's northbound interface or management interface.

Reactive Flow Provisioning:

Reactive flow provisioning is the dynamic provisioning of flow entries into the controller's forwarding table based on traffic forwarded by the SDN nodes through controller's southbound interface.

Path:

A path is the route taken by a flow while traversing from a source node to destination node.

Standalone Mode:

Single controller handling all control plane functionalities.

Cluster/Redundancy Mode:

Group of controllers handling all control plane functionalities .

Synchronous Message:

Any message from the SDN node that triggers a response message from the controller e.g., Keepalive request and response message, flow setup request and response message etc.,

3. Scope

This document defines a number of tests to measure the networking aspects of SDN controllers. These tests are recommended for execution in lab environments rather than in real time deployments.

4. Test Setup

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

4.1 SDN Network - Controller working in Standalone Mode

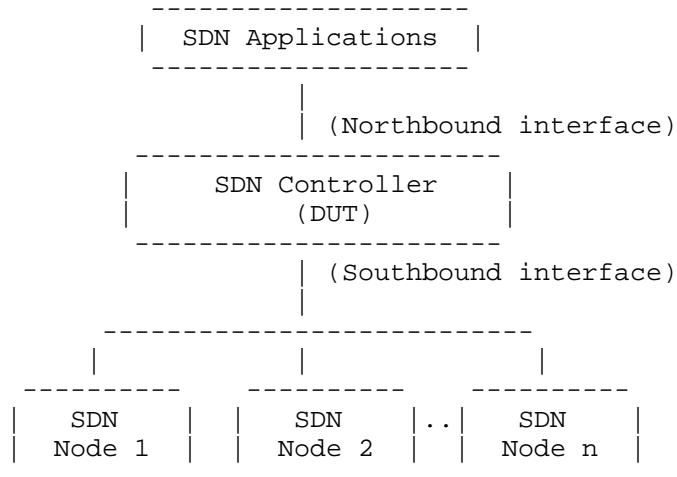


Figure 1

4.2 SDN Network - Controller working in Cluster Mode

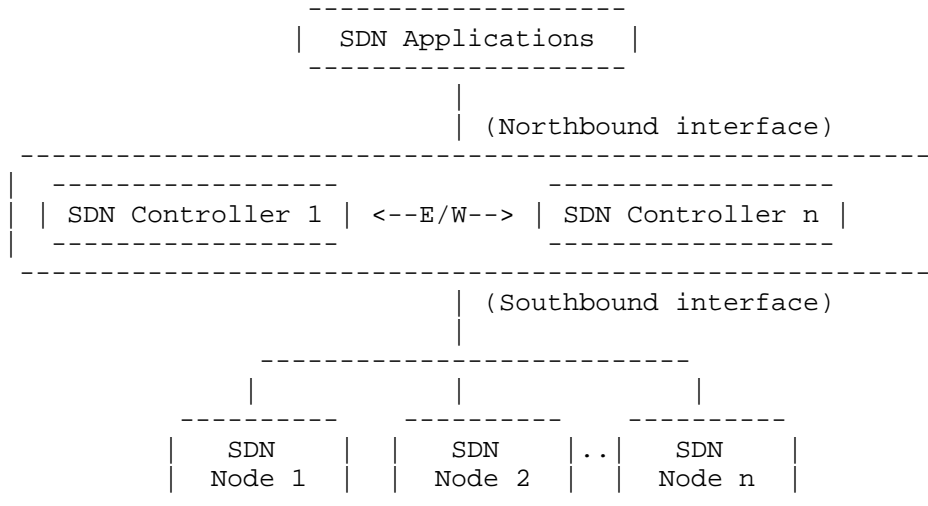


Figure 2

4.3 SDN Network with Traffic Endpoints (TE) - Controller working in Standalone Mode

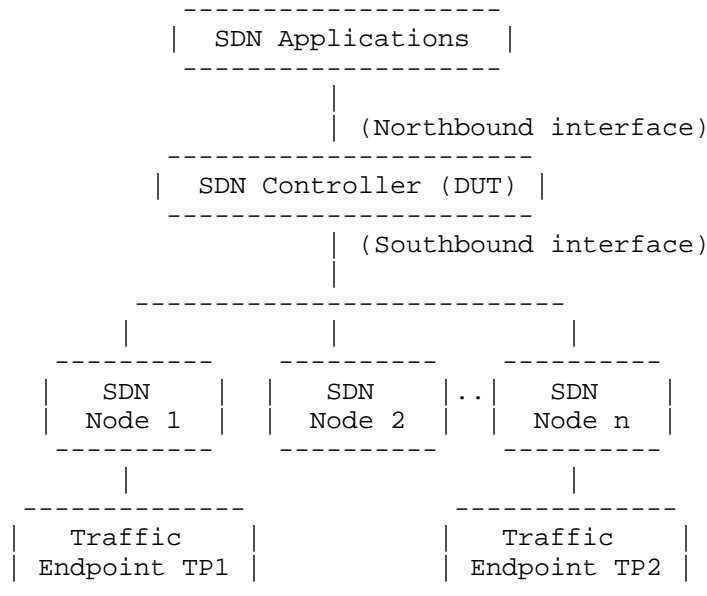
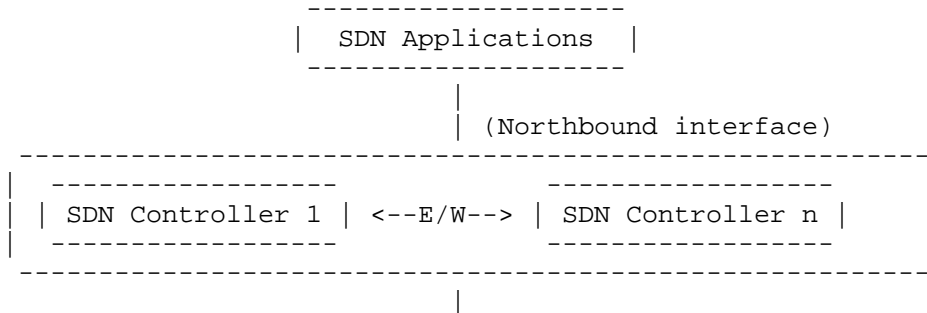


Figure 3

4.4 SDN Network with Traffic Endpoints (TE) - Controller working in Cluster Mode



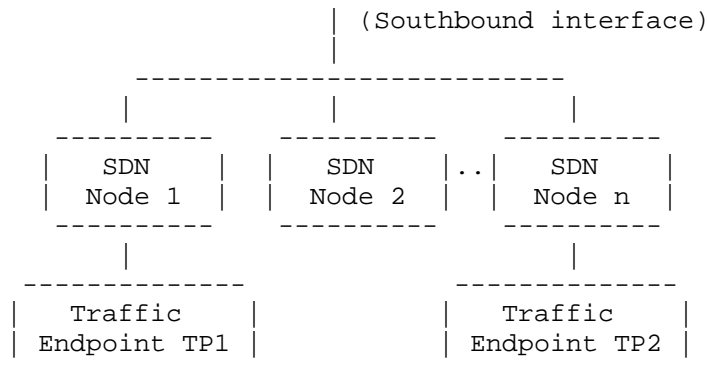


Figure 4

4.5 SDN Node with Traffic Endpoints (TE) - Controller working in Standalone Mode

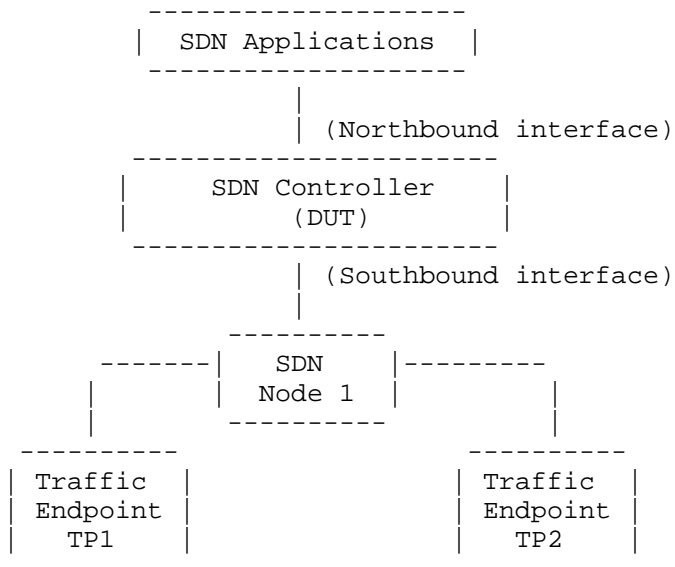


Figure 5

4.6 SDN Node with Traffic Endpoints (TE) - Controller working in Cluster Mode

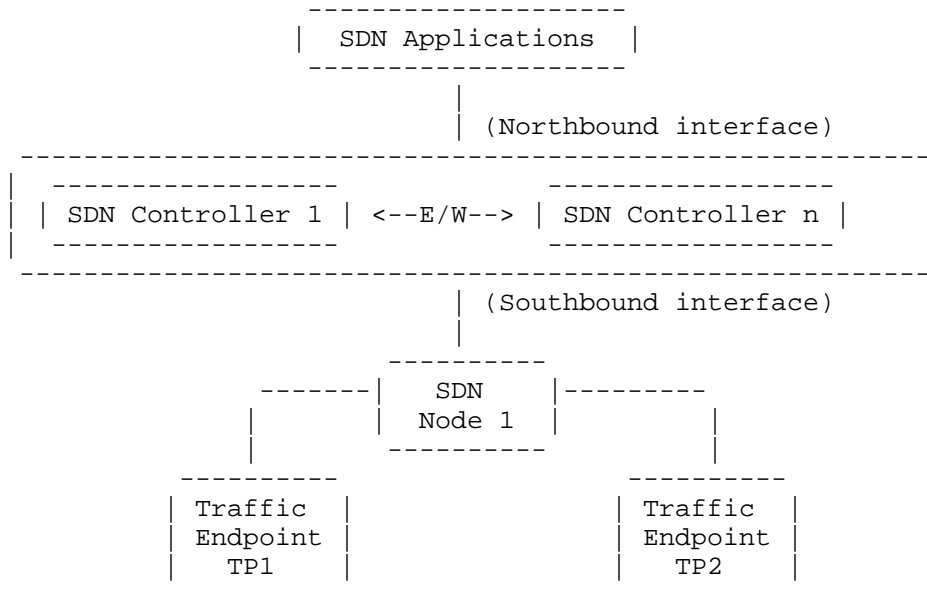


Figure 6

5. Test Considerations

5.1 Network Topology

The network SHOULD be deployed with SDN nodes interconnected in either fully meshed, tree or linear topology. Care should be taken to make sure that the loop prevention mechanism is enabled either in the SDN controller or in the network. To get complete performance characterization of SDN controller, it is recommended that the controller be benchmarked for many network topologies. These network topologies can be deployed using real hardware or emulated in hardware platforms.

5.2 Test Traffic

Test traffic can be used to notify the controller about the arrival of new flows or generate notifications/events towards controller. In either case, it is recommended that at least five different frame sizes and traffic types be used, depending on the intended network deployment.

5.3 Connection Setup

There may be controller implementations that support unencrypted and encrypted network connections with SDN nodes. Further, the controller may have backward compatibility with SDN nodes running older versions of southbound protocols. It is recommended that the controller performance be measured with the applicable connection setup methods.

1. Unencrypted connection with SDN nodes, running same protocol version.
2. Unencrypted connection with SDN nodes, running different (previous) protocol versions.
3. Encrypted connection with SDN nodes, running same protocol version
4. Encrypted connection with SDN nodes, running different (previous) protocol versions.

5.4 Measurement Accuracy

The measurement accuracy depends on the point of observation where the indications are captured. For example, the notification can be observed at the ingress or egress point of the SDN node. If it is observed at the egress point of the SDN node, the measurement includes the latency within the SDN node also. It is recommended to make observation at the ingress point of the SDN node unless it is explicitly mentioned otherwise in the individual test.

5.5 Real World Scenario

Benchmarking tests discussed in the document are to be performed on a "black-box" basis, relying solely on measurements observable external to the controller. The network deployed and the test parameters should be identical to the deployment scenario to obtain value added measures.

6. Test Reporting

Each test has a reporting format which is specific to individual test. In addition, the following configuration parameters SHOULD be reflected in the test report.

1. Controller name and version
2. Northbound protocols and version
3. Southbound protocols and version
4. Controller redundancy mode (Standalone or Cluster Mode)
5. Connection setup (Unencrypted or Encrypted)
6. Network Topology (Mesh or Tree or Linear)
7. SDN Node Type (Physical or Virtual or Emulated)
8. Number of Nodes
9. Number of Links
10. Test Traffic Type

7. Benchmarking Tests

7.1 Performance

7.1.1 Network Topology Discovery Time

Objective:

To measure the time taken to discover the network topology- nodes and its connectivity by a controller, expressed in milliseconds.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the number of nodes present in the defined network topology

Test setup parameters:

Test Iterations (Tr) - Defines the number of times the test needs to be repeated. The recommended value is 3.

Test Interval (To)- Defines the maximum time for the test to complete, expressed in milliseconds.

Test Setup:

The test can use one of the test setup described in section 4.1 and 4.2 of this document.

Prerequisite:

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

Procedure:

1. Initialize the controller - network applications, northbound and southbound interfaces.
2. Deploy the network with the given number of nodes using mesh or linear topology.
3. Initialize the network connections between controller and network nodes.
4. Record the time for the first discovery message exchange between the controller and the network node (Tm1).
5. Query the controller continuously for the discovered network topology information and compare it with the deployed network topology information.
6. Stop the test when the discovered topology information is matching with the deployed network topology or the expiry of test interval (To).

7. Record the time last discovery message exchange between the controller and the network node (Tmn) when the test completed successfully.

Note: While recording the Tmn value, it is recommended that the messages that are used for aliveness check or session management be ignored.

Measurement:

Topology Discovery Time $Tr1 = Tmn - Tm1$.

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

Note:

1. To increase the certainty of measured result, it is recommended that this test be performed several times with same number of nodes using same topology.
2. To get the full characterization of a controller's topology discovery functionality
 - a. Perform the test with varying number of nodes using same topology
 - b. Perform the test with same number of nodes using different topologies.

Reporting Format:

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

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

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

7.1.2 Synchronous Message Processing Time

Objective:

To measure the time taken by the controller to process a synchronous message, expressed in milliseconds.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the number of nodes present in the defined network topology

Test setup parameters:

Test Iterations (Tr) - Defines the number of times the test needs to be repeated. The recommended value is 3.

Test Duration (Td) - Defines the duration of test iteration, expressed in seconds. The recommended value is 5 seconds.

Test Setup:

The test can use one of the test setup described in section 4.1 and 4.2 of this document.

Prerequisite:

1. The controller should have completed the network topology discovery for the connected nodes.

Procedure:

1. Generate a synchronous message from every connected nodes one at a time and wait for the response before generating the next message.
2. Record total number of messages sent to the controller by all nodes (Ntx) and the responses received from the controller (Nrx) within the test duration (Td).

Measurement:

$$\text{Synchronous Message Processing Time } Tr1 = \frac{Td}{Nrx}$$

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

Note:

1. The above test measures the controller's message processing time at lower traffic rate. To measure the controller's message processing time at full connection rate, apply the same measurement equation with the Td and Nrx values obtained from Synchronous Message Processing Rate test (defined in Section 7.1.3).
2. To increase the certainty of measured result, it is recommended that this test be performed several times with same number of nodes using same topology.
3. To get the full characterization of a controller's synchronous message processing time
 - a. Perform the test with varying number of nodes using same topology
 - b. Perform the test with same number of nodes using different topologies.

Reporting Format:

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

The report should capture the following information in addition to the configuration parameters captured in section 6.
- Offered rate (Ntx)

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

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

7.1.3 Synchronous Message Processing Rate

Objective:

To measure the maximum number of synchronous messages (session aliveness check message, new flow arrival notification message etc.) a controller can process within the test duration, expressed in messages processed per second.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the number of nodes present in the defined network topology.

Test setup parameters:

Test Iterations (Tr) - Defines the number of times the test needs to be repeated. The recommended value is 3.

Test Duration (Td) - Defines the duration of test iteration, expressed in seconds. The recommended value is 5 seconds.

Test Setup:

The test can use one of the test setup described in section 4.1 and 4.2 of this document.

Prerequisite:

1. The controller should have completed the network topology discovery for the connected nodes.

Procedure:

1. Generate synchronous messages from all the connected nodes at the full connection capacity for the Test Duration (Td).
2. Record total number of messages sent to the controller by all nodes (Ntx) and the responses received from the controller (Nrx) within the test duration (Td).

Measurement:

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

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

Note:

1. To increase the certainty of measured result, it is recommended that this test be performed several times with same number of nodes using same topology.
2. To get the full characterization of a controller's synchronous message processing rate
 - a. Perform the test with varying number of nodes using same topology.
 - b. Perform the test with same number of nodes using different topologies.

Reporting Format:

The Synchronous Message Processing Rate results SHOULD be reported in the format of a table with a row for each iteration. The last row of the table indicates the average Synchronous Message Processing Rate.

The report should capture the following information in addition to the configuration parameters captured in section 6.

- Offered rate (Ntx)

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

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

7.1.4 Path Provisioning Time

Objective:

To measure the time taken by the controller to setup a path between source and destination node, expressed in milliseconds.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the number of nodes present in the defined network topology

Number of data path nodes (Ndp) - Defines the number of nodes present in the path between source and destination node.

Test setup parameters:

Test Iterations (Tr) - Defines the number of times the test needs to be repeated. The recommended value is 3.

Test Interval (To) - Defines the maximum time for the test to complete, expressed in milliseconds.

Test Setup:

The test can use one of the test setups described in section 4.3 and 4.4 of this document.

Prerequisite:

1. The controller should contain the network topology information for the deployed network topology.
2. The network topology information can be learnt through dynamic Topology Discovery Mechanism or static configuration.
3. The controller should have learnt about the location of source/destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.
4. The SDN Node should send all new flows to the controller when it receives.

Procedure:

Reactive Path Provisioning:

1. Send traffic with source as source endpoint address and destination as destination endpoint address from TP1.
2. Record the time for the first frame sent to the source SDN node (Tsf1).
3. Wait for the arrival of first frame from the destination node or the expiry of test interval (To).
4. Record the time when the first frame received from the destination SDN node (Tdf1).

Proactive Path Provisioning:

1. Send traffic with source as source endpoint address and destination as destination endpoint address from TP1.
2. Install the flow with the learnt source and destination address through controller's northbound or management interface.
3. Record the time when a successful response for the flow installation is received (Tp) from the controller.
4. Wait for the arrival of first frame from the destination node or the expiry of test interval (To).
5. Record the time when the first frame received from the destination node (Tdf1).

Measurement:

Reactive Path Provisioning:

Flow Provisioning Time Tr1 = Tdf1-Tsf1.

Proactive Path Provisioning:

Path Provisioning Time Tr1 = Tdf1-Tp.

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

Note:

1. To increase the certainty of measured result, it is recommended that this test be performed several times with same number of nodes using same topology.
2. To get the full characterization of a controller's path provisioning time
 - a. Perform the test with varying number of nodes using same topology
 - b. Perform the test with same number of nodes using different topologies.

Reporting Format:

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

The report should capture the following information in addition to the configuration parameters captured in section 6.

- Number of data path nodes

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

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

7.1.5 Path Provisioning Rate

Objective:

To measure the maximum number of paths a controller can setup between sources and destination node within the test duration, expressed in paths per second.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the number of nodes present in the defined network topology.

Test setup parameters:

Test Iterations (Tr) - Defines the number of times the test needs to be repeated. The recommended value is 3.

Test Duration (Td) - Defines the duration of test iteration, expressed in seconds. The recommended value is 5 seconds.

Test Setup:

The test can use one of the test setup described in section 4.3 and 4.4 of this document.

Prerequisite:

1. The controller should contain the network topology information for the deployed network topology.
2. The network topology information can be learnt through dynamic Topology Discovery Mechanism or static configuration.
3. The controller should have learnt about the location of source/destination endpoints for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.
4. The SDN Node should send all new flows to the controller when it receives.

Procedure:

Reactive Path Provisioning:

1. Send traffic at the individual node's synchronous message processing rate with unique source and/or destination addresses from test port TP1.
2. Record total number of unique frames received by the destination node (Ndf) within the test duration (Td).

Proactive Path Provisioning:

1. Send traffic continuously with unique source and destination addresses from the source node.
2. Install flows with the learnt source and destination addresses through controller's northbound or management interface.
3. Record total number of unique frames received from the destination node (Ndf) within the test duration (Td).

Measurement:

Proactive/Reactive Path Provisioning:

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

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

Note:

1. To increase the certainty of measured result, it is recommended that this test be performed several times with same number of nodes using same topology.
2. To get the full characterization of a controller's path provisioning rate
 - a. Perform the test with varying number of nodes using same topology
 - b. Perform the test with same number of nodes using different topologies.

Reporting Format:

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

The report should capture the following information in addition to the configuration parameters captured in section 6.

- Number of Nodes in the path
- Provisioning Type (Proactive/Reactive)
- Offered rate

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

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

7.1.6 Network Topology Change Detection Time

Objective:

To measure the time taken by the controller to detect any changes in the network topology, expressed in milliseconds.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the number of nodes present in the defined network topology

Test setup parameters:

Test Iterations (Tr) - Defines the number of times the test needs to be repeated. The recommended value is 3.

Test Interval (To) - Defines the maximum time for the test to complete, expressed in milliseconds. Test not completed within this time interval is considered as incomplete.

Test Setup:

The test can use one of the test setup described in section 4.1 and 4.2 of this document.

Prerequisite:

1. The controller should have discovered the network topology information for the deployed network topology.
2. The periodic network discovery operation should be configured to twice the Test Interval (To) value.

Procedure:

1. Trigger a topology change event through one of the operation (e.g., Add a new node or bring down an existing node or a link).
2. Record the time when the first topology change notification is sent to the controller (Tcn).
3. Stop the test when the controller sends the first topology re-discovery message to the SDN node or the expiry of test interval (To).
4. Record the time when the first topology re-discovery message is received from the controller (Tcd).

Measurement:

Network Topology Change Detection Time $Tr1 = Tcd - Tcn$.

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

Note:

1. To increase the certainty of measured result, it is recommended that this test be performed several times with same number of nodes using same topology.

Reporting Format:

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

7.2 Scalability

7.2.1 Network Discovery Size

Objective:

To measure the network size (number of nodes) that a controller can discover within a stipulated time.

Setup Parameters:

The following parameters MUST be defined:

Network setup parameters:

Number of nodes (N) - Defines the initial number of nodes present in the defined network topology

Test setup parameters:

Network Discovery Time (Tnd) - Defines the stipulated time acceptable by the user, expressed in seconds.

Test Setup:

The test can use one of the test setup described in section 4.1 and 4.2 of this document.

Prerequisite:

1. The controller should support automatic network discovery.
2. Tester should be able to retrieve the discovered topology information either through controller's management interface or northbound interface.
3. Controller should be operational.
4. Network with the given number of nodes and intended topology (Mesh or Linear or Tree) should be deployed.

Procedure:

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

Measurement:

Network Discovery Size = Ns.

Note:

This test may be performed with different topologies to obtain the controller's scalability factor for various network topologies.

Reporting Format:

The Network Discovery Size results SHOULD be reported in addition to the configuration parameters captured in section 6.

7.2.2 Flow Scalable Limit

Objective:

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

Setup Parameters:

The following parameters MUST be defined:

Test Setup:

The test can use one of the test setups described in section 4.5 and 4.6 of this document.

Prerequisite:

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

Procedure:

Reactive Path Provisioning:

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

Proactive Path Provisioning:

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

Note:

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

Measurement:

Proactive Path Provisioning:

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

Reactive Path Provisioning:

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

Flow Scalable Limit = Max Flow Entries.

Reporting Format:

The Flow Scalable Limit results SHOULD be tabulated with the following information in addition to the configuration parameters captured in section 6.

- Provisioning Type (Proactive/Reactive)

7.3 Security

7.3.1 Exception Handling

Objective:

To determine the effect of handling error packets and notifications on performance tests. The impact SHOULD be measured for the following performance tests

- a. Path Programming Rate
- b. Path Programming Time
- c. Network Topology Change Detection Time

Prerequisite:

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

Procedure:

1. Perform the above listed performance tests and send 1% of messages from the Synchronous Message Processing Rate as invalid messages from the connected nodes.
2. Perform the above listed performance tests and send 2% of messages from the Synchronous Message Processing Rate as invalid messages from the connected nodes.

Note:

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

Measurement:

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

Reporting Format:

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

- Without Exceptions
- With 1% Exceptions
- With 2% Exceptions

7.3.2 Denial of Service Handling

Objective:

To determine the effect of handling DoS attacks on performance and scalability tests The impact SHOULD be measured for the following tests

- a. Path Programming Rate
- b. Path Programming Time
- c. Network Topology Change Detection Time
- d. Network Discovery Size

Prerequisite:

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

Procedure:

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

Note:

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

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

Measurement:

Measurement should be done as per the equation defined in the corresponding test's measurement section.

Reporting Format:

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

- Without any attacks
- With attacks

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

7.4 Reliability

7.4.1 Controller Failover Time

Objective:

To compute the time taken to switch from one controller to another when the controllers are teamed and the active controller fails.

Setup Parameters:

The following parameters MUST be defined:

Controller setup parameters:

Number of cluster nodes (CN) - Defines the number of member nodes present in the cluster.

Redundancy Mode (RM) - Defines the controller clustering mode e.g., Active - Standby or Active - Active.

Test Setup:

The test can use the test setup described in section 4.4 of this document.

Prerequisite:

1. Master controller election should be completed.
2. Nodes are connected to the controller cluster as per the Redundancy Mode (RM).
3. The controller cluster should have completed the network topology discovery.
4. The SDN Node should send all new flows to the controller when it receives.

Procedure:

1. Send bi-directional traffic continuously with unique source and/or destination addresses from test ports TP1 and TP2 at the rate that the controller processes without any drops.
2. Bring down the active controller.
3. Stop the test when a first frame received on TP2 after failover operation.
4. Record the test duration (Td), total number of frames sent (Nsnt) on TP1 and number of frames received (Nrvd) on TP2.

Measurement:

Controller Failover Time = $((Td/Nrvd) - (Td/Nsnt))$
Packet Loss = $Nsnt - Nrvd$

Reporting Format:

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

- Number of cluster nodes
- Redundancy mode
- Controller Failover
- Time Packet Loss

7.4.2 Network Re-Provisioning Time

Objective:

To compute the time taken to re-route the traffic by the controller when there is a failure in existing traffic paths.

Setup Parameters:

Same setup parameters as defined in the Path Programming Rate performance test (Section 7.1.5).

Prerequisite:

Network with the given number of nodes and intended topology (Mesh or Tree) with redundant paths should be deployed.

Procedure:

1. Perform the test procedure mentioned in Path Programming Rate test (Section 7.1.5).
2. Send bi-directional traffic continuously with unique sequence number for one particular traffic endpoint.
3. Bring down a link or switch in the traffic path.
4. Stop the test after receiving first frame after network re-convergence (timeline).
5. Record the time of last received frame prior to the frame loss at TP2 (TP2-Tlfr) and the time of first frame received after the frame loss at TP2 (TP2-Tffr).
6. Record the time of last received frame prior to the frame loss at TP1 (TP1-Tlfr) and the time of first frame received after the frame loss at TP1 (TP1-Tffr).

Measurement:

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

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

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

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

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

Reporting Format:

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

- Number of nodes in the primary path
- Number of nodes in the alternate path
- Network Re-Provisioning Time
- Forward Direction Packet Loss
- Reverse Direction Packet Loss

8. Test Coverage

	Performance	Scalability	Reliability
Setup	1. Network Topology Discovery 2. Path Provisioning Time 3. Path Provisioning Rate	1. Network Discovery Size	
Operational	1. Synchronous Message Processing Rate 2. Synchronous Message Processing Time	1. Flow Scalable Limit	1. Network Topology Change Detection Time 2. Exception Handling 3. Denial of Service Handling 4. Network Re-Provisioning Time
Tear Down			1. Controller Failover Time

9. References

9.1 Normative References

- [RFC6241] R. Enns, M. Bjorklund, J. Schoenwaelder, A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.
- [RFC6020] M. Bjorklund, "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010

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

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

[I-D.i2rs-architecture] A. Atlas, J. Halpern, S. Hares, D. Ward, T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-05 (Work in progress), July 20, 2014.

9.2 Informative References

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

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

10. IANA Considerations

This document does not have any IANA requests.

11. Security Considerations

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

12. Acknowledgements

The authors would like to acknowledge the following individuals for their help and participation of the compilation of this document: Al Morton (AT&T), Brain Castelli (Spirent), Sandeep Gangadharan(HP), Sarah Banks (VSS Monitoring) who made significant suggestions to the current and earlier versions of this document.

13. Authors' Addresses

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

Email: bhuvaneswaran.vengainathan@veryxtech.com

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

Email: anton.basil@veryxtech.com

Vishwas Manral
Ionos Corp,
4100 Moorpark Ave,
San Jose, CA

Email: vishwas@ionosnetworks.com

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

Email: mark.tassinari@hp.com

Internet Engineering Task Force
Internet-Draft, Intended status: Informational
Expires April 2015
October 17, 2014

L. Avramov
Cisco Systems
J. Rapp
Hewlett-Packard

Data Center Benchmarking Methodology
draft-bmwg-dcbench-methodology-03

Abstract

The purpose of this informational document is to establish test and evaluation methodology and measurement techniques for physical network equipment in the data center.

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

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

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document MUST include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	5
1.2. Methodology format and repeatability recommendation	5
2. Line Rate Testing	5
2.1 Objective	5
2.2 Methodology	5
2.3 Reporting Format	6
3. Buffering Testing	7
3.1 Objective	7
3.2 Methodology	7
3.3 Reporting format	10
4. Microburst Testing	10
4.1 Objective	10
4.2 Methodology	10
4.3 Reporting Format	11
5. Head of Line Blocking	11
5.1 Objective	11
5.2 Methodology	11
5.3 Reporting Format	13
6. Incast Stateful and Stateless Traffic	13
6.1 Objective	13
6.2 Methodology	13
6.3 Reporting Format	14
7. References	15
7.1. Normative References	16
7.2. Informative References	16
7.3. URL References	16
Authors' Addresses	16

1. Introduction

Traffic patterns in the data center are not uniform and are constantly changing. They are dictated by the nature and variety of applications utilized in the data center. It can be largely east-west traffic flows in one data center and north-south in another, while some may combine both. Traffic patterns can be bursty in nature and contain many-to-one, many-to-many, or one-to-many flows. Each flow may also be small and latency sensitive or large and throughput sensitive while containing a mix of UDP and TCP traffic. All of which can coexist in a single cluster and flow through a single network device all at the same time. Benchmarking of network devices have long used RFC1242, RFC2432, RFC2544, RFC2889 and RFC3918. These benchmarks have largely been focused around various latency attributes and max throughput of the Device Under Test [DUT] being

benchmarked. These standards are good at measuring theoretical max throughput, forwarding rates and latency under testing conditions however, they do not represent real traffic patterns that may affect these networking devices.

The following provides a methodology for benchmarking Data Center DUT including congestion scenarios, switch buffer analysis, microburst, head of line blocking, while also using a wide mix of traffic conditions.

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 [6].

1.2. Methodology format and repeatability recommendation

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

-Objective

-Methodology

-Reporting Format

MUST: minimum test for the scenario described

SHOULD: recommended test for the scenario described

MAY: ideal test for the scenario described

For each test methodology described, it is key to obtain repeatability of the results. The recommendation is to perform enough iterations of the given test to make sure the result is accurate, this is especially important for section 3) as the buffering testing has been historically the least reliable.

2. Line Rate Testing

2.1 Objective

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

2.2 Methodology

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

For all tests, the percentage of traffic per port capacity sent MUST be 99.98% at most, with no PPM adjustment to ensure stressing the DUT

in worst case conditions. Tests results at a lower rate MAY be provided for better understanding of performance increase in terms of latency and jitter when the rate is lower than 99.98%. The receiving rate of the traffic needs to be captured during this test in % of line rate.

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

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

Alternatively when a traffic generator CAN NOT be connected to all ports on the DUT, a snake test MUST be used for line rate testing, excluding latency and jitter as those became then irrelevant. The snake test consists in the following method: -connect the first and last port of the DUT to a traffic generator-connect back to back sequentially all the ports in between: port 2 to 3, port 4 to 5 etc to port n-2 to port n-1; where n is the total number of ports of the DUT-configure port 1 and 2 in the same vlan X, port 3 and 4 in the same vlan Y, etc. port n-1 and port n in the same vlan ZZZ. This snake test provides a capability to test line rate for Layer 2 and Layer 3 RFC 2544/3918 in instance where a traffic generator with only two ports is available. The latency and jitter are not to be considered with this test.

2.3 Reporting Format

The report MUST include:

- physical layer calibration information as defined into (Placeholder for definitions draft)

- number of ports used

- reading for throughput received in percentage of bandwidth, while sending 99.98% of port capacity on each port, across packet size from 64 byte all the way to 9216. As guidance, an increment of 64 byte packet size between each iteration being ideal, a 256 byte and 512 bytes being also often time used, the most common packets sizes order for the report is: 64b,128b,256b,512b,1024b,1518b,4096,8000,9216b.

The pattern for testing can be expressed using RFC 6985 [IMIX Genome: Specification of Variable Packet Sizes for Additional Testing]

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

-for packet drops, they MUST be expressed in packet count value and SHOULD be expressed in % of line rate

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

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

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

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

3. Buffering Testing

3.1 Objective

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

3.2 Methodology

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

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

represents the maximum port buffer size at the measured inflexion point.

1) Measure the highest buffer efficiency

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

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

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

When the B value is found to provide the highest buffer size, this is the highest buffer efficiency

2) Measure maximum port buffer size

At fixed packet size B determined in 3.2.1, for a fixed default COS value of 0 and for unicast traffic proceed with the following:

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

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

Last iteration: ingress port N-2 sending line rate traffic to egress port N-1, while port N sending a known low amount of over

subscription traffic (1% recommended) with same packet size to the egress port N Measure the buffer size value by multiplying the number of extra frames sent by the frame size.

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

3) Measure maximum port pair buffer sizes

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

Second iteration: ingress port 1 sending line rate to egress port 2; ingress port 3 sending line rate to egress port 4 etc. Ingress port N-1 and N will respectively over subscribe at 1% of line rate egress port 4 and port 5. Measure the buffer size value by multiplying the number of extra frames sent by the frame size for each egress port.

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

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

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

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

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

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

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

Unicast traffic and then Multicast traffic SHOULD be used in order to determine the proportion of buffer for documented selection of tests.

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

3.3 Reporting format

The report MUST include:

- The packet size used for the most efficient buffer used, along with COS value
- The maximum port buffer size for each port
- The maximum DUT buffer size
- The packet size used in the test
- The amount of over subscription if different than 1%
- The number of ingress and egress ports along with their location on the DUT.

4 Microburst Testing

4.1 Objective

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

4.2 Methodology

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

The burst MUST be measure with an intensity of 100%, meaning the burst of packets will be sent with a minimum inter-packet gap. The amount of packet contained in the burst will be variable and increase until there is a non-zero packet loss measured. The aggregate amount of packets from all the senders will be used to calculate the maximum amount of microburst the DUT can sustain.

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

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

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

An example would be:

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

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

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

4.3 Reporting Format

The report MUST include:

- The maximum value of packets received per ingress port with the maximum burst size obtained with zero packet loss
- The packet size used in the test
- The number of ingress and egress ports along with their location on the DUT

5. Head of Line Blocking

5.1 Objective

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

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

5.2 Methodology

In order to cause congestion, head of line blocking, groups of four ports are used. A group has 2 ingress and 2 egress ports. The first

ingress port MUST have two flows configured each going to a different egress port. The second ingress port will congest the second egress port by sending line rate. The goal is to measure if there is loss for the first egress port which is not not oversubscribed.

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

1) Measure two groups with eight DUT ports

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

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

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

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

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

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

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

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

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

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

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

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

5.3 Reporting Format

For each test the report MUST include:

- The port configuration including the number and location of ingress and egress ports located on the DUT
- If HOLB was observed
- Percent of traffic loss

6. Incast Stateful and Stateless Traffic

6.1 Objective

The objective of this test is to measure the effect of TCP Goodput and latency with a mix of large and small flows. The test is designed to simulate a mixed environment of stateful flows that require high rates of goodput and stateless flows that require low latency.

6.2 Methodology

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

One ingress port MUST be maintaining a TCP connection through the ingress port to a receiver connected to an egress port. Traffic in the TCP stream MUST be sent at the maximum rate allowed by the traffic generator. At the same time the TCP traffic is flowing

through the DUT the stateless traffic is sent destined to a receiver on the same egress port. The stateless traffic MUST be a microburst of 100% intensity.

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

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

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

For example:

Stateful Traffic port variation:

During Iterations number of Egress ports MAY vary as well.

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

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

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

Stateless Traffic port variation:

During Iterations number of Egress ports MAY vary as well. First Iteration: 1 Ingress port receiving stateful TCP traffic and 1 Ingress port receiving stateless traffic destined to 1 Egress Ports

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

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

6.3 Reporting Format

The report MUST include the following:

- Number of ingress and egress ports along with designation of stateful or stateless.
- Stateful goodput

- Stateless latency

7. References

7.1. Normative References

- [1] Bradner, S. "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, July 1991.
- [2] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.

7.2. Informative References

- [3] Mandeville R. and Perser J., "Benchmarking Methodology for LAN Switching Devices", RFC 2889, August 2000.
- [4] Stopp D. and Hickman B., "Methodology for IP Multicast Benchmarking", BCP 26, RFC 3918, October 2004.

7.3. URL References

- [5] Yanpei Chen, Rean Griffith, Junda Liu, Randy H. Katz, Anthony D. Joseph, "Understanding TCP Incast Throughput Collapse in Datacenter Networks",
<http://www.eecs.berkeley.edu/~ychen2/professional/TCPIncastWREN2009.pdf>
".

Authors' Addresses

Lucien Avramov
Cisco Systems
170 West Tasman drive
San Jose, CA 95134
United States
Phone: +1 408 526 7686
Email: lavramov@cisco.com

Jacob Rapp
Hewlett-Packard
3000 Hanover Street
Palo Alto, CA
United States
Phone: +1 650 857 3367
Email: jacob.h.rapp@hp.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 5, 2015

W. Cervený
Arbor Networks
R. Bonica
Juniper Networks
July 4, 2014

Benchmarking Neighbor Discovery
draft-cervený-bmwg-ipv6-nd-05

Abstract

This document is a benchmarking instantiation of RFC 6583: "Operational Neighbor Discovery Problems" [RFC6583]. It describes a general testing procedure and measurements that can be performed to evaluate how the problems described in RFC 6583 may impact the functionality or performance of intermediate nodes.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 5, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Overview of Relevant NDP and Intermediate Node Behavior . . .	3
4. Test Setup	5
4.1. Testing Interfaces	6
5. Modifiers (Variables)	6
5.1. Frequency of NDP Triggering Packets	6
6. Tests	7
6.1. Stale Entry Time Determination	7
6.1.1. General Testing Procedure	7
6.2. Neighbor Cache Exhaustion Determination	7
6.2.1. General Testing Procedure	8
6.3. Determine Neighbor Discovery Behavior During Address Scan	8
6.3.1. General Testing Procedure	8
6.4. Pre-established Flow Treatment	8
6.4.1. General Testing Procedure	8
6.5. Stopped Flow Recovery Behavior	9
6.5.1. General Testing Procedure	9
7. Measurements Explicitly Excluded	9
7.1. DUT CPU Utilization	9
7.2. Malformed Packets	9
8. DUT Initialization	9
9. IANA Considerations	10
10. Security Considerations	10
11. Acknowledgements	10
12. References	10
12.1. Normative References	10
12.2. Informative References	11
Authors' Addresses	11

1. Introduction

This document is a benchmarking instantiation of RFC 6583: "Operational Neighbor Discovery Problems" [RFC6583]. It describes a general testing procedure and measurements that can be performed to evaluate how the problems described in RFC 6583 may impact the functionality or performance of intermediate nodes.

2. Terminology

Intermediate Node A router, switch, firewall or any other device which separates end-nodes. The tests in this document can be completed with any intermediate node which maintains a neighbor cache, although not all measurements and performance characteristics may apply.

Neighbor Cache The neighbor cache is a database which correlates the link-layer address and the adjacent interface with an IPv6 address.

Neighbor Discovery See Section 1 of RFC 4861 [RFC4861]

Scanner Network The network from which the scanning tested is connected.

Scanning Interface The interface from which the scanning activity is conducted.

Stale Entry Time This is the duration for which a neighbor cache entry marked "Reachable" will continue to be marked "Reachable" if an update for the address is not received.

Target Network The network for which the scanning tests is targeted.

Target Network Destination Interface The interface that resides on the target network, which is primarily used to measure DUT performance while the scanning activity is occurring.

3. Overview of Relevant NDP and Intermediate Node Behavior

In a traditional network, an intermediate node must support a mapping between a connected node's IP address and the connected node's link-layer address and interface the node is connected to. With IPv4, this process is handled by ARP [RFC0826]. With IPv6, this process is handled by NDP and is documented in [RFC4861]. With IPv6, when a packet arrives on one of an intermediate node's interfaces and the destination address is determined to be reachable via an adjacent network:

1. The intermediate node first determines if the destination IPv6 address is present in its neighbor cache.
2. If the address is present in the neighbor cache, the intermediate node forwards the packet to the destination node using the appropriate link-layer address and interface.

3. If the destination IPv6 address is not in the intermediate node's neighbor cache:
 1. An entry for the IPv6 address is added to the neighbor cache and the entry is marked "INCOMPLETE".
 2. The intermediate node sends a neighbor solicitation packet to the solicited-node multicast address on the interface considered on-link.
 3. If a solicited neighbor advertisement for the IPv6 address is received by the intermediate node, the neighbor cache entry is marked "REACHABLE" and remains in this state for 30 seconds.
 4. If a neighbor advertisement is not received, the intermediate node will continue sending neighbor solicitation packets every second until either a neighbor solicitation is received or the maximum number of solicitations has been sent. If a neighbor advertisement is not received in this period, the entry can be discarded.

There are two scenarios where a neighbor cache can grow to a very large size:

1. There are a large number of real nodes connected via an intermediate node's interface and a large number of these nodes are sending and receiving traffic simultaneously.
2. There are a large number of addresses for which a scanning activity is occurring and no real node will respond to the neighbor solicitation. This scanning activity can be unintentional or malicious. In addition to maintaining the "INCOMPLETE" neighbor cache entry, the intermediate node must send a neighbor solicitation packet every second for the maximum number of solicitations. With today's network link bandwidths, a scanning event could cause a lot of entries to be added to the neighbor cache and solicited for in the time that it takes for a neighbor cache entry to be discarded.

An intermediate node's neighbor cache is of a finite size and can only accommodate a specific number of entries, which can be limited by available memory or a preset operating system limit. If the maximum number of entries in a neighbor cache is reached, the intermediate node must either drop an existing entry to make space for the new entry or deny the new IP address to MAC address/interface mapping with an entry in the neighbor cache. In an extreme

case, the intermediate node's memory may become exhausted, causing the intermediate node to crash or begin paging memory.

At the core of the neighbor discovery problems presented in RFC 6583 [RFC6583], unintentional or malicious IPv6 traffic can transit the intermediate node that resembles an IP address scan similar to an IPv4-based network scan. Unlike IPv4 networks, an IPv6 end network is typically configured with a /64 address block, allowing for upwards of 2^{64} addresses. When a network node attempts to scan all the addresses in a /64 address block directly attached to the intermediate node, it is possible to create a huge amount of state in the intermediate node's neighbor cache, which may stress processing or memory resources.

Section 7.1 of RFC 6583 recommends how intermediate nodes should behave when the neighbor cache is exceeded. Section 6 of RFC 6583 [RFC6583] recommends how damage from an IPv6 address scan may be mitigated. Section 6.2 of RFC 6583 [RFC6583] discusses queue tuning.

4. Test Setup

The network needs to minimally have two subnets: one from which the scanner(s) source their scanning activity and the other which is the target network of the address scans.

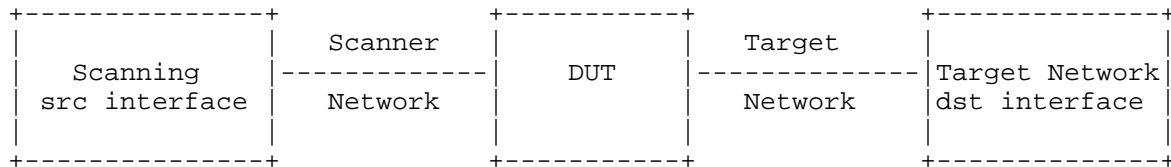
It is assumed that the latency for all network segments is negligible. By default, the target network's subnet shall be 64-bits in length, although some tests may involve increasing the prefix length.

Although packet size shouldn't have a direct impact, packet per second (pps) rates will have an impact. Smaller packet sizes should be utilized to facilitate higher packet per second rates.

For purposes of this test, the packet type being sent by the scanning device isn't important, although most scanning applications might want to send packets that would elicit responses from nodes within a subnet (such as an ICMPv6 echo request). Since it is not intended that responses be evoked from the target network node, such packets aren't necessary.

At the beginning of each test the intermediate node should be initialized. Minimally, the neighbor cache should be cleared.

Basic format of test network. Note that optional "non-participating network" is a third network not related to the scanner or target network.



4.1. Testing Interfaces

Two tester interfaces are configured for most tests:

- o Scanning source (src) interface: This is the interface from which test packets are sourced. This interface sources traffic to destination IPv6 addresses on the target network from a single link-local address, similar to how an adjacent intermediate node would transit traffic through the intermediate node.
- o Target network destination (dst) interface: This interface responds to neighbor solicitations as appropriate and confirms when an intermediate node has forwarded a packet to the interface for consumption. Where appropriate, the target network destination interface will respond to neighbor solicitations with a unique link-layer address per IPv6 address solicited.

5. Modifiers (Variables)

5.1. Frequency of NDP Triggering Packets

The frequency of NDP triggering packets could be as high as the maximum packet per second rate that the scanner network will support (or is rated for). However, it may not be necessary to send packets at a particularly high rate. In fact a goal of testing could be to identify if the DUT is able to withstand scans at rates which otherwise would not impact the performance of the DUT.

Optimistically, the scanning rate should be incremented until the DUT's performance begins deteriorating. Depending on the software and system being used to implement the scanning, it may be challenging to achieve a sufficient rate. Where this maximum threshold cannot be determined, the test results should note the highest rate tested and that DUT performance deterioration was not noticed at this rate.

The lowest rate tested should be the rate for which packets can be expected to have an impact on the DUT -- this value is of course, subjective.

6. Tests

6.1. Stale Entry Time Determination

This test determines the time interval when the intermediate node (DUT) identifies an address as stale.

RFC 4861, section 6.3.2 [RFC4861] states that an address can be marked "stale" at a random value between 15 and 45 seconds (as defined via constants in the RFC). This test confirms what value is being used by the intermediate node. Note that RFC 4861 states that this random time can be changed "at least every few hours."

6.1.1. General Testing Procedure

1. Send a packet from the scanning source interface to an address in target network. Observe that the intermediate node sends a neighbor solicitation to the solicited-node multicast address on the target network, for which tester destination interface should respond with a neighbor advertisement. The intermediate node should create an entry in neighbor cache for the address, marking the address as "reachable". As this point, the packet should be forwarded to the tester destination interface.
2. Wait 15 seconds.
3. Send a packet from tester source address to tester destination address. Determine if intermediate node sends neighbor solicitation. If intermediate node does send neighbor solicitation, the stale entry time has not been exceeded.
4. If a neighbor solicitation was not sent after one second, wait 2 seconds. If neighbor solicitation was not received, increment the wait time by one second and repeat this process until the intermediate node sends a neighbor solicitation for the address. The stale entry time is the number of seconds that has elapsed between the first packet and when the neighbor solicitation was sent.

6.2. Neighbor Cache Exhaustion Determination

Discover the point at which the neighbor cache is exhausted and evaluate intermediate node behavior when this threshold is reached.

6.2.1. General Testing Procedure

1. Send packets incrementally to unique addresses in the target network, simultaneously resending packets of previously discovered addresses within the stale entry time.
2. Observe what happens when one address greater than the maximum neighbor cache size ("n") is reached. When "n+1" is reached, if either the first or most recent cache entry are dropped, this may be acceptable.
3. Confirm intermediate node doesn't crash when "n+1" is reached.

6.3. Determine Neighbor Discovery Behavior During Address Scan

This test is a prerequisite for later tests, for which it is confirmed how an intermediate node behaves in the presence of an address scan. If adding the flow after the address scan results in abnormal behavior, it will be difficult to evaluate correct behavior for later tests.

6.3.1. General Testing Procedure

1. Start sending $n/2$ (n determined in "Neighbor Cache Exhaustion" test) flows at a rate of one packet per second to valid addresses (valid addresses are defined as addresses for which the tester responds to neighbor solicitation).
2. Send $n/2 + 1$ flow and determine if intermediate node takes a long time to process NS/NA for valid addresses.

6.4. Pre-established Flow Treatment

This test expands on "Determine neighbor discovery behavior during address scan". This test confirms behavior described in RFC 6583 [RFC6583], where it is expected that in the presence of an address scan, flows for successfully cached addresses will continue to flow across the intermediate node.

6.4.1. General Testing Procedure

1. Start $n/2$ flows (one packet per second per flow) to valid addresses.
2. Start address scan to invalid addresses (addresses for which DUT does not receive a neighbor advertisement).

3. Determine if flows continue for existing, valid flows, without unexpected loss or delay.

6.5. Stopped Flow Recovery Behavior

This test determines how a stopped flow recovers from the stale state in the presence of an address scan. It confirms that the intermediate node continues to prefer addresses that had previously been added to the neighbor cache, even when the address is marked "stale" in the neighbor cache.

6.5.1. General Testing Procedure

1. Start $n/2$ flows (one packet per second per flow) to valid addresses.
2. Start address scan to invalid addresses (addresses without responding host).
3. Stop one flow to valid address.
4. Wait stale time period for address to be marked "stale" in intermediate node neighbor cache.
5. Restart stopped flow and confirm that address is marked "active" immediately (not stuck behind address scan).

7. Measurements Explicitly Excluded

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

7.1. DUT CPU Utilization

This measurement relies on the DUT to provide utilization information, which is subjective.

7.2. Malformed Packets

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

8. DUT Initialization

At the beginning of each test, the neighbor cache of the DUT should be initialized.

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

Benchmarking activities as described in this memo are limited to technology characterization using controlled stimuli in a laboratory environment, with dedicated address space and the constraints specified in the sections above.

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

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the DUT/SUT. Special capabilities SHOULD NOT exist in the DUT/SUT specifically for benchmarking purposes.

Any implications for network security arising from the DUT/SUT SHOULD be identical in the lab and in production networks.

11. Acknowledgements

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

12. References

12.1. Normative References

- [RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", STD 37, RFC 826, November 1982.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.

- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.
- [RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, May 2008.
- [RFC6583] Gashinsky, I., Jaeggli, J., and W. Kumari, "Operational Neighbor Discovery Problems", RFC 6583, March 2012.

12.2. Informative References

- [RFC7048] Nordmark, E. and I. Gashinsky, "Neighbor Unreachability Detection Is Too Impatient", RFC 7048, January 2014.

Authors' Addresses

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

Email: wcerveney@arbor.net

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

Email: rbonica@juniper.net

Internet Engineering Task Force
Internet-Draft, Intended status: Informational
Expires: April 20, 2015
October 17, 2014

L. Avramov
Cisco Systems
J. Rapp
Hewlett-Packard

Data Center Benchmarking Definitions and Metrics
draft-dcbench-def-02

Abstract

The purpose of this informational document is to establish definitions, discussion and measurement techniques for data center benchmarking. Also, it is to introduce new terminologies applicable to data center performance evaluations. The purpose of this document is not to define the test methodology, but rather establish the important concepts when one is interested in benchmarking network equipment in the data center.

Status of this Memo

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

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

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

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

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

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these

documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	4
1.2. Definition format	4
2. Latency	4
2.1. Definition	4
2.2 Discussion	5
2.3 Measurement Units	5
3 Jitter	6
3.1 Definition	6
3.2 Discussion	6
3.3 Measurement Units	6
4 Physical Layer Calibration	6
4.1 Definition	6
4.2 Discussion	7
4.3 Measurement Units	7
5 Line rate	7
5.1 Definition	7
5.2 Discussion	8
5.3 Measurement Units	9
6 Buffering	10
6.1 Buffer	10
6.1.1 Definition	10
6.1.3 Discussion	11
6.1.3 Measurement Units	11
6.2 Incast	12
6.2.1 Definition	12
6.2.2 Discussion	12
6.2.3 Measurement Units	13
7 Application Throughput: Data Center Goodput	13
7.1. Definition	13
7.2. Discussion	13
7.3. Measurement Units	13
8. References	14
3.1. Normative References	14
3.2. Informative References	14
3.3. URL References	14
3.4. Acknowledgments	15
Authors' Addresses	15

1. Introduction

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

The following defines a set of definitions, metrics and terminologies including congestion scenarios, switch buffer analysis and redefines basic definitions in order to represent a wide mix of traffic conditions.

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 [6].

1.2. Definition format

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

Definition: The specific definition for the term.

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

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

2. Latency

2.1. Definition

Latency is a the amount of time it takes a frame to transit the DUT.

The Latency interval can be assessed between different combinations of events, irrespectively of the type of switching device (bit forwarding aka cut-through or store forward type of device)

Traditionally the latency measurement definitions are:

FILO (First In Last Out) The time interval starting when the end of the first bit of the input frame reaches the input port and ending when the last bit of the output frame is seen on the output port

FIFO (First In First Out) The time interval starting when the end of the first bit of the input frame reaches the input port and ending when the start of the first bit of the output frame is seen on the output port

LILO (Last In Last Out) The time interval starting when the last bit of the input frame reaches the input port and the last bit of the output frame is seen on the output port

LIFO (Last In First Out) The time interval starting when the last bit of the input frame reaches the input port and ending when the

first bit of the output frame is seen on the output port.

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

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

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

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

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

2.2 Discussion

FILO is the most important measuring definition. Any type of switches MUST be measured with the FILO mechanism: FILO will include the latency of the switch and the latency of the frame as well as the serialization delay. It is a picture of the 'whole' latency going through the DUT. For applications, which are latency sensitive and can function with initial bytes of the frame, FIFO MAY be an additional type of measuring to supplement FILO.

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

2.3 Measurement Units

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

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

2) FIFO MAY be used for certain applications able to proceed data as

the first bits arrive (FPGA for example)

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

3 Jitter

3.1 Definition

The definition of Jitter is covered extensively in RFC 3393. This definition is not meant to replace that definition, but it is meant to provide guidance of use for data center network devices.

The use of Jitter is in according with the variation delay definition from RFC 3393:

The second meaning has to do with the variation of a metric (e.g., delay) with respect to some reference metric (e.g., average delay or minimum delay). This meaning is frequently used by computer scientists and frequently (but not always) refers to variation in delay.

Even with the reference to RFC 3393, there are many definitions of "jitter" possible. The one selected for Data Center Benchmarking is closest to RFC 3393.

3.2 Discussion

Jitter can be measured in different scenarios:-packet to packet delay variation-delta between min and max packet delay variation for all packets sent.

3.3 Measurement Units

The jitter MUST be measured when sending packets of the same size. Jitter MUST be measured as packet to packet delay variation and delta between min and max packet delay variation of all packets sent. A histogram MAY be provided as a population of packets measured per latency or latency buckets.

4 Physical Layer Calibration

4.1 Definition

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

the DUT.

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

- type of device used to generate traffic / measure traffic
- type of line cards used on the traffic generator
- type of transceivers on traffic generator
- type of transceivers on DUT
- type of cables
- length of cables
- software name, and version of traffic generator and DUT
- list of enabled features on DUT MAY be provided and is recommended [especially the control plane protocols such as LLDP, Spanning-Tree etc.]. A comprehensive configuration file MAY be provided to this effect.

4.2 Discussion

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

4.3 Measurement Units

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

5 Line rate

5.1 Definition

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

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

The term port capacity term defines the maximum speed capability for the given port; for example 1GE, 10GE, 40GE, 100GE etc.

The frequency ("clock rate") of the transmit clock in any two connected interfaces will never be precisely the same, therefore a tolerance is needed, this will be expressed by Parts Per Million (PPM) value. The IEEE standards allow a specific +/- variance in the transmit clock rate, and Ethernet is designed to allow for small, normal variations between the two clock rates. This results in a tolerance of the line rate value when traffic is generated from a testing equipment to a DUT.

5.2 Discussion

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

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

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

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

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

When testing an Ethernet switch throughput at "line rate", any specific switch will have a clock rate variance. If a test set is running +1 PPM faster than a switch under test, and a sustained line rate test is performed, a gradual increase in latency and eventually packet drops as buffers fill and overflow in the switch can be observed. Depending on how much clock variance there is between the two connected systems, the effect may be seen after the traffic stream has been running for a few hundred microseconds, a few milliseconds, or seconds. The same low latency and no-packet-loss can be demonstrated by setting the test set link occupancy to slightly less than 100 percent link occupancy. Typically 99 percent link occupancy produces excellent low-latency and no packet loss. No Ethernet switch or router will have a transmit clock rate of exactly +/- 0.0 PPM. Very few (if any) test sets have a clock rate that is precisely +/- 0.0 PPM.

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

5.3 Measurement Units

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

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

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

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

In a production network, it is very unlikely to see precise line rate over a very brief period. There is no observable difference between

dropping packets at 99% of line rate and 100% of line rate. -Line rate CAN measured at 100% of line rate with a -100PPM adjustment. - Line rate SHOULD be measured at 99,98% with 0 PPM adjustment.-The PPM adjustment SHOULD only be used for a line rate type of measurement

6 Buffering

6.1 Buffer

6.1.1 Definition

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

Example: Buffer Size of DUT when sending 1518 bytes frames is 18 Mb.

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

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

Example: a DUT has been measured to have 3KB of port buffer for 1518

frame size packets and a total of 4.7 MB of maximum port buffer for 1518 frame size packets and a cos of 0.

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

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

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

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

Intensity of Microburst: this is a percentage, representing the level of microburst between 1 and 100%. The higher the number the higher the microburst is. $I = [1 - \frac{(Tp2 - Tp1) + (Tp3 - Tp2) + \dots + (TpN - Tp(n-1))}{\text{Sum}(\text{packets})}] * 100$

6.1.3 Discussion

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

6.1.3 Measurement Units

When Buffer is measured:-the buffer size MUST be measured-the port

buffer size MAY be provided for each port-the maximum port buffer size MUST be measured-the maximum DUT buffer size MUST be measured-the intensity of microburst MAY be mentioned when a microburst test is performed-the cos or dscp value set during the test SHOULD be provided

6.2 Incast

6.2.1 Definition

The term Incast, very commonly utilized in the data center, refers to the traffic pattern of many-to-one or many-to-many conversations. Typically in the data center it would refer to many different ingress server ports(many), sending traffic to a common uplink (one), or multiple uplinks (many). This pattern is generalized for any network as many incoming ports sending traffic to one or few uplinks. It can also be found in many-to-many traffic patterns.

Synchronous arrival time: When two, or more, frames of respective sizes L1 and L2 arrive at their respective one or multiple ingress ports, and there is an overlap of the arrival time for any of the bits on the DUT, then the frames L1 and L2 have a synchronous arrival times. This is called incast.

Asynchronous arrival time: Any condition not defined by synchronous.

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

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

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

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

6.2.2 Discussion

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

egress buffer mechanism, the egress buffer of the one outgoing port would be used.

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

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

6.2.3 Measurement Units

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

7 Application Throughput: Data Center Goodput

7.1. Definition

In Data Center Networking, a balanced network is a function of maximal throughput 'and' minimal loss at any given time. This is defined by the Goodput. Goodput is the application-level throughput. It is measured in bytes / second. Goodput is the measurement of the actual payload of the packet being sent.

7.2. Discussion

In data center benchmarking, the goodput is a value that SHOULD be measured. It provides a realistic idea of the usage of the available bandwidth. A goal in data center environments is to maximize the goodput while minimizing the loss.

7.3. Measurement Units

When S is the total bytes received from all senders [not inclusive of packet headers or TCP headers - it's the payload] and Ft is the Finishing Time of the last sender; the Goodput G is then measured by the following formula: $G = S / Ft$ bytes per second

Example: a TCP file transfer over HTTP protocol on a 10Gb/s media. The file cannot be transferred over Ethernet as a single continuous stream. It must be broken down into individual frames of 1500 bytes when the standard MTU [Maximum Transmission Unit] is used. Each packet requires 20 bytes of IP header information and 20 bytes of TCP

header information, therefore 1460 byte are available per packet for the file transfer. Linux based systems are further limited to 1448 bytes as they also carry a 12 byte timestamp. Finally, the date is transmitted in this example over Ethernet which adds a 26 byte overhead per packet.

$G = 1460/1526 \times 10$ Gbit/s which is 9.567 Gbit/s or 1.196 Gigabytes per second.

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

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

- the TCP Stack used

- OS Versions

- NIC firmware version and model

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

8. References

3.1. Normative References

- [1] Bradner, S. "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, July 1991.
- [2] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.

3.2. Informative References

- [3] Mandeville R. and Perser J., "Benchmarking Methodology for LAN Switching Devices", RFC 2889, August 2000.
- [4] Stopp D. and Hickman B., "Methodology for IP Multicast Benchmarking", BCP 26, RFC 3918, October 2004.

3.3. URL References

- [5] Yanpei Chen, Rean Griffith, Junda Liu, Randy H. Katz, Anthony D.

Joseph, "Understanding TCP Incast Throughput Collapse in
Datacenter Networks",
<http://www.eecs.berkeley.edu/~ychen2/professional/TCPIncastWREN2009.pdf>
".

3.4. Acknowledgments

The authors would like to thank Ian Cox and Tim Stevenson for
their reviews and feedback.

Authors' Addresses

Lucien Avramov
Cisco Systems
170 West Tasman drive
San Jose, CA 95134
United States
Phone: +1 408 526 7686
Email: lavramov@cisco.com

Jacob Rapp
Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304
United States
Phone: +1 650 857 3367
Email: jacob.h.rapp@hp.com

Network Working Group
Internet Draft
Intended status: Informational
Expires: March 2015

M. Georgescu
NAIST
September 24, 2014

IPv6 Transition Technologies Benchmarking Methodology
draft-georgescu-ipv6-transition-tech-benchmarking-00.txt

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on March 24, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

There are benchmarking methodologies addressing the performance of network interconnect devices which are IPv4 or IPv6-capable. However, the IPv6 transition technologies are outside of their scope. This document provides complementary guidelines for evaluating the performance of IPv6 transition technologies. The methodology also includes a tentative metric for benchmarking scalability.

Table of Contents

1. Introduction.....	3
1.1. IPv6 transition technologies.....	3
2. Conventions used in this document.....	4
3. Test environment setup.....	4
3.1. Single-stack transition technologies.....	4
3.2. Encapsulation/Translation based transition technologies...	5
4. Test traffic.....	5
4.1. Frame formats and sizes.....	5
4.1.1. Frame sizes to be used over Ethernet.....	6
4.1.2. Frame sizes to be used over SONET.....	6
4.2. Protocol addresses.....	6
4.3. Traffic setup.....	6
5. Modifiers.....	7
6. Benchmarking tests.....	7
6.1. Throughput.....	7
6.2. Latency.....	7
6.3. Frame loss rate.....	7
6.4. Back-to-back frames.....	7
6.5. System recovery.....	8
6.6. Reset.....	8
7. Scalability.....	8
7.1. Test setup.....	8
7.1.1. Single-stack transition technologies.....	8
7.1.2. Encapsulation/Translation transition technologies...	9
7.2. Benchmarking performance degradation.....	9
8. Security Considerations.....	10
9. IANA Considerations.....	10
10. Conclusions.....	10
11. References.....	11
11.1. Normative References.....	11
11.2. Informative References.....	11
12. Acknowledgments.....	11
Appendix A. Theoretical maximum frame rates.....	12
A.1. Ethernet.....	12

1. Introduction

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

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

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

1.1. IPv6 transition technologies

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

Tentatively, we can consider a basic production IP-based network as being constructed using the following components:

- o a Customer Edge (CE) segment
- o a Core network segment
- o a Provider Edge (PE) segment

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

1. Single-stack: either IPv4 or IPv6 is used to traverse the core network and translation is used at one of the edges
2. Dual-stack: the core network devices implement both IP protocols

3. Encapsulation-based: an encapsulation mechanism is used to traverse the core network; CE nodes encapsulate the IPvX packets in IPvY packets, while PE nodes are responsible for the decapsulation process.
4. Translation-based: a translation mechanism is employed for the traversal of the network core; CE nodes translate IPvX packets to IPvY packets and PE nodes translate the packets back to IPvX.

The performance of Dual-stack transition technologies can be very well evaluated using the benchmarking methodology presented by [RFC2544] and [RFC5180]. Consequently the focus of this document is represented by the other 3 categories: Single-stack, Encapsulation-based and Translation-based transition technologies.

2. Conventions used in this document

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

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying [RFC2119] significance.

3. Test environment setup

The test environment setup options recommended for IPv6 transition technologies benchmarking are very similar to the ones presented in Section 6 of [RFC2544]. In the case of the tester setup, the options presented in [RFC2544] can be applied here as well. However, the Device under test (DUT) setup options should be explained in the context of the 3 targeted categories of IPv6 transition technologies: Single-stack, Encapsulation-based and Translation-based transition technologies.

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

3.1. Single-stack transition technologies

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

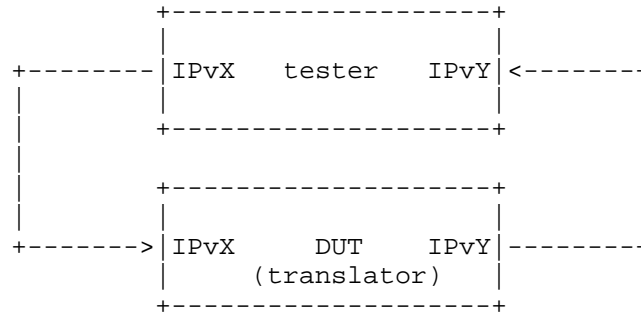


Figure 1

3.2. Encapsulation/Translation based transition technologies

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

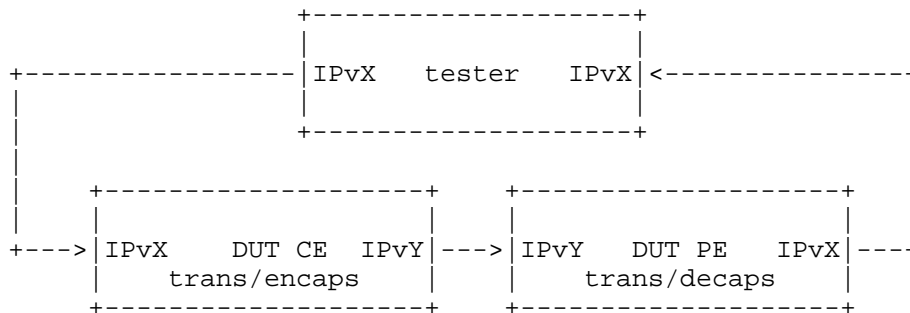


Figure 2

4. Test traffic

The test traffic represents the experimental workload and SHOULD meet the requirements specified in this section. The requirements are dedicated to unicast IP traffic.

4.1. Frame formats and sizes

[RFC5180] describes the frame size requirements for two commonly used media types: Ethernet and SONET (Synchronous Optical Network).

Internet-Draft IPv6 transition tech benchmarking September 2014
[RFC2544] covers also other media types, such as token ring and FDDI. The two documents can be referred for the dual-stack transition technologies. For the rest of the transition technologies the frame overhead introduced by translation or encapsulation MUST be considered.

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

4.1.1. Frame sizes to be used over Ethernet

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

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

4.1.2. Frame sizes to be used over SONET

Based on the recommendations of [RFC5180], the frame sizes for SONET traffic SHOULD be: 47, 64, 128, 256, 512, 1024, 1280, 1518, 2048, 4096 bytes.

An example of theoretical maximum frame rates calculation is shown in Appendix A2.

4.2. Protocol addresses

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

Note: testing traffic with extension headers might not be possible for the transition technologies which employ translation.

4.3. Traffic setup

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

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

6. Benchmarking tests

The benchmarking tests condition described in [RFC2544] (Sections 24, 25, 26) are also recommended here. The following sub-sections contain the list of all recommended benchmarking tests.

6.1. Throughput

Objective: To determine the DUT throughput as defined in [RFC1242].

Procedure: As described by [RFC2544].

Reporting Format: As described by [RFC2544].

6.2. Latency

Objective: To determine the latency as defined in [RFC1242].

Procedure: As described by [RFC2544].

Reporting Format: As described by [RFC2544].

6.3. Frame loss rate

Objective: To determine the frame loss rate, as defined in [RFC1242], of a DUT throughout the entire range of input data rates and frame sizes.

Procedure: As described by [RFC2544].

Reporting Format: As described by [RFC2544].

6.4. Back-to-back frames

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

Procedure: As described by [RFC2544].

6.5. System recovery

Objective: To characterize the speed at which a DUT recovers from an overload condition.

Procedure: As described by [RFC2544].

Reporting Format: As described by [RFC2544].

6.6. Reset

Objective: To characterize the speed at which a DUT recovers from a device or software reset.

Procedure: As described by [RFC6201].

Reporting Format: As described by [RFC6201].

7. Scalability

Scalability has been often discussed, however, in the context of network devices, a formal definition or a measurement method have not been approached yet.

Scalability can be defined as the ability of each transition technology to accommodate network growth.

Poor scalability usually leads to poor performance. Considering this, scalability can be measured by quantifying the network performance degradation while the network grows.

7.1. Test setup

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

7.1.1. Single-stack transition technologies

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

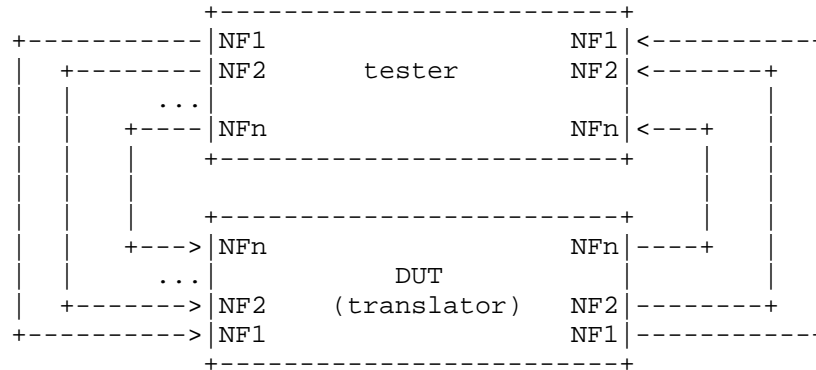


Figure 3

7.1.2. Encapsulation/Translation transition technologies

Similarly, for the encapsulation/translation based technologies a multi-flow setup is recommended. As for most transition technologies the provider edge device is designed to support more than one customer edge network, the recommended test setup is a n:1 design, where n is the number of CE DUTs connected to the same PE DUT (See Figure 4).

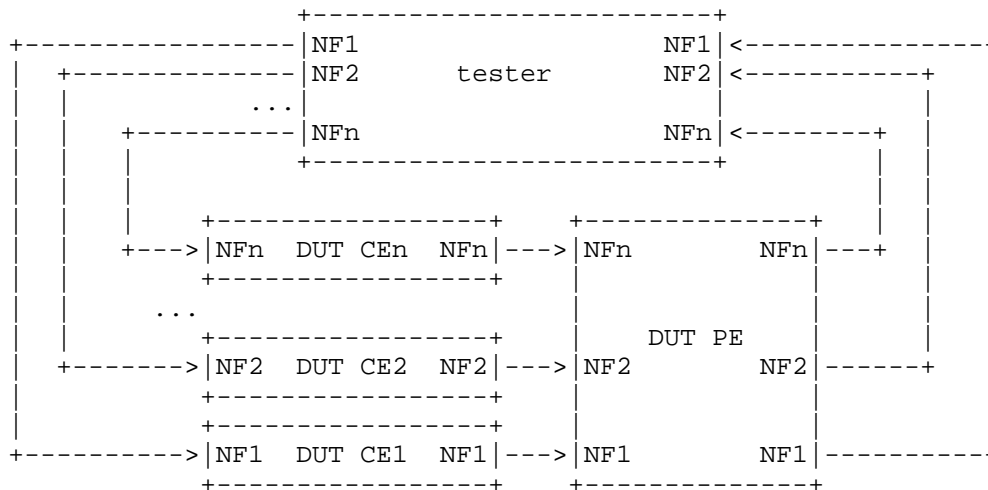


Figure 4

7.2. Benchmarking performance degradation

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

Internet-Draft IPv6 transition tech benchmarking September 2014
Procedure: First the benchmarking tests presented in Section 6 have to be performed for one network flow.

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

$$Xpd = \frac{X_n - X_1}{X_1} \times 100, \text{ where: } X_1 - \text{result for 1-flow} \\ X_n - \text{result for n-flows}$$

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

8. Security Considerations

The benchmarking methodology described in this document MUST be used in conjunction with a controlled experimental environment.

The benchmarking environment MUST be isolated and the generated traffic MUST NOT be forwarded into production networks.

Given the isolated nature of the experimental environment, no other security considerations are required.

9. IANA Considerations

The IANA has allocated the prefix 2001:0002::/48 [RFC5180] for IPv6 benchmarking. For IPv4 benchmarking, the 198.18.0.0/15 prefix was reserved, as described in [RFC6890]. The two ranges are sufficient for benchmarking IPv6 transition technologies.

10. Conclusions

The methodologies described in [RFC2544] and [RFC5180] can be used for benchmarking the performance of IPv4-only, IPv6-only and dual-stack supporting network devices. This document presents complementary recommendations dedicated to IPv6 transition technologies. Furthermore, the methodology includes a tentative approach for benchmarking scalability by quantifying the performance degradation associated with network growth.

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.
- [RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.
- [RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.
- [RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.
- [RFC6333] Cotton, M., Vegoda, L., Bonica, R., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC6890, April 2013.

11.2. Informative References

- [RFC1242] Bradner, S., "Benchmarking Terminology for Network Interconnection Devices", [RFC1242], July 1991.
- [RFC2544] Bradner, S., "Benchmarking Terminology for Network Interconnection Devices", [RFC1242], July 1991.
- [RFC5180] Popoviciu, C., Hamza, A., Van de Velde, G., and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", RFC 5180, May 2008.
- [RFC6201] Asati, R., Pignataro, C., Calabria, F., and C. Olvera, "Device Reset Characterization", RFC 6201, March 2011.

12. Acknowledgments

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

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

A.1. Ethernet

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

$$\frac{\text{Line Rate (bps)}}{(8\text{bits/byte}) * (X + O + 20)\text{bytes/frame}}$$

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

$$\frac{10,000,000(\text{bps})}{(8\text{bits/byte}) * (64 + 20 + 20)\text{bytes/frame}} = 12,019 \text{ fps}$$

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

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

A.2. SONET

Similarly for SONET, if X is the target frame size and O the frame size overhead, the recommended formula for calculating the maximum theoretical frame rate is:

$$\frac{\text{Line Rate (bps)}}{(8\text{bits/byte}) * (X+O+1)\text{bytes/frame}}$$

The calculation formula is based on the recommendation of RFC5180 in Appendix A2.

As an example, the frame rate recommended for testing a 6in4 implementation over a 10Mb/s PoS interface with 64 bytes frames is:

$$\frac{10,000,000(\text{bps})}{(8\text{bits/byte}) * (64+20+1)\text{bytes/frame}} = 14,706 \text{ fps}$$

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

Frame size (bytes)	10 Mb/s (fps)	100 Mb/s (fps)	1000 Mb/s (fps)	10000 Mb/s (fps)
47	18,382	183,824	1,838,235	18,382,353
64	14,706	147,059	1,470,588	14,705,882
128	8,389	83,893	838,926	8,389,262
256	4,513	45,126	451,264	4,512,635
512	2,345	23,452	234,522	2,345,216
1024	1,196	11,962	119,617	1,196,172
2048	604	6,042	60,416	604,157
4096	304	3,036	30,362	303,619

Marius Georgescu
Nara Institute of Science and Technology (NAIST)
Takayama 8916-5
Nara
Japan

Phone: +81 743 72 5216
Email: liviumarius-g@is.naist.jp

BMWG
Internet-Draft
Intended status: Informational
Expires: April 30, 2015

L. Huang, Ed.
R. Gu, Ed.
D. Liu
China Mobile
Bob. Mandeville
Iometrix
Brooks. Hickman
Spirent Communications
Guang. Zhang
IXIA
October 27, 2014

Benchmarking Methodology for Virtualization Network Performance
draft-huang-bmwg-virtual-network-performance-00

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 30, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Terminology	3
4. Key Performance Indicators	5
5. Test Setup	5
6. Benchmarking Tests	6
6.1. Throughput	6
6.1.1. Objectives	7
6.1.2. Configuration parameters	7
6.1.3. Test parameters	7
6.1.4. Test process	7
6.1.5. Test result format	7
6.2. CPU consumption	8
6.2.1. Objectives	8
6.2.2. Configuration parameters	8
6.2.3. Test parameters	9
6.2.4. Test process	9
6.2.5. Test result format	9
6.3. MEM consumption	9
6.3.1. Objectives	10
6.3.2. Configuration parameters	10
6.3.3. Test parameters	10
6.3.4. Test process	10
6.3.5. Test result format	10
6.4. Latency	11
6.4.1. Objectives	11
6.4.2. Configuration parameters	11
6.4.3. Test parameters	12
6.4.4. Test process	12
6.4.5. Test result format	12
7. Security Considerations	12
8. IANA Considerations	12
9. Normative References	12
Authors' Addresses	13

1. Introduction

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

2. Terminology

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

3. Terminology

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

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

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

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

Sample calibration permutation:

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

Figure 1: Sample Calibration Permutation

Key points are as following:

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

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

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

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

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

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

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

2. vSwitch has an influence on performance penalty due to extra VM addition.

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

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

4. Key Performance Indicators

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

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

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

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

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

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

5. Test Setup

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

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

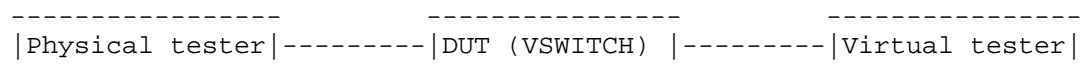


Figure 2: test model A

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

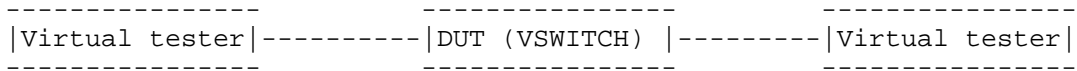


Figure 3: test model B

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

This test setup is also available in the VxLAN measurement.

6. Benchmarking Tests

6.1. Throughput

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

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

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

6.1.1. Objectives

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

6.1.2. Configuration parameters

Network parameters should be defined as follows:

- a) the number of virtual tester (VMs)
- b) the number of vNIC of virtual tester
- c) the CPU type of the server
- d) vCPU allocated for virtual tester (VMs)
- e) memory allocated for virtual tester (VMs)
- f) the number and rate of server NIC

6.1.3. Test parameters

- a) test repeated times
- b) test frame length

6.1.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch.
2. Increase the number of vCPU in the tester until the traffic has no packet loss.
3. Record the max throughput on VSwitch.
4. Change the frame length and repeat from step1 to step4.

6.1.5. Test result format

Byte	Throughput (Gbps)
0	0
128	0.46
256	0.84
512	1.56
1024	2.88
1518	4.00

Figure 4: test result format

6.2. CPU consumption

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

6.2.1. Objectives

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

6.2.2. Configuration parameters

Network parameters should be defined as follows:

- the number of virtual tester (VMs)
- the number of vNIC of virtual tester
- the CPU type of the server
- vCPU allocated for virtual tester (VMs)
- memory allocated for virtual tester (VMs)
- the number and rate of server NIC

6.2.3. Test parameters

- a) test repeated times
- b) test frame length

6.2.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch with the traffic value of throughput tested in 6.1.
2. Under the same throughput, record the CPU load value of server in the condition of shutting down and bypassing the DUT (VSWITCH), respectively.
3. Calculate the increase of the CPU load value due to establishing the DUT (VSWITCH).

6.2.5. Test result format

Byte	Throughput(GE)	Server CPU(MHZ)	VM CPU(MHz)
0	0	515	3042
128	0.46	6395	3040
256	0.84	6517	3042
512	1.56	6668	3041
1024	2.88	6280	3043
1450	4.00	6233	3045

test result format

6.3. MEM consumption

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

6.3.1. Objectives

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

6.3.2. Configuration parameters

Network parameters should be defined as follows:

- a) the number of virtual tester (VMs)
- b) the number of vNIC of virtual tester
- c) the CPU type of the server
- d) vCPU allocated for virtual tester (VMs)
- e) memory allocated for virtual tester (VMs)
- f) the number and rate of server NIC

6.3.3. Test parameters

- a) test repeated times
- b) test frame length

6.3.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch with the traffic value of throughput tested in 6.1.
2. Under the same throughput, record the memory consumption value of server in the condition of shutting down and bypassing the DUT (VSWITCH), respectively.
3. Calculate the increase of the memory consumption value due to establishing the DUT (VSWITCH).

6.3.5. Test result format

Byte	Throughput(GE)	Host Memory	VM Memory	
0	0	3042	696	
128	0.46	3040	696	
256	0.84	3042	696	
512	1.56	3041	696	
1024	2.88	3043	696	
1450	4.00	3045	696	

test result format

6.4. Latency

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

In the future, we may consider some other ways to have a better synchronization of clock to improve the accuracy of the test.

6.4.1. Objectives

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

6.4.2. Configuration parameters

Network parameters should be defined as follows:

- a) the number of virtual tester (VMs)
- b) the number of vNIC of virtual tester
- c) the CPU type of the server

- d) vCPU allocated for virtual tester (VMs)
- e) memory allocated for virtual tester (VMs)
- f) the number and rate of server NIC

6.4.3. Test parameters

- a) test repeated times
- b) test frame length

6.4.4. Test process

1. Configure the VM tester to offer traffic to the V-Switch with the traffic value of throughput tested in 6.1.
2. Under the same throughput, record the latency value of server in the condition of shutting down and bypassing the DUT (VSWITCH), respectively.
3. Calculate the increase of the latency value due to establishing the DUT (VSWITCH).

6.4.5. Test result format

TBD

7. Security Considerations

None.

8. IANA Considerations

None.

9. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

Authors' Addresses

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

Email: huanglu@chinamobile.com

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

Email: gurong@chinamobile.com

Dapeng Liu
China Mobile
32 Xuanwumen West Ave, Xicheng District
Beijing 100053
China

Email: gurong@chinamobile.com

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

Email: bob@iometrix.com

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

Email: Brooks.Hickman@spirent.com

Guang Zhang
IXIA

Email: GZhang@ixiacom.com

Benchmarking Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 27, 2014

R. Papneja
Huawei Technologies
B. Parise
Cisco Systems
S. Hares
Adara Networks
D. Lee
IXIA
I. Varlashkin
Easynet Global Services
June 25, 2014

Basic BGP Convergence Benchmarking Methodology for Data Plane
Convergence
draft-ietf-bmwg-bgp-basic-convergence-02.txt

Abstract

BGP is widely deployed and used by several service providers as the default Inter AS routing protocol. It is of utmost importance to ensure that when a BGP peer or a downstream link of a BGP peer fails, the alternate paths are rapidly used and routes via these alternate paths are installed. This document provides the basic BGP Benchmarking Methodology using existing BGP Convergence Terminology, RFC 4098.

Status of this Memo

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

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

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

This Internet-Draft will expire on December 27, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	4
1.1. Benchmarking Definitions	4
1.2. Purpose of BGP FIB (Data Plane) Convergence	4
1.3. Control Plane Convergence	5
1.4. Benchmarking Testing	5
2. Existing Definitions and Requirements	5
3. Test Topologies	6
3.1. General Reference Topologies	6
4. Test Considerations	8
4.1. Number of Peers	9
4.2. Number of Routes per Peer	9
4.3. Policy Processing/Reconfiguration	9
4.4. Configured Parameters (Timers, etc..)	9
4.5. Interface Types	11
4.6. Measurement Accuracy	11
4.7. Measurement Statistics	11
4.8. Authentication	12
4.9. Convergence Events	12
4.10. High Availability	12
5. Test Cases	12
5.1. Basic Convergence Tests	13
5.1.1. RIB-IN Convergence	13
5.1.2. RIB-OUT Convergence	15
5.1.3. eBGP Convergence	16
5.1.4. iBGP Convergence	16
5.1.5. eBGP Multihop Convergence	17
5.2. BGP Failure/Convergence Events	18
5.2.1. Physical Link Failure on DUT End	18
5.2.2. Physical Link Failure on Remote/Emulator End	19
5.2.3. ECMP Link Failure on DUT End	20
5.3. BGP Adjacency Failure (Non-Physical Link Failure) on Emulator	20
5.4. BGP Hard Reset Test Cases	21
5.4.1. BGP Non-Recovering Hard Reset Event on DUT	21
5.5. BGP Soft Reset	22
5.6. BGP Route Withdrawal Convergence Time	24
5.7. BGP Path Attribute Change Convergence Time	26
5.8. BGP Graceful Restart Convergence Time	27
6. Reporting Format	29
7. IANA Considerations	32
8. Security Considerations	32
9. Acknowledgements	32
10. References	32
10.1. Normative References	32
10.2. Informative References	33
Authors' Addresses	33

1. Introduction

This document defines the methodology for benchmarking data plane FIB convergence performance of BGP in routers and switches using topologies of 3 or 4 nodes. The methodology proposed in this document applies to both IPv4 and IPv6 and if a particular test is unique to one version, it is marked accordingly. For IPv6 benchmarking the device under test will require the support of Multi-Protocol BGP (MP-BGP) [RFC4760, RFC2545]. Similarly both iBGP & eBGP are covered in the tests as applicable.

The scope of this document is to provide methodology for BGP protocol FIB convergence measurements with BGP functionality limited to IPv4 & IPv6 as defined in RFC 4271 and Multi-Protocol BGP (MP-BGP) [RFC4760, RFC2545]. Other BGP extensions to support layer-2, layer-3 virtual private networks (VPN) are outside the scope of this document. Interaction with IGP (IGP interworking) is outside the scope of this document.

1.1. Benchmarking Definitions

The terminology used in this document is defined in [RFC4098]. One additional term is defined in this draft: FIB (Data plane) BGP Convergence.

FIB (Data plane) convergence is defined as the completion of all FIB changes so that all forwarded traffic now takes the new proposed route. RFC 4098 defines the terms BGP device, FIB and the forwarded traffic. Data plane convergence is different than control plane convergence within a node.

This document defines methodology to test

- Data plane convergence on a single BGP device that supports the BGP functionality with scope as outlined above
- using test topology of 3 or 4 nodes which are sufficient to recreate the Convergence events used in the various tests of this draft

1.2. Purpose of BGP FIB (Data Plane) Convergence

In the current Internet architecture the Inter-Autonomous System (inter-AS) transit is primarily available through BGP. To maintain reliable connectivity within intra-domains or across inter-domains, fast recovery from failures remains most critical. To ensure minimal traffic losses, many service providers are requiring BGP implementations to converge the entire Internet routing table within

sub-seconds at FIB level.

Furthermore, to compare these numbers amongst various devices, service providers are also looking at ways to standardize the convergence measurement methods. This document offers test methods for simple topologies. These simple tests will provide a quick high-level check of the BGP data plane convergence across multiple implementations from different vendors.

1.3. Control Plane Convergence

The convergence of BGP occurs at two levels: RIB and FIB convergence. RFC 4098 defines terms for BGP control plane convergence. Methodologies which test control plane convergence are out of scope for this draft.

1.4. Benchmarking Testing

In order to ensure that the results obtained in tests are repeatable, careful setup of initial conditions and exact steps are required.

This document proposes these initial conditions, test steps, and result checking. To ensure uniformity of the results all optional parameters SHOULD be disabled and all settings SHOULD be changed to default, these may include BGP timers as well.

2. Existing Definitions and Requirements

RFC 1242, "Benchmarking Terminology for Network Interconnect Devices" [RFC1242] and RFC 2285, "Benchmarking Terminology for LAN Switching Devices" [RFC2285] SHOULD be reviewed in conjunction with this document. WLAN-specific terms and definitions are also provided in Clauses 3 and 4 of the IEEE 802.11 standard [802.11]. Commonly used terms may also be found in RFC 1983 [RFC1983].

For the sake of clarity and continuity, this document adopts the general template for benchmarking terminology set out in Section 2 of RFC 1242. Definitions are organized in alphabetical order, and grouped into sections for ease of reference. The following terms are assumed to be taken as defined in RFC 1242 [RFC1242]: Throughput, Latency, Constant Load, Frame Loss Rate, and Overhead Behavior. In addition, the following terms are taken as defined in [RFC2285]: Forwarding Rates, Maximum Forwarding Rate, Loads, Device Under Test (DUT), and System Under Test (SUT).

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].

3. Test Topologies

This section describes the test setups for use in BGP benchmarking tests measuring convergence of the FIB (data plane) after the BGP updates has been received.

These test setups have 3 or 4 nodes with the following configuration:

1. Basic Test Setup
2. Three node setup for iBGP or eBGP convergence
3. Setup for eBGP multihop test scenario
4. Four node setup for iBGP or eBGP convergence

Individual tests refer to these topologies.

Figures 1-4 use the following conventions

- o AS-X: Autonomous System X
- o Loopback Int: Loopback interface on the BGP enabled device
- o HLP,HLP1,HLP2: Helper routers running the same version of BGP as DUT
- o Enable NTP or use any external clock source to synchronize to the nodes

3.1. General Reference Topologies

Emulator acts as 1 or more BGP peers for different testcases.

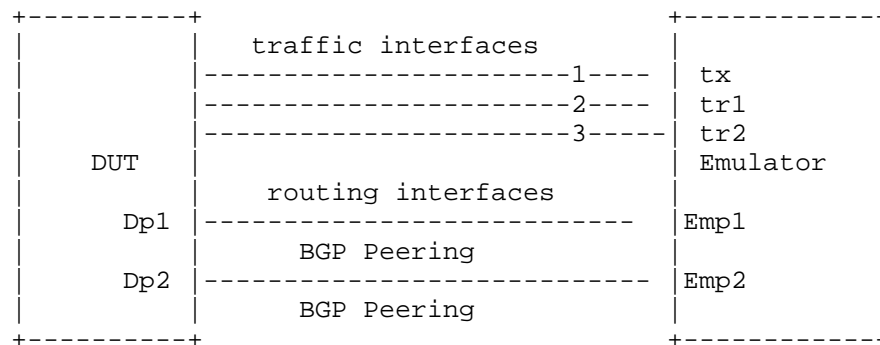


Figure 1 Basic Test Setup

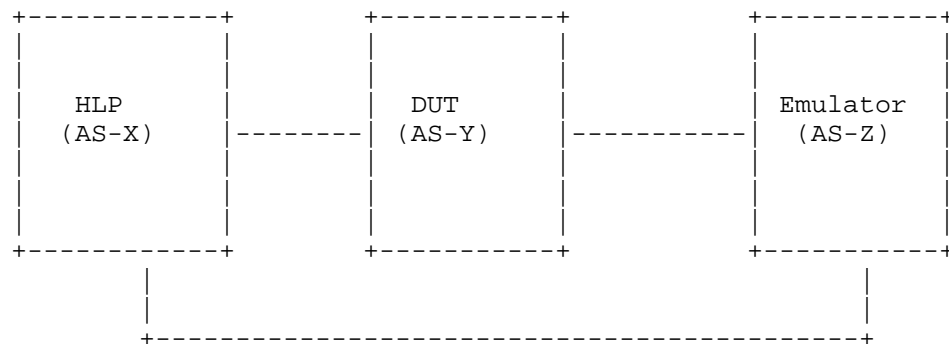


Figure 2 Three Node Setup for eBGP and iBGP Convergence

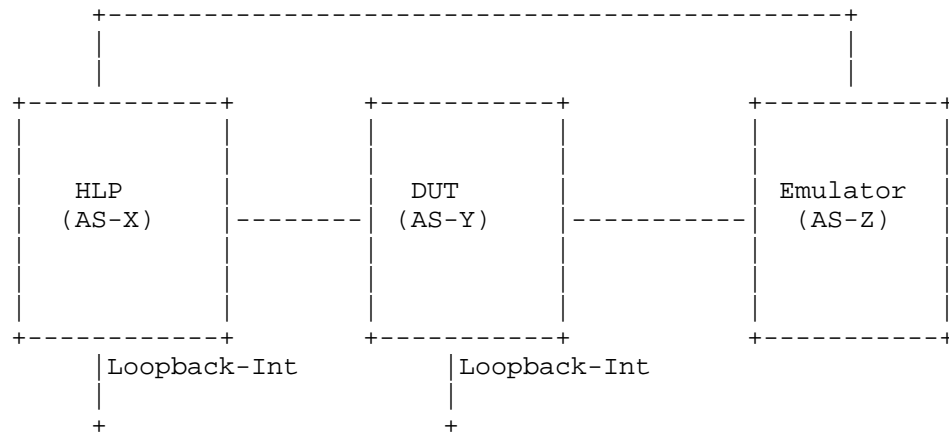


Figure 3 BGP Convergence for eBGP Multihop Scenario

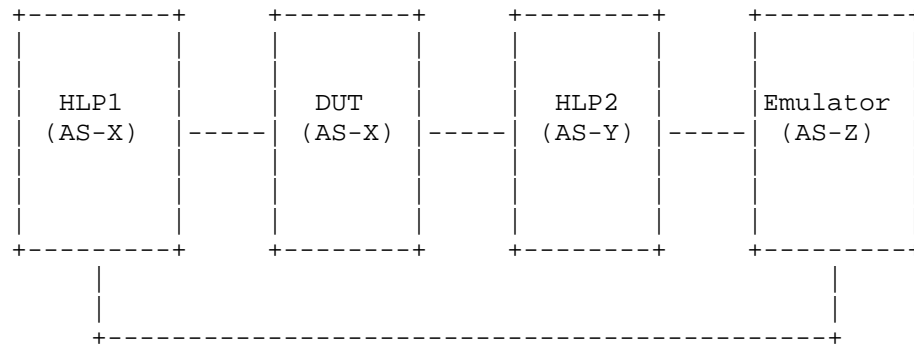


Figure 4 Four Node Setup for EBGP and IBGP Convergence

4. Test Considerations

The test cases for measuring convergence for iBGP and eBGP are different. Both iBGP and eBGP use different mechanisms to advertise, install and learn the routes. Typically, an iBGP route on the DUT is installed and exported when the next-hop is valid. For eBGP the

route is installed on the DUT with the remote interface address as the next-hop, with the exception of the multihop test case (as specified in the test).

4.1. Number of Peers

Number of Peers is defined as the number of BGP neighbors or sessions the DUT has at the beginning of the test. The peers are established before the tests begin. The relationship could be either, iBGP or eBGP peering depending upon the test case requirement.

The DUT establishes one or more BGP sessions with one more emulated routers or helper nodes. Additional peers can be added based on the testing requirements. The number of peers enabled during the testing should be well documented in the report matrix.

4.2. Number of Routes per Peer

Number of Routes per Peer is defined as the number of routes advertised or learnt by the DUT per session or through a neighbor relationship with an emulator or helper node. The tester, emulating as neighbor MUST advertise at least one route per peer.

Each test run must identify the route stream in terms of route packing, route mixture, and number of routes. This route stream must be well documented in the reporting stream. RFC 4098 defines these terms.

It is RECOMMENDED that the user consider advertising the entire current Internet routing table per peering session using an Internet route mixture with unique or non-unique routes. If multiple peers are used, it is important to precisely document the timing sequence between the peer sending routes (as defined in RFC 4098).

4.3. Policy Processing/Reconfiguration

The DUT MUST run one baseline test where policy is Minimal policy as defined in RFC 4098. Additional runs may be done with policy set-up before the tests begin. Exact policy settings MUST be documented as part of the test.

4.4. Configured Parameters (Timers, etc..)

There are configured parameters and timers that may impact the measured BGP convergence times.

The benchmark metrics MAY be measured at any fixed values for these configured parameters.

It is RECOMMENDED these configure parameters have the following settings: a) default values specified by the respective RFC b) platform-specific default parameters and c) values as expected in the operational network. All optional BGP settings MUST be kept consistent across iterations of any specific tests

Examples of the configured parameters that may impact measured BGP convergence time include, but are not limited to:

1. Interface failure detection timer
2. BGP Keepalive timer
3. BGP Holdtime
4. BGP update delay timer
5. ConnectRetry timer
6. TCP Segment Size
7. Minimum Route Advertisement Interval (MRAI)
8. MinASOriginationInterval (MAOI)
9. Route Flap Dampening parameters
10. TCP MD5
11. Maximum TCP Window Size
12. MTU

The basic-test settings for the parameters should be:

1. Interface failure detection timer (0 ms)
2. BGP Keepalive timer (1 min)
3. BGP Holdtime (3 min)
4. BGP update delay timer (0 s)

5. ConnectRetry timer (1 s)
6. TCP Segment Size (4096)
7. Minimum Route Advertisement Interval (MRAI) (0 s)
8. MinASOriginationInterval (MAOI)(0 s)
9. Route Flap Dampening parameters (off)
10. TCP MD5 (off)

4.5. Interface Types

The type of media dictate which test cases may be executed, each interface type has unique mechanism for detecting link failures and the speed at which that mechanism operates will influence the measurement results. All interfaces MUST be of the same media and throughput for all iterations of each test case.

4.6. Measurement Accuracy

Since observed packet loss is used to measure the route convergence time, the time between two successive packets offered to each individual route is the highest possible accuracy of any packet-loss based measurement. When packet jitter is much less than the convergence time, it is a negligible source of error and hence it will be treated as within tolerance.

Other options to measure convergence are the Time-Based Loss Method (TBLM) and Timestamp Based Method(TBM)[MPLSProt].

An exterior measurement on the input media (such as Ethernet) is defined by this specification.

4.7. Measurement Statistics

The benchmark measurements may vary for each trial, due to the statistical nature of timer expirations, CPU scheduling, etc. It is recommended to repeat the test multiple times. Evaluation of the test data must be done with an understanding of generally accepted testing practices regarding repeatability, variance and statistical significance of a small number of trials.

For any repeated tests that are averaged to remove variance, all parameters MUST remain the same.

4.8. Authentication

Authentication in BGP is done using the TCP MD5 Signature Option [RFC5925]. The processing of the MD5 hash, particularly in devices with a large number of BGP peers and a large amount of update traffic, can have an impact on the control plane of the device. If authentication is enabled, it MUST be documented correctly in the reporting format.

4.9. Convergence Events

Convergence events or triggers are defined as abnormal occurrences in the network, which initiate route flapping in the network, and hence forces the re-convergence of a steady state network. In a real network, a series of convergence events may cause convergence latency operators desire to test.

These convergence events must be defined in terms of the sequences defined in RFC 4098. This basic document begins all tests with a router initial set-up. Additional documents will define BGP data plane convergence based on peer initialization.

The convergence events may or may not be tied to the actual failure. A Soft Reset (RFC 4098) does not clear the RIB or FIB tables. A Hard reset clears the BGP peer sessions, the RIB tables, and FIB tables.

4.10. High Availability

Due to the different Non-Stop-Routing (sometimes referred to High-Availability) solutions available from different vendors, it is RECOMMENDED that any redundancy available in the routing processors should be disabled during the convergence measurements. For cases where the redundancy cannot be disabled, the results are no longer comparable and the level of impacts on the measurements is out of scope of this document.

5. Test Cases

All tests defined under this section assume the following:

- a. BGP peers are in established state
- b. BGP state should be cleared from established state to idle prior to each test. This is recommended to ensure that all tests start with the BGP peers being forced back to idle state and databases flushed.

- c. Furthermore the traffic generation and routing should be verified in the topology to ensure there is no packet loss observed on any advertised routes
- d. The arrival timestamp of advertised routes can be measured by installing an inline monitoring device between the emulator and DUT, or by the span port of DUT connected with an external analyzer. The time base of such inline monitor or external analyzer needs to be synchronized with the protocol and traffic emulator. Some modern emulator may have the capability to capture and timestamp every NLRI packets leaving and arriving at the emulator ports. The timestamps of these NLRI packets will be almost identical to the arrival time at DUT if the cable distance between the emulator and DUT is relatively short.

5.1. Basic Convergence Tests

These test cases measure characteristics of a BGP implementation in non-failure scenarios like:

- 1. RIB-IN Convergence
- 2. RIB-OUT Convergence
- 3. eBGP Convergence
- 4. iBGP Convergence

5.1.1. RIB-IN Convergence

Objective:

This test measures the convergence time taken to receive and install a route in RIB using BGP.

Reference Test Setup:

This test uses the setup as shown in figure 1

Procedure:

- A. All variables affecting Convergence should be set to a basic test state (as defined in section 4-4).
- B. Establish BGP adjacency between DUT and one peer of Emulator, Empl.
- C. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- D. Start the traffic from the Emulator tx towards the DUT targeted at a routes specified in route mixture (ex. routeA) Initially no traffic SHOULD be observed on the egress interface as the routeA is not installed in the forwarding database of the DUT.
- E. Advertise routeA from the peer(Empl) to the DUT and record the time.

This is $Tup(Empl, Rt-A)$ also named ' $XMT-Rt-time(Rt-A)$ '.

- F. Record the time when the routeA from Empl is received at the DUT.

This $Tup(DUT, Rt-A)$ also named ' $RCV-Rt-time(Rt-A)$ '.

- G. Record the time when the traffic targeted towards routeA is received by Emulator on appropriate traffic egress interface.

This is $TR(TDr, Rt-A)$. This is also named $DUT-XMT-Data-Time(Rt-A)$.

- H. The difference between the $Tup(DUT, RT-A)$ and traffic received time ($TR(TDr, Rt-A)$) is the FIB Convergence Time for routeA in the route mixture. A full convergence for the route update is the measurement between the 1st route ($Rt-A$) and the last route ($Rt-last$)

Route update convergence is

$TR(TDr, Rt-last) - Tup(DUT, Rt-A)$ or

$(DUT-XMT-Data-Time - RCV-Rt-Time)(Rt-A)$

Note: It is recommended that a single test with the same route mixture be repeated several times. A report should provide the Standard Deviation of all tests and the Average.

Running tests with a varying number of routes and route mixtures is important to get a full characterization of a single peer.

5.1.2. RIB-OUT Convergence

Objective:

This test measures the convergence time taken by an implementation to receive, install and advertise a route using BGP.

Reference Test Setup:

This test uses the setup as shown in figure 2.

Procedure:

- A. The Helper node (HLP) MUST run same version of BGP as DUT.
- B. All devices MUST be synchronized using NTP or some local reference clock.
- C. All configuration variables for HLP, DUT and Emulator SHOULD be set to the same values. These values MAY be basic-test or a unique set completely described in the test set-up.
- D. Establish BGP adjacency between DUT and Emulator.
- E. Establish BGP adjacency between DUT and Helper Node.
- F. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- G. Start the traffic from the Emulator towards the Helper Node targeted at a specific route (e.g. routeA). Initially no traffic SHOULD be observed on the egress interface as the routeA is not installed in the forwarding database of the DUT.
- H. Advertise routeA from the Emulator to the DUT and note the time.

This is $T_{up}(EMx, Rt-A)$, also named EM-XMT-Data-Time(Rt-A)

- I. Record when routeA is received by DUT.

This is $Tup(DUTr, Rt-A)$, also named $DUT-RCV-Rt-Time(Rt-A)$

- J. Record the time when the routeA is forwarded by DUT towards the Helper node.

This is $Tup(DUTx, Rt-A)$, also named $DUT-XMT-Rt-Time(Rt-A)$

- K. Record the time when the traffic targeted towards routeA is received on the Route Egress Interface. This is $TR(EMr, Rt-A)$, also named $DUT-XMT-Data\ Time(Rt-A)$.

$FIB\ convergence = (DUT-RCV-Rt-Time - DUT-XMT-Data-Time)(Rt-A)$

$RIB\ convergence = (DUT-RCV-Rt-Time - DUT-XMT-Rt-Time)(Rt-A)$

Convergence for a route stream is characterized by

- a) Individual route convergence for FIB, RIB
- b) All route convergence of

$FIB-convergence = DUT-RCV-Rt-Time(first) - DUT-XMT-Data-Time(last)$

$RIB-convergence = DUT-RCV-Rt-Time(first) - DUT-XMT-Rt-Time(last)$

5.1.3. eBGP Convergence

Objective:

This test measures the convergence time taken by an implementation to receive, install and advertise a route in an eBGP Scenario.

Reference Test Setup:

This test uses the setup as shown in figure 2 and the scenarios described in RIB-IN and RIB-OUT are applicable to this test case.

5.1.4. iBGP Convergence

Objective:

This test measures the convergence time taken by an implementation to receive, install and advertise a route in an iBGP Scenario.

Reference Test Setup:

This test uses the setup as shown in figure 2 and the scenarios described in RIB-IN and RIB-OUT are applicable to this test case.

5.1.5. eBGP Multihop Convergence

Objective:

This test measures the convergence time taken by an implementation to receive, install and advertise a route in an eBGP Multihop Scenario.

Reference Test Setup:

This test uses the setup as shown in figure 3. DUT is used along with a helper node.

Procedure:

- A. The Helper Node (HLP) MUST run the same version of BGP as DUT.
- B. All devices MUST be synchronized using NTP or some local reference clock.
- C. All variables affecting Convergence like authentication, policies, timers SHOULD be set to basic-settings
- D. All 3 devices, DUT, Emulator and Helper Node are configured with different Autonomous Systems.
- E. Loopback Interfaces are configured on DUT and Helper Node and connectivity is established between them using any config options available on the DUT.
- F. Establish BGP adjacency between DUT and Emulator.
- G. Establish BGP adjacency between DUT and Helper Node.
- H. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test

- I. Start the traffic from the Emulator towards the DUT targeted at a specific route (e.g. routeA).
- J. Initially no traffic SHOULD be observed on the egress interface as the routeA is not installed in the forwarding database of the DUT.
- K. Advertise routeA from the Emulator to the DUT and note the time (Tup(EMx,RouteA) also named Route-Tx-time(Rt-A).
- L. Record the time when the route is received by the DUT. This is Tup(EMr,DUT) named Route-Rcv-time(Rt-A).
- M. Record the time when the traffic targeted towards routeA is received from Egress Interface of DUT on emulator. This is Tup(EMd,DUT) named Data-Rcv-time(Rt-A)
- N. Record the time when the routeA is forwarded by DUT towards the Helper node. This is Tup(EMf,DUT) also named Route-Fwd-time(Rt-A)

$$\text{FIB Convergence} = (\text{Data-Rcv-time} - \text{Route-Rcv-time})(\text{Rt-A})$$

$$\text{RIB Convergence} = (\text{Route-Fwd-time} - \text{Route-Rcv-time})(\text{Rt-A})$$

Note: It is recommended that the test be repeated with varying number of routes and route mixtures. With each set route mixture, the test should be repeated multiple times. The results should record average, mean, Standard Deviation

5.2. BGP Failure/Convergence Events

5.2.1. Physical Link Failure on DUT End

Objective:

This test measures the route convergence time due to local link failure event at DUT's Local Interface.

Reference Test Setup:

This test uses the setup as shown in figure 1. Shutdown event is defined as an administrative shutdown event on the DUT.

Procedure:

- A. All variables affecting Convergence like authentication, policies, timers should be set to basic-test policy.
- B. Establish 2 BGP adjacencies from DUT to Emulator, one over the peer interface and the other using a second peer interface.
- C. Advertise the same route, routeA over both the adjacencies and (Emp1) Interface to be the preferred next hop.
- D. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- E. Start the traffic from the Emulator towards the DUT targeted at a specific route (e.g. routeA). Initially traffic would be observed on the best egress route (Emp1) instead of Emp2.
- F. Trigger the shutdown event of Best Egress Interface on DUT (Dp1).
- G. Measure the Convergence Time for the event to be detected and traffic to be forwarded to Next-Best Egress Interface (Dp2)
$$\text{Time} = \text{Data-detect}(\text{Emp2}) - \text{Shutdown time}$$
- H. Stop the offered load and wait for the queues to drain and Restart.
- I. Bring up the link on DUT Best Egress Interface.
- J. Measure the convergence time taken for the traffic to be rerouted from (Dp2) to Best Interface (Dp1)
$$\text{Time} = \text{Data-detect}(\text{Emp1}) - \text{Bring Up time}$$
- K. It is recommended that the test be repeated with varying number of routes and route mixtures or with number of routes & route mixtures closer to what is deployed in operational networks.

5.2.2. Physical Link Failure on Remote/Emulator End

Objective:

This test measures the route convergence time due to local link failure event at Tester's Local Interface.

Reference Test Setup:

This test uses the setup as shown in figure 1. Shutdown event is defined as shutdown of the local interface of Tester via logical shutdown event. The procedure used in 5.2.1 is used for the termination.

5.2.3. ECMP Link Failure on DUT End

Objective:

This test measures the route convergence time due to local link failure event at ECMP Member. The FIB configuration and BGP is set to allow two ECMP routes to be installed. However, policy directs the routes to be sent only over one of the paths

Reference Test Setup:

This test uses the setup as shown in figure 1 and the procedure uses 5.2.1.

5.3. BGP Adjacency Failure (Non-Physical Link Failure) on Emulator

Objective:

This test measures the route convergence time due to BGP Adjacency Failure on Emulator.

Reference Test Setup:

This test uses the setup as shown in figure 1.

Procedure:

- A. All variables affecting Convergence like authentication, policies, timers should be basic-policy set.
- B. Establish 2 BGP adjacencies from DUT to Emulator, one over the Best Egress Interface and the other using the Next-Best Egress Interface.
- C. Advertise the same route, routeA over both the adjacencies and make Best Egress Interface to be the preferred next hop

- D. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- E. Start the traffic from the Emulator towards the DUT targeted at a specific route (e.g. routeA). Initially traffic would be observed on the Best Egress interface.
- F. Remove BGP adjacency via a software adjacency down on the Emulator on the Best Egress Interface. This time is called BGPadj-down-time also termed BGPpeer-down
- G. Measure the Convergence Time for the event to be detected and traffic to be forwarded to Next-Best Egress Interface. This time is Tr-rr2 also called TR2-traffic-on

$$\text{Convergence} = \text{TR2-traffic-on} - \text{BGPpeer-down}$$

- H. Stop the offered load and wait for the queues to drain and Restart.
- I. Bring up BGP adjacency on the Emulator over the Best Egress Interface. This time is BGP-adj-up also called BGPpeer-up
- J. Measure the convergence time taken for the traffic to be rerouted to Best Interface. This time is BGP-adj-up also called BGPpeer-up

5.4. BGP Hard Reset Test Cases

5.4.1. BGP Non-Recovering Hard Reset Event on DUT

Objective:

This test measures the route convergence time due to Hard Reset on the DUT.

Reference Test Setup:

This test uses the setup as shown in figure 1.

Procedure:

- A. The requirement for this test case is that the Hard Reset Event should be non-recovering and should affect only the adjacency between DUT and Emulator on the Best Egress

Interface.

- B. All variables affecting SHOULD be set to basic-test values.
- C. Establish 2 BGP adjacencies from DUT to Emulator, one over the Best Egress Interface and the other using the Next-Best Egress Interface.
- D. Advertise the same route, routeA over both the adjacencies and make Best Egress Interface to be the preferred next hop.
- E. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- F. Start the traffic from the Emulator towards the DUT targeted at a specific route (e.g routeA). Initially traffic would be observed on the Best Egress interface.
- G. Trigger the Hard Reset event of Best Egress Interface on DUT.
- H. Measure the Convergence Time for the event to be detected and traffic to be forwarded to Next-Best Egress Interface.

Time of convergence = time-traffic flow - time-reset

- I. Stop the offered load and wait for the queues to drain and Restart.
- J. It is recommended that the test be repeated with varying number of routes and route mixtures or with number of routes & route mixtures closer to what is deployed in operational networks.
- K. When varying number of routes are used, convergence Time is measured using the Loss Derived method [IGPData].
- L. Convergence Time in this scenario is influenced by Failure detection time on Tester, BGP Keep Alive Time and routing, forwarding table update time.

5.5. BGP Soft Reset

Objective:

This test measures the route convergence time taken by an implementation to service a BGP Route Refresh message and advertise a route.

Reference Test Setup:

This test uses the setup as shown in figure 2.

Procedure:

- A. The BGP implementation on DUT & Helper Node needs to support BGP Route Refresh Capability [RFC2918].
- B. All devices MUST be synchronized using NTP or some local reference clock.
- C. All variables affecting Convergence like authentication, policies, timers should be set to basic-test defaults.
- D. DUT and Helper Node are configured in the same Autonomous System whereas Emulator is configured under a different Autonomous System.
- E. Establish BGP adjacency between DUT and Emulator.
- F. Establish BGP adjacency between DUT and Helper Node.
- G. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- H. Configure a policy under BGP on Helper Node to deny routes received from DUT.
- I. Advertise routeA from the Emulator to the DUT.
- J. The DUT will try to advertise the route to Helper Node will be denied.
- K. Wait for 3 KeepAlives.
- L. Start the traffic from the Emulator towards the Helper Node targeted at a specific route say routeA. Initially no traffic would be observed on the Egress interface, as routeA is not present.
- M. Remove the policy on Helper Node and issue a Route Refresh request towards DUT. Note the timestamp of this event. This is the RefreshTime.

- N. Record the time when the traffic targeted towards routeA is received on the Egress Interface. This is RecTime.
- O. The following equation represents the Route Refresh Convergence Time per route.

$$\text{Route Refresh Convergence Time} = (\text{RecTime} - \text{RefreshTime})$$

5.6. BGP Route Withdrawal Convergence Time

Objective:

This test measures the route convergence time taken by an implementation to service a BGP Withdraw message and advertise the withdraw.

Reference Test Setup:

This test uses the setup as shown in figure 2.

Procedure:

- A. This test consists of 2 steps to determine the Total Withdraw Processing Time.
- B. Step 1:
 - (1) All devices MUST be synchronized using NTP or some local reference clock.
 - (2) All variables should be set to basic-test parameters.
 - (3) DUT and Helper Node are configured in the same Autonomous System whereas Emulator is configured under a different Autonomous System.
 - (4) Establish BGP adjacency between DUT and Emulator.
 - (5) To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
 - (6) Start the traffic from the Emulator towards the DUT targeted at a specific route (e.g. routeA). Initially no traffic would be observed on the Egress interface as the routeA is not present on DUT.

- (7) Advertise routeA from the Emulator to the DUT.
- (8) The traffic targeted towards routeA is received on the Egress Interface.
- (9) Now the Tester sends request to withdraw routeA to DUT, TRx(Awith) also called WdrawTime1(Rt-A).
- (10) Record the time when no traffic is observed on the Egress Interface. This is the RouteRemoveTime1(Rt-A).
- (11) The difference between the RouteRemoveTime1 and WdrawTime1 is the WdrawConvTime1

$$\text{WdrawConvTime1(Rt-A)} = \text{RouteRemoveTime1(Rt-A)} - \text{WdrawTime1(Rt-A)}$$

C. Step 2:

- (1) Continuing from Step 1, re-advertise routeA back to DUT from Tester.
- (2) The DUT will try to advertise the routeA to Helper Node (This assumes there exists a session between DUT and helper node).
- (3) Start the traffic from the Emulator towards the Helper Node targeted at a specific route (e.g. routeA). Traffic would be observed on the Egress interface after routeA is received by the Helper Node

$$\text{WATime} = \text{time traffic first flows}$$

- (4) Now the Tester sends a request to withdraw routeA to DUT. This is the WdrawTime2(Rt-A)

$$\text{WAWtime-TRx(Rt-A)} = \text{WdrawTime2(Rt-A)}$$

- (5) DUT processes the withdraw and sends it to Helper Node.
- (6) Record the time when no traffic is observed on the Egress Interface of Helper Node. This is

$$\text{TR-WAW(DUT,RouteA)} = \text{RouteRemoveTime2(Rt-A)}$$

(7) Total withdraw processing time is

$$\text{TotalWdrawTime(Rt-A)} = ((\text{RouteRemoveTime2(Rt-A)} - \text{WdrawTime2(Rt-A)}) - \text{WdrawConvTime1(Rt-A)})$$

5.7. BGP Path Attribute Change Convergence Time

Objective:

This test measures the convergence time taken by an implementation to service a BGP Path Attribute Change.

Reference Test Setup:

This test uses the setup as shown in figure 1.

Procedure:

- A. This test only applies to Well-Known Mandatory Attributes like Origin, AS Path, Next Hop.
- B. In each iteration of test only one of these mandatory attributes need to be varied whereas the others remain the same.
- C. All devices MUST be synchronized using NTP or some local reference clock.
- D. All variables should be set to basic-test parameters.
- E. Advertise the route, routeA over the Best Egress Interface only, making it the preferred named Tbest.
- F. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- G. Start the traffic from the Emulator towards the DUT targeted at the specific route (e.g. routeA). Initially traffic would be observed on the Best Egress interface.
- H. Now advertise the same route routeA on the Next-Best Egress Interface but by varying one of the well-known mandatory attributes to have a preferred value over that interface. We call this Tbetter. The other values need to be same as what was advertised on the Best-Egress adjacency

$TRx(\text{Path-Change}(Rt-A)) = \text{Path Change Event Time}(Rt-A)$

- I. Measure the Convergence Time for the event to be detected and traffic to be forwarded to Next-Best Egress Interface

$DUT(\text{Path-Change}, Rt-A) = \text{Path-switch time}(Rt-A)$

$\text{Convergence} = \text{Path-switch time}(Rt-A) - \text{Path Change Event Time}(Rt-A)$

- J. Stop the offered load and wait for the queues to drain and Restart.

- K. Repeat the test for various attributes.

5.8. BGP Graceful Restart Convergence Time

Objective:

This test measures the route convergence time taken by an implementation during a Graceful Restart Event as detailed in the Terminology document [RFC4098].

Reference Test Setup:

This test uses the setup as shown in figure 4.

Procedure:

- A. It measures the time taken by an implementation to service a BGP Graceful Restart Event and advertise a route.
- B. The Helper Nodes are the same model as DUT and run the same BGP implementation as DUT.
- C. The BGP implementation on DUT & Helper Node needs to support BGP Graceful Restart Mechanism [RFC4724].
- D. All devices MUST be synchronized using NTP or some local reference clock.
- E. All variables are set to basic-test values.
- F. DUT and Helper Node-1(HLP1) are configured in the same Autonomous System whereas Emulator and Helper Node-2(HLP2) are configured under different Autonomous Systems.

- G. Establish BGP adjacency between DUT and Helper Nodes.
- H. Establish BGP adjacency between Helper Node-2 and Emulator.
- I. To ensure adjacency establishment, wait for 3 KeepAlives from the DUT or a configurable delay before proceeding with the rest of the test.
- J. Configure a policy under BGP on Helper Node-1 to deny routes received from DUT.
- K. Advertise routeA from the Emulator to Helper Node-2.
- L. Helper Node-2 advertises the route to DUT and DUT will try to advertise the route to Helper Node-1 which will be denied.
- M. Wait for 3 KeepAlives.
- N. Start the traffic from the Emulator towards the Helper Node-1 targeted at the specific route (e.g. routeA). Initially no traffic would be observed on the Egress interface as the routeA is not present.
- O. Perform a Graceful Restart Trigger Event on DUT and note the time. This is the GREventTime.
- P. Remove the policy on Helper Node-1.
- Q. Record the time when the traffic targeted towards routeA is received on the Egress Interface

TRr(DUT, routeA). This is also called RecTime(Rt-A)
- R. The following equation represents the Graceful Restart Convergence Time

$$\text{Graceful Restart Convergence Time(Rt-A)} = ((\text{RecTime(Rt-A)} - \text{GREventTime}) - \text{RIB-IN})$$
- S. It is assumed in this test case that after a Switchover is triggered on the DUT, it will not have any cycles to process BGP Refresh messages. The reason for this assumption is that there is a narrow window of time where after switchover when we remove the policy from Helper Node-1, implementations might generate Route-Refresh automatically and this request might be serviced before the DUT actually switches over and reestablishes BGP adjacencies with the peers.

6. Reporting Format

For each test case, it is recommended that the reporting tables below are completed and all time values SHOULD be reported with resolution as specified in [RFC4098].

Parameter	Units
Test case	Test case number
Test topology	1,2,3 or 4
Parallel links	Number of parallel links
Interface type	GigE, POS, ATM, other
Convergence Event	Hard reset, Soft reset, link failure, or other defined
eBGP sessions	Number of eBGP sessions
iBGP sessions	Number of iBGP sessions
eBGP neighbor	Number of eBGP neighbors
iBGP neighbor	Number of iBGP neighbors
Routes per peer	Number of routes
Total unique routes	Number of routes
Total non-unique routes	Number of routes
IGP configured	ISIS, OSPF, static, or other
Route Mixture	Description of Route mixture
Route Packing	Number of routes in an update
Policy configured	Yes, No
Packet size offered to the DUT	Bytes
Offered load	Packets per second
Packet sampling interval on tester	Seconds
Forwarding delay threshold	Seconds
Timer Values configured on DUT	
Interface failure indication delay	Seconds
Hold time	Seconds
MinRouteAdvertisementInterval (MRAI)	Seconds
MinASOriginationInterval (MAOI)	Seconds
Keepalive Time	Seconds
ConnectRetry	Seconds
TCP Parameters for DUT and tester	
MSS	Bytes
Slow start threshold	Bytes
Maximum window size	Bytes

Test Details:

- a. If the Offered Load matches a subset of routes, describe how this subset is selected.
- b. Describe how the Convergence Event is applied, does it cause instantaneous traffic loss or not.
- c. If there is any policy configured, describe the configured policy.

Complete the table below for the initial Convergence Event and the reversion Convergence Event

Parameter	Unit
Convergence Event	Initial or reversion
Traffic Forwarding Metrics	
Total number of packets offered to DUT	Number of packets
Total number of packets forwarded by DUT	Number of packets
Connectivity Packet Loss	Number of packets
Convergence Packet Loss	Number of packets
Out-of-order packets	Number of packets
Duplicate packets	Number of packets
Convergence Benchmarks	
Rate-derived Method[IGP-Data]:	
First route convergence time	Seconds
Full convergence time	Seconds
Loss-derived Method [IGP-Data]:	
Loss-derived convergence time	Seconds
Route-Specific Loss-Derived Method:	
Minimum R-S convergence time	Seconds
Maximum R-S convergence time	Seconds
Median R-S convergence time	Seconds
Average R-S convergence time	Seconds
Loss of Connectivity Benchmarks	
Loss-derived Method:	
Loss-derived loss of connectivity period	Seconds
Route-Specific loss-derived Method:	
Minimum LoC period [n]	Array of seconds
Minimum Route LoC period	Seconds
Maximum Route LoC period	Seconds
Median Route LoC period	Seconds
Average Route LoC period	Seconds

7. IANA Considerations

This draft does not require any new allocations by IANA.

8. Security Considerations

Benchmarking activities as described in this memo are limited to technology characterization using controlled stimuli in a laboratory environment, with dedicated address space and the constraints specified in the sections above.

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

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

9. Acknowledgements

We would like to thank Anil Tandon, Arvind Pandey, Mohan Nanduri, Jay Karthik, Eric Brendel for their input and discussions on various sections in the document. We also like to acknowledge Will Liu, Semion Lisiansky, Faisal Shah for their review and feedback to the document.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2918] Chen, E., "Route Refresh Capability for BGP-4", RFC 2918, September 2000.
- [RFC4098] Berkowitz, H., Davies, E., Hares, S., Krishnaswamy, P., and M. Lepp, "Terminology for Benchmarking BGP Device Convergence in the Control Plane", RFC 4098, June 2005.

- [RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.
- [RFC6412] Poretsky, S., Imhoff, B., and K. Michielsen, "Terminology for Benchmarking Link-State IGP Data-Plane Route Convergence", RFC 6412, November 2011.

10.2. Informative References

- [RFC1242] Bradner, S., "Benchmarking terminology for network interconnection devices", RFC 1242, July 1991.
- [RFC1983] Malkin, G., "Internet Users' Glossary", RFC 1983, August 1996.
- [RFC2285] Mandeville, R., "Benchmarking Terminology for LAN Switching Devices", RFC 2285, February 1998.
- [RFC2545] Marques, P. and F. Dupont, "Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing", RFC 2545, March 1999.
- [RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y. Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724, January 2007.
- [RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, January 2007.
- [RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, June 2010.

Authors' Addresses

Rajiv Papneja
Huawei Technologies

Email: rajiv.papneja@huawei.com

Bhavani Parise
Cisco Systems

Email: bhavani@cisco.com

Susan Hares
Adara Networks

Email: shares@ndzh.com

Dean Lee
IXIA

Email: dlee@ixiacom.com

Ilya Varlashkin
Easynet Global Services

Email: ilya.varlashkin@easynet.com

Benchmarking Methodology Working Group
Internet-Draft
Intended status: Informational
Expires: May 16, 2015

C. Davids
Illinois Institute of Technology
V. Gurbani
Bell Laboratories,
Alcatel-Lucent
S. Poretsky
Allot Communications
November 12, 2014

Methodology for Benchmarking Session Initiation Protocol (SIP) Devices:
Basic session setup and registration
draft-ietf-bmwg-sip-bench-meth-12

Abstract

This document provides a methodology for benchmarking the Session Initiation Protocol (SIP) performance of devices. Terminology related to benchmarking SIP devices is described in the companion terminology document. Using these two documents, benchmarks can be obtained and compared for different types of devices such as SIP Proxy Servers, Registrars and Session Border Controllers. The term "performance" in this context means the capacity of the device-under-test (DUT) to process SIP messages. Media streams are used only to study how they impact the signaling behavior. The intent of the two documents is to provide a normalized set of tests that will enable an objective comparison of the capacity of SIP devices. Test setup parameters and a methodology are necessary because SIP allows a wide range of configuration and operational conditions that can influence performance benchmark measurements.

Status of this Memo

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

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

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

This Internet-Draft will expire on May 16, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Terminology	4
2. Introduction	4
3. Benchmarking Topologies	5
4. Test Setup Parameters	7
4.1. Selection of SIP Transport Protocol	7
4.2. Connection-oriented Transport Management	7
4.3. Signaling Server	8
4.4. Associated Media	8
4.5. Selection of Associated Media Protocol	8
4.6. Number of Associated Media Streams per SIP Session	8
4.7. Codec Type	8
4.8. Session Duration	8
4.9. Attempted Sessions per Second (sps)	9
4.10. Benchmarking algorithm	9
5. Reporting Format	11
5.1. Test Setup Report	11
5.2. Device Benchmarks for session setup	12
5.3. Device Benchmarks for registrations	12
6. Test Cases	13
6.1. Baseline Session Establishment Rate of the test bed	13
6.2. Session Establishment Rate without media	13
6.3. Session Establishment Rate with Media not on DUT	13
6.4. Session Establishment Rate with Media on DUT	14
6.5. Session Establishment Rate with TLS Encrypted SIP	14
6.6. Session Establishment Rate with IPsec Encrypted SIP	15
6.7. Registration Rate	15
6.8. Re-Registration Rate	16
7. IANA Considerations	16
8. Security Considerations	16
9. Acknowledgments	17
10. References	17
10.1. Normative References	17
10.2. Informative References	17
Appendix A. R Code Component to simulate benchmarking algorithm	18
Authors' Addresses	20

1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, conforming to [RFC2119] and indicate requirement levels for compliant implementations.

RFC 2119 defines the use of these key words to help make the intent of standards track documents as clear as possible. While this document uses these keywords, this document is not a standards track document. The term Throughput is defined in [RFC2544].

Terms specific to SIP [RFC3261] performance benchmarking are defined in [I-D.sip-bench-term].

2. Introduction

This document describes the methodology for benchmarking Session Initiation Protocol (SIP) performance as described in the Terminology document [I-D.sip-bench-term]. The methodology and terminology are to be used for benchmarking signaling plane performance with varying signaling and media load. Media streams, when used, are used only to study how they impact the signaling behavior. This document concentrates on benchmarking SIP session setup and SIP registrations only.

The device-under-test (DUT) is a RFC3261-capable [RFC3261] network intermediary that plays the role of a registrar, redirect server, stateful proxy, a Session Border Controller (SBC) or a B2BUA. This document does not require the intermediary to assume the role of a stateless proxy. Benchmarks can be obtained and compared for different types of devices such as a SIP proxy server, Session Border Controllers (SBC), SIP registrars and a SIP proxy server paired with a media relay.

The test cases provide metrics for benchmarking the maximum 'SIP Registration Rate' and maximum 'SIP Session Establishment Rate' that the DUT can sustain over an extended period of time without failures (extended period of time is defined in the algorithm in Section 4.10). Some cases are included to cover encrypted SIP. The test topologies that can be used are described in the Test Setup section. Topologies in which the DUT handles media as well as those in which the DUT does not handle media are both considered. The measurement of the performance characteristics of the media itself is outside the scope of these documents.

Benchmark metrics could possibly be impacted by Associated Media. The selected values for Session Duration and Media Streams per Session enable benchmark metrics to be benchmarked without Associated Media. Session Setup Rate could possibly be impacted by the selected value for Maximum Sessions Attempted. The benchmark for Session Establishment Rate is measured with a fixed value for maximum Session Attempts.

Finally, the overall value of these tests is to serve as a comparison function between multiple SIP implementations. One way to use these tests is to derive benchmarks with SIP devices from Vendor-A, derive a new set of benchmarks with similar SIP devices from Vendor-B and perform a comparison on the results of Vendor-A and Vendor-B. This document does not make any claims on the interpretation of such results.

3. Benchmarking Topologies

Test organizations need to be aware that these tests generate large volumes of data and consequently ensure that networking devices like hubs, switches or routers are able to handle the generated volume.

The test cases enumerated in Section 6.1 to Section 6.6 operate on two test topologies: one in which the DUT does not process the media (Figure 1) and the other in which it does process media (Figure 2). In both cases, the tester or emulated agent (EA) sends traffic into the DUT and absorbs traffic from the DUT. The diagrams in Figure 1 and Figure 2 represent the logical flow of information and do not dictate a particular physical arrangement of the entities.

Figure 1 depicts a layout in which the DUT is an intermediary between the two interfaces of the EA. If the test case requires the exchange of media, the media does not flow through the DUT but rather passes directly between the two endpoints. Figure 2 shows the DUT as an intermediary between the two interfaces of the EA. If the test case requires the exchange of media, the media flows through the DUT between the endpoints.

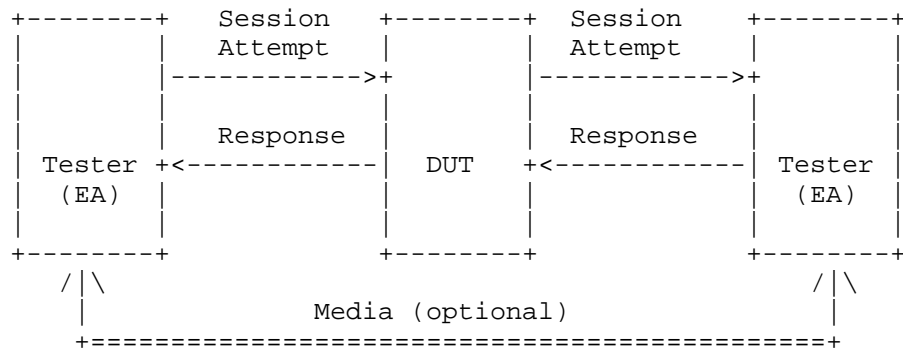


Figure 1: DUT as an intermediary, end-to-end media

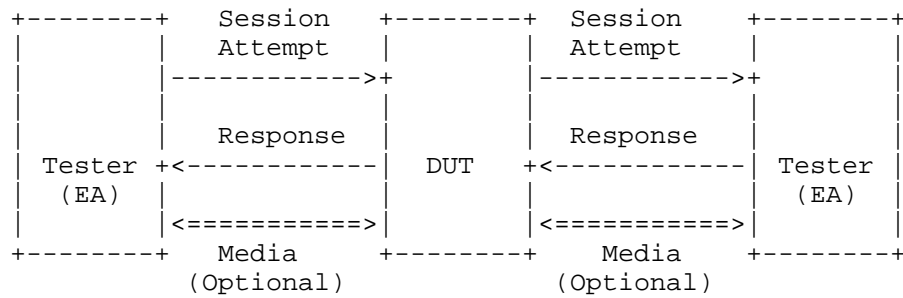


Figure 2: DUT as an intermediary forwarding media

The test cases enumerated in Section 6.7 and Section 6.8 use the topology in Figure 3 below.

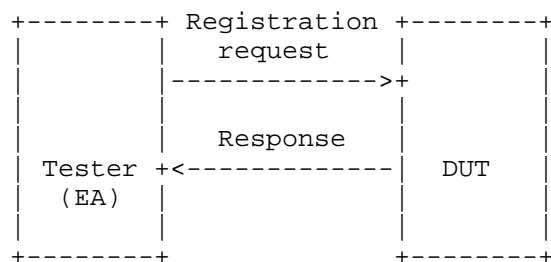


Figure 3: Registration and Re-registration tests

During registration or re-registration, the DUT may involve backend network elements and data stores. These network elements and data stores are not shown in Figure 3, but it is understood that they will impact the time required for the DUT to generate a response.

This document explicitly separates a registration test (Section 6.7) from a re-registration test (Section 6.8) because in certain networks, the time to re-register may vary from the time to perform an initial registration due to the backend processing involved. It is expected that the registration tests and the re-registration test will be performed with the same set of backend network elements in order to derive a stable metric.

4. Test Setup Parameters

4.1. Selection of SIP Transport Protocol

Test cases may be performed with any transport protocol supported by SIP. This includes, but is not limited to, TCP, UDP, TLS and websockets. The protocol used for the SIP transport protocol must be reported with benchmarking results.

SIP allows a DUT to use different transports for signaling on either side of the connection to the EAs. Therefore, this document assumes that the same transport is used on both sides of the connection; if this is not the case in any of the tests, the transport on each side of the connection **MUST** be reported in the test reporting template.

4.2. Connection-oriented Transport Management

SIP allows a device to open one connection and send multiple requests over the same connection (responses are normally received over the same connection that the request was sent out on). The protocol also allows a device to open a new connection for each individual request. A connection management strategy will have an impact on the results obtained from the test cases, especially for connection-oriented transports such as TLS. For such transports, the cryptographic handshake must occur every time a connection is opened.

The connection management strategy, i.e., use of one connection to send all requests or closing an existing connection and opening a new connection to send each request, **MUST** be reported with the benchmarking result.

4.3. Signaling Server

The Signaling Server is defined in the companion terminology document, ([I-D.sip-bench-term], Section 3.2.2). The Signaling Server is a DUT.

4.4. Associated Media

Some tests require Associated Media to be present for each SIP session. The test topologies to be used when benchmarking DUT performance for Associated Media are shown in Figure 1 and Figure 2.

4.5. Selection of Associated Media Protocol

The test cases specified in this document provide SIP performance independent of the protocol used for the media stream. Any media protocol supported by SIP may be used. This includes, but is not limited to, RTP, and SRTP. The protocol used for Associated Media MUST be reported with benchmarking results.

4.6. Number of Associated Media Streams per SIP Session

Benchmarking results may vary with the number of media streams per SIP session. When benchmarking a DUT for voice, a single media stream is used. When benchmarking a DUT for voice and video, two media streams are used. The number of Associated Media Streams MUST be reported with benchmarking results.

4.7. Codec Type

The test cases specified in this document provide SIP performance independent of the media stream codec. Any codec supported by the EAs may be used. The codec used for Associated Media MUST be reported with the benchmarking results.

4.8. Session Duration

The value of the DUT's performance benchmarks may vary with the duration of SIP sessions. Session Duration MUST be reported with benchmarking results. A Session Duration of zero seconds indicates transmission of a BYE immediately following a successful SIP establishment. Setting this parameter to the value '0' indicates that a BYE will be sent by the EA immediately after the EA receives a 200 OK to the INVITE. Setting this parameter to a time value greater than the duration of the test indicates that a BYE is never sent.

4.9. Attempted Sessions per Second (sps)

The value of the DUT's performance benchmarks may vary with the Session Attempt Rate offered by the tester. Session Attempt Rate MUST be reported with the benchmarking results.

The test cases enumerated in Section 6.1 to Section 6.6 require that the EA is configured to send the final 2xx-class response as quickly as it can. This document does not require the tester to add any delay between receiving a request and generating a final response.

4.10. Benchmarking algorithm

In order to benchmark the test cases uniformly in Section 6, the algorithm described in this section should be used. A prosaic description of the algorithm and a pseudo-code description are provided below, and a simulation written in the R statistical language [Rtool] is provided in Appendix A.

The goal is to find the largest value, *R*, a SIP Session Attempt Rate, measured in sessions-per-second (sps), which the DUT can process with zero errors over a defined, extended period. This period is defined as the amount of time needed to attempt *N* SIP sessions, where *N* is a parameter of test, at the attempt rate, *R*. An iterative process is used to find this rate. The algorithm corresponding to this process converges to *R*.

If the DUT vendor provides a value for *R*, the tester can use this value. In cases where the DUT vendor does not provide a value for *R*, or where the tester wants to establish the *R* of a system using local media characteristics, the algorithm should be run by setting "*r*", the session attempt rate, equal to a value of the tester's choice. For example the tester may initialize "*r* = 100" to start the algorithm and observe the value at convergence. The algorithm dynamically increases and decreases "*r*" as it converges to the a maximum sps value for *R*. The dynamic increase and decrease rate is controlled by the weights "*w*" and "*d*", respectively.

The pseudo-code corresponding to the description above follows, and a simulation written in the R statistical language is provided in Appendix A.

```
; ---- Parameters of test, adjust as needed
N := 50000 ; Global maximum; once largest session rate has
            ; been established, send this many requests before
            ; calling the test a success
m := {...} ; Other attributes that affect testing, such
```

```

; as media streams, etc.
r  := 100    ; Initial session attempt rate (in sessions/sec).
; Adjust as needed (for example, if DUT can handle
; thousands of calls in steady state, set to
; appropriate value in the thousands).
w  := 0.10   ; Traffic increase weight (0 < w <= 1.0)
d  := max(0.10, w / 2) ; Traffic decrease weight

; ---- End of parameters of test

proc find_R

    R = max_sps(r, m, N) ; Setup r sps, each with m media
; characteristics until N sessions have been attempted.
; Note that if a DUT vendor provides this number, the tester
; can use the number as a Session Attempt Rate, R, instead
; of invoking max_sps()

end proc

; Iterative process to figure out the largest number of
; sps that we can achieve in order to setup n sessions.
; This function converges to R, the Session Attempt Rate.
proc max_sps(r, m, n)
    s      := 0    ; session setup rate
    old_r  := 0    ; old session setup rate
    h      := 0    ; Return value, R
    count  := 0

    ; Note that if w is small (say, 0.10) and r is small
    ; (say, <= 9), the algorithm will not converge since it
    ; uses floor() to increment r dynamically. It is best
    ; off to start with the defaults (w = 0.10 and
    ; r >= 100)

    while (TRUE) {
        s := send_traffic(r, m, n) ; Send r sps, with m media
; characteristics until n sessions have been attempted.
        if (s == n) {
            if (r > old_r) {
                old_r = r
            }
        }
        else {
            count = count + 1
            if (count >= 10) {
                # We've converged.
                h := max(r, old_r)
                break
            }
        }
    }
end proc
```

```

        }
    }
    r := floor(r + (w * r))
}
else {
    r := floor(r - (d * r))
    d := max(0.10, d / 2)
    w := max(0.10, w / 2)
}
}
return h
end proc

```

5. Reporting Format

5.1. Test Setup Report

SIP Transport Protocol = _____
 (valid values: TCP|UDP|TLS|SCTP|websockets|specify-other)
 (specify if same transport used for connections to the DUT
 and connections from the DUT. If different transports
 used on each connection, enumerate the transports used)

Connection management strategy for connection oriented
 transports

DUT receives requests on one connection = _____
 (Yes or no. If no, DUT accepts a new connection for
 every incoming request, sends a response on that
 connection and closes the connection)
 DUT sends requests on one connection = _____
 (yes or no. If no, DUT initiates a new connection to
 send out each request, gets a response on that
 connection and closes the connection)

Session Attempt Rate _____
 (Session attempts/sec)
 (The initial value for "r" in Benchmarking Algorithm of
 Section 4.10)

Session Duration = _____
 (In seconds)

Total Sessions Attempted = _____
(Total sessions to be created over duration of test)

Media Streams Per Session = _____
(number of streams per session)

Associated Media Protocol = _____
(RTP|SRTP|specify-other)

Codec = _____
(Codec type as identified by the organization that specifies the codec)

Media Packet Size (audio only) = _____
(Number of bytes in an audio packet)

Establishment Threshold time = _____
(Seconds)

TLS ciphersuite used
(for tests involving TLS) = _____
(E.g., TLS_RSA_WITH_AES_128_CBC_SHA)

IPSec profile used
(For tests involving IPSEC) = _____

5.2. Device Benchmarks for session setup

Session Establishment Rate, "R" = _____
(sessions per second)
Is DUT acting as a media relay (yes/no) = _____

5.3. Device Benchmarks for registrations

Registration Rate = _____
(registrations per second)

Re-registration Rate = _____
(registrations per second)

Notes = _____
(List any specific backend processing required or other parameters that may impact the rate)

6. Test Cases

6.1. Baseline Session Establishment Rate of the test bed

Objective:

To benchmark the Session Establishment Rate of the Emulated Agent (EA) with zero failures.

Procedure:

1. Configure the DUT in the test topology shown in Figure 1.
2. Set media streams per session to 0.
3. Execute benchmarking algorithm as defined in Section 4.10 to get the baseline session establishment rate. This rate MUST be recorded using any pertinent parameters as shown in the reporting format of Section 5.1.

Expected Results: This is the scenario to obtain the maximum Session Establishment Rate of the EA and the test bed when no DUT is present. The results of this test might be used to normalize test results performed on different test beds or simply to better understand the impact of the DUT on the test bed in question.

6.2. Session Establishment Rate without media

Objective:

To benchmark the Session Establishment Rate of the DUT with no associated media and zero failures.

Procedure:

1. Configure a DUT according to the test topology shown in Figure 1 or Figure 2.
2. Set media streams per session to 0.
3. Execute benchmarking algorithm as defined in Section 4.10 to get the session establishment rate. This rate MUST be recorded using any pertinent parameters as shown in the reporting format of Section 5.1.

Expected Results: Find the Session Establishment Rate of the DUT when the EA is not sending media streams.

6.3. Session Establishment Rate with Media not on DUT

Objective:

To benchmark the Session Establishment Rate of the DUT with zero failures when Associated Media is included in the benchmark test but the media is not running through the DUT.

Procedure:

1. Configure a DUT according to the test topology shown in Figure 1.
2. Set media streams per session to 1.
3. Execute benchmarking algorithm as defined in Section 4.10 to get the session establishment rate with media. This rate **MUST** be recorded using any pertinent parameters as shown in the reporting format of Section 5.1.

Expected Results: Session Establishment Rate results obtained with Associated Media with any number of media streams per SIP session are expected to be identical to the Session Establishment Rate results obtained without media in the case where the DUT is running on a platform separate from the Media Relay.

6.4. Session Establishment Rate with Media on DUT**Objective:**

To benchmark the Session Establishment Rate of the DUT with zero failures when Associated Media is included in the benchmark test and the media is running through the DUT.

Procedure:

1. Configure a DUT according to the test topology shown in Figure 2.
2. Set media streams per session to 1.
3. Execute benchmarking algorithm as defined in Section 4.10 to get the session establishment rate with media. This rate **MUST** be recorded using any pertinent parameters as shown in the reporting format of Section 5.1.

Expected Results: Session Establishment Rate results obtained with Associated Media may be lower than those obtained without media in the case where the DUT and the Media Relay are running on the same platform. It may be helpful for the tester to be aware of the reasons for this degradation, although these reasons are not parameters of the test. For example, the degree of performance degradation may be due to what the DUT does with the media (e.g., relaying vs. transcoding), the type of media (audio vs. video vs. data), and the codec used for the media. There may also be cases where there is no performance impact, if the DUT has dedicated media-path hardware.

6.5. Session Establishment Rate with TLS Encrypted SIP

Objective:

To benchmark the Session Establishment Rate of the DUT with zero failures when using TLS encrypted SIP signaling.

Procedure:

1. If the DUT is being benchmarked as a proxy or B2BUA, then configure the DUT in the test topology shown in Figure 1 or Figure 2.
2. Configure the tester to enable TLS over the transport being used during benchmarking. Note the ciphersuite being used for TLS and record it in Section 5.1.
3. Set media streams per session to 0 (media is not used in this test).
4. Execute benchmarking algorithm as defined in Section 4.10 to get the session establishment rate with TLS encryption.

Expected Results: Session Establishment Rate results obtained with TLS Encrypted SIP may be lower than those obtained with plaintext SIP.

6.6. Session Establishment Rate with IPsec Encrypted SIP**Objective:**

To benchmark the Session Establishment Rate of the DUT with zero failures when using IPsec Encrypted SIP signaling.

Procedure:

1. Configure a DUT according to the test topology shown in Figure 1 or Figure 2.
2. Set media streams per session to 0 (media is not used in this test).
3. Configure tester for IPSec. Note the IPSec profile being used for and record it in Section 5.1.
4. Execute benchmarking algorithm as defined in Section 4.10 to get the session establishment rate with encryption.

Expected Results: Session Establishment Rate results obtained with IPSec Encrypted SIP may be lower than those obtained with plaintext SIP.

6.7. Registration Rate**Objective:**

To benchmark the maximum registration rate the DUT can handle over an extended time period with zero failures.

Procedure:

1. Configure a DUT according to the test topology shown in Figure 3.
2. Set the registration timeout value to at least 3600 seconds.
3. Each register request MUST be made to a distinct address of record (AoR). Execute benchmarking algorithm as defined in Section 4.10 to get the maximum registration rate. This rate MUST be recorded using any pertinent parameters as shown in the reporting format of Section 5.1. For example, the use of TLS or IPSec during registration must be noted in the reporting format. In the same vein, any specific backend processing (use of databases, authentication servers, etc.) SHOULD be recorded as well.

Expected Results: Provides a maximum registration rate.

6.8. Re-Registration Rate

Objective:

To benchmark the re-registration rate of the DUT with zero failures using the same backend processing and parameters used during Section 6.7.

Procedure:

1. Configure a DUT according to the test topology shown in Figure 3.
2. First, execute test detailed in Section 6.7 to register the endpoints with the registrar and obtain the registration rate.
3. After at least 5 minutes of Step 2, but no more than 10 minutes after Step 2 has been performed, re-register the same AoRs used in Step 3 of Section 6.7. This will count as a re-registration because the SIP AoRs have not yet expired.

Expected Results: Note the rate obtained through this test for comparison with the rate obtained in Section 6.7.

7. IANA Considerations

This document does not requires any IANA considerations.

8. Security Considerations

Documents of this type do not directly affect the security of Internet or corporate networks as long as benchmarking is not performed on devices or systems connected to production networks.

Security threats and how to counter these in SIP and the media layer is discussed in RFC3261, RFC3550, and RFC3711 and various other drafts. This document attempts to formalize a set of common methodology for benchmarking performance of SIP devices in a lab environment.

9. Acknowledgments

The authors would like to thank Keith Drage and Daryl Malas for their contributions to this document. Dale Worley provided an extensive review that lead to improvements in the documents. We are grateful to Barry Constantine, William Cerveney and Robert Sparks for providing valuable comments during the document's last calls and expert reviews. Al Morton and Sarah Banks have been exemplary working group chairs, we thank them for tracking this work to completion. Tom Taylor provided an in-depth review and subsequent comments on the benchmarking convergence algorithm in Section 4.10.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.
- [I-D.sip-bench-term]
Davids, C., Gurbani, V., and S. Poretsky, "SIP Performance Benchmarking Terminology",
draft-ietf-bmwg-sip-bench-term-12 (work in progress),
November 2014.

10.2. Informative References

- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [Rtool] R Development Core Team, "R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org/>, , 2011.

Appendix A. R Code Component to simulate benchmarking algorithm

```
# Copyright (c) 2014 IETF Trust and Vijay K. Gurbani. All
# rights reserved.
#
# Redistribution and use in source and binary forms, with
# or without modification, are permitted provided that the
# following conditions are met:
#
# * Redistributions of source code must retain the above
#   copyright notice, this list of conditions and the following
#   disclaimer.
# * Redistributions in binary form must reproduce the above
#   copyright notice, this list of conditions and the following
#   disclaimer in the documentation and/or other materials
#   provided with the distribution.
# * Neither the name of Internet Society, IETF or IETF Trust,
#   nor the names of specific contributors, may be used
#   to endorse or promote products derived from this software
#   without specific prior written permission.
#
# THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
# CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,
# INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
# MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
# DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
# CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
# SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
# BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
# SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
# INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
# WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
# NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
# USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
# OF SUCH DAMAGE.

w = 0.10
d = max(0.10, w / 2)
DUT_max_sps = 460      # Change as needed to set the max sps value
                        # for a DUT

# Returns R, given r (initial session attempt rate).
# E.g., assume that a DUT handles 460 sps in steady state
# and you have saved this code in a file simulate.r. Then,
# start an R session and do the following:
#
# > source("simulate.r")
```

```

# > find_R(100)
# ... debug output omitted ...
# [1] 458
#
# Thus, the max sps that the DUT can handle is 458 sps, which is
# close to the absolute maximum of 460 sps the DUT is specified to
# do.
find_R <- function(r) {
  s      = 0
  old_r  = 0
  h      = 0
  count  = 0

  # Note that if w is small (say, 0.10) and r is small
  # (say, <= 9), the algorithm will not converge since it
  # uses floor() to increment r dynamically. It is best
  # off to start with the defaults (w = 0.10 and
  # r >= 100)

  cat("r    old_r    w      d \n")
  while (TRUE) {
    cat(r, ' ', old_r, ' ', w, ' ', d, '\n')
    s = send_traffic(r)
    if (s == TRUE) {      # All sessions succeeded

      if (r > old_r) {
        old_r = r
      }
      else {
        count = count + 1

        if (count >= 10) {
          # We've converged.
          h = max(r, old_r)
          break
        }
      }

      r = floor(r + (w * r))
    }
    else {
      r = floor(r - (d * r))
      d = max(0.10, d / 2)
      w = max(0.10, w / 2)
    }
  }

  h
}

```

```
}  
  
send_traffic <- function(r) {  
  n = TRUE  
  
  if (r > DUT_max_sps) {  
    n = FALSE  
  }  
  
  n  
}
```

Authors' Addresses

Carol Davids
Illinois Institute of Technology
201 East Loop Road
Wheaton, IL 60187
USA

Phone: +1 630 682 6024
Email: davids@iit.edu

Vijay K. Gurbani
Bell Laboratories, Alcatel-Lucent
1960 Lucent Lane
Rm 9C-533
Naperville, IL 60566
USA

Phone: +1 630 224 0216
Email: vkg@bell-labs.com

Scott Poretsky
Allot Communications
300 TradeCenter, Suite 4680
Woburn, MA 08101
USA

Phone: +1 508 309 2179
Email: sporetsky@allot.com

Benchmarking Methodology Working Group
Internet-Draft
Intended status: Informational
Expires: May 16, 2015

C. Davids
Illinois Institute of Technology
V. Gurbani
Bell Laboratories,
Alcatel-Lucent
S. Poretsky
Allot Communications
November 12, 2014

Terminology for Benchmarking Session Initiation Protocol (SIP) Devices:
Basic session setup and registration
draft-ietf-bmwg-sip-bench-term-12

Abstract

This document provides a terminology for benchmarking the Session Initiation Protocol (SIP) performance of devices. Methodology related to benchmarking SIP devices is described in the companion methodology document. Using these two documents, benchmarks can be obtained and compared for different types of devices such as SIP Proxy Servers, Registrars and Session Border Controllers. The term "performance" in this context means the capacity of the device-under-test (DUT) to process SIP messages. Media streams are used only to study how they impact the signaling behavior. The intent of the two documents is to provide a normalized set of tests that will enable an objective comparison of the capacity of SIP devices. Test setup parameters and a methodology is necessary because SIP allows a wide range of configuration and operational conditions that can influence performance benchmark measurements. A standard terminology and methodology will ensure that benchmarks have consistent definition and were obtained following the same procedures.

Status of this Memo

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

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

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

This Internet-Draft will expire on May 16, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Terminology	4
2. Introduction	4
2.1. Scope	6
3. Term Definitions	7
3.1. Protocol Components	7
3.1.1. Session	7
3.1.2. Signaling Plane	8
3.1.3. Media Plane	8
3.1.4. Associated Media	9
3.1.5. Overload	9
3.1.6. Session Attempt	10
3.1.7. Established Session	10
3.1.8. Session Attempt Failure	11
3.2. Test Components	11
3.2.1. Emulated Agent	11
3.2.2. Signaling Server	12
3.2.3. SIP Transport Protocol	12
3.3. Test Setup Parameters	13
3.3.1. Session Attempt Rate	13
3.3.2. Establishment Threshold Time	13
3.3.3. Session Duration	14
3.3.4. Media Packet Size	14
3.3.5. Codec Type	15
3.4. Benchmarks	15
3.4.1. Session Establishment Rate	16
3.4.2. Registration Rate	16
3.4.3. Registration Attempt Rate	17
4. IANA Considerations	17
5. Security Considerations	17
6. Acknowledgments	18
7. References	18
7.1. Normative References	18
7.2. Informational References	19
Authors' Addresses	19

1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC2119 [RFC2119]. RFC 2119 defines the use of these key words to help make the intent of standards track documents as clear as possible. While this document uses these keywords, this document is not a standards track document. The term Throughput is defined in RFC2544 [RFC2544].

For the sake of clarity and continuity, this document adopts the template for definitions set out in Section 2 of RFC 1242 [RFC1242].

The term Device Under Test (DUT) is defined in the following BMWG documents:

Device Under Test (DUT) (c.f., Section 3.1.1 RFC 2285 [RFC2285]).

Many commonly used SIP terms in this document are defined in RFC 3261 [RFC3261]. For convenience the most important of these are reproduced below. Use of these terms in this document is consistent with their corresponding definition in the base SIP specification [RFC3261] as amended by [RFC4320], [RFC5393] and [RFC6026].

- o Call Stateful: A proxy is call stateful if it retains state for a dialog from the initiating INVITE to the terminating BYE request. A call stateful proxy is always transaction stateful, but the converse is not necessarily true.
- o Stateful Proxy: A logical entity, as defined by [RFC3261], that maintains the client and server transaction state machines during the processing of a request. (Also known as a transaction stateful proxy.) The behavior of a stateful proxy is further defined in Section 16 of RFC 3261 [RFC3261]. A transaction stateful proxy is not the same as a call stateful proxy.
- o Back-to-back User Agent: A back-to-back user agent (B2BUA) is a logical entity that receives a request and processes it as a user agent server (UAS). In order to determine how the request should be answered, it acts as a user agent client (UAC) and generates requests. Unlike a proxy server, it maintains dialog state and must participate in all requests sent on the dialogues it has established. Since it is a concatenation of a UAC and a UAS, no explicit definitions are needed for its behavior.

2. Introduction

Service Providers and IT Organizations deliver Voice Over IP (VoIP) and Multimedia network services based on the IETF Session Initiation

Protocol (SIP) [RFC3261]. SIP is a signaling protocol originally intended to be used to dynamically establish, disconnect and modify streams of media between end users. As it has evolved it has been adopted for use in a growing number of services and applications. Many of these result in the creation of a media session, but some do not. Examples of this latter group include text messaging and subscription services. The set of benchmarking terms provided in this document is intended for use with any SIP-enabled device performing SIP functions in the interior of the network, whether or not these result in the creation of media sessions. The performance of end-user devices is outside the scope of this document.

A number of networking devices have been developed to support SIP-based VoIP services. These include SIP Servers, Session Border Controllers (SBC) and Back-to-back User Agents (B2BUA). These devices contain a mix of voice and IP functions whose performance may be reported using metrics defined by the equipment manufacturer or vendor. The Service Provider or IT Organization seeking to compare the performance of such devices will not be able to do so using these vendor-specific metrics, whose conditions of test and algorithms for collection are often unspecified.

SIP functional elements and the devices that include them can be configured many different ways and can be organized into various topologies. These configuration and topological choices impact the value of any chosen signaling benchmark. Unless these conditions-of-test are defined, a true comparison of performance metrics across multiple vendor implementations will not be possible.

Some SIP-enabled devices terminate or relay media as well as signaling. The processing of media by the device impacts the signaling performance. As a result, the conditions-of-test must include information as to whether or not the device under test processes media. If the device processes media during the test, a description of the media must be provided. This document and its companion methodology document [I-D.ietf-bmwg-sip-bench-meth] provide a set of black-box benchmarks for describing and comparing the performance of devices that incorporate the SIP User Agent Client and Server functions and that operate in the network's core.

The definition of SIP performance benchmarks necessarily includes definitions of Test Setup Parameters and a test methodology. These enable the Tester to perform benchmarking tests on different devices and to achieve comparable results. This document provides a common set of definitions for Test Components, Test Setup Parameters, and Benchmarks. All the benchmarks defined are black-box measurements of the SIP signaling plane. The Test Setup Parameters and Benchmarks defined in this document are intended for use with the companion

Methodology document.

2.1. Scope

The scope of this document is summarized as follows:

- o This terminology document describes SIP signaling performance benchmarks for black-box measurements of SIP networking devices. Stress and debug scenarios are not addressed in this document.
- o The DUT must be RFC 3261 capable network equipment. This may be a Registrar, Redirect Server, or Stateful Proxy. This document does not require the intermediary to assume the role of a stateless proxy. A DUT may also include a B2BUA, SBC functionality.
- o The Tester acts as multiple "Emulated Agents" (EA) that initiate (or respond to) SIP messages as session endpoints and source (or receive) associated media for established connections.
- o SIP Signaling in presence of media
 - * The media performance is not benchmarked.
 - * Some tests require media, but the use of media is limited to observing the performance of SIP signaling. Tests that require media will annotate the media characteristics as a condition of test.
 - * The type of DUT dictates whether the associated media streams traverse the DUT. Both scenarios are within the scope of this document.
 - * SIP is frequently used to create media streams; the signaling plane and media plane are treated as orthogonal to each other in this document. While many devices support the creation of media streams, benchmarks that measure the performance of these streams are outside the scope of this document and its companion methodology document [I-D.ietf-bmwg-sip-bench-meth]. Tests may be performed with or without the creation of media streams. The presence or absence of media streams MUST be noted as a condition of the test as the performance of SIP devices may vary accordingly. Even if the media is used during benchmarking, only the SIP performance will be benchmarked, not the media performance or quality.
- o Both INVITE and non-INVITE scenarios (registrations) are addressed in this document. However, benchmarking SIP presence or subscribe-notify extensions is not a part of this document.
- o Different transport -- such as UDP, TCP, SCTP, or TLS -- may be used. The specific transport mechanism MUST be noted as a condition of the test as the performance of SIP devices may vary accordingly.
- o REGISTER and INVITE requests may be challenged or remain unchallenged for authentication purpose. Whether or not the REGISTER and INVITE requests are challenged is a condition of test which will be recorded along with other such parameters which may impact the SIP performance of the device or system under test.

- o Re-INVITE requests are not considered in scope of this document since the benchmarks for INVITEs are based on the dialog created by the INVITE and not on the transactions that take place within that dialog.
- o Only session establishment is considered for the performance benchmarks. Session disconnect is not considered in the scope of this document. This is because our goal is to determine the maximum capacity of the device or system under test, that is the number of simultaneous SIP sessions that the device or system can support. It is true that there are BYE requests being created during the test process. These transactions do contribute to the load on the device or system under test and thus are accounted for in the metric we derive. We do not seek a separate metric for the number of BYE transactions a device or system can support.
- o IMS-specific scenarios are not considered, but test cases can be applied with 3GPP-specific SIP signaling and the P-CSCF as a DUT.
- o The benchmarks described in this document are intended for a laboratory environment and are not intended to be used on a production network. Some of the benchmarks send enough traffic that a denial of service attack is possible if used in production networks.

3. Term Definitions

3.1. Protocol Components

3.1.1. Session

Definition:

The combination of signaling and media messages and associated processing that enable a single SIP-based audio or video call, or SIP registration.

Discussion:

The term "session" commonly implies a media session. In this document the term is extended to cover the signaling and any media specified and invoked by the corresponding signaling.

Measurement Units:

N/A.

Issues:

None.

See Also:

- Media Plane
- Signaling Plane
- Associated Media

3.1.2. Signaling Plane

Definition:

The plane in which SIP messages [RFC3261] are exchanged between SIP Agents [RFC3261].

Discussion:

SIP messages are used to establish sessions in several ways: directly between two User Agents [RFC3261], through a Proxy Server [RFC3261], or through a series of Proxy Servers. The Session Description Protocol (SDP) is included in the Signaling Plane.

Measurement Units:

N/A.

Issues:

None.

See Also:

- Media Plane
- EAs

3.1.3. Media Plane

Definition:

The data plane in which one or more media streams and their associated media control protocols (e.g., RTCP [RFC3550]) are exchanged between User Agents after a media connection has been created by the exchange of signaling messages in the Signaling Plane.

Discussion:

Media may also be known as the "bearer channel". The Media Plane MUST include the media control protocol, if one is used, and the media stream(s). Examples of media are audio and video. The media streams are described in the SDP of the Signaling Plane.

Measurement Units:

N/A.

Issues:

None.

See Also:

Signaling Plane

3.1.4. Associated Media

Definition:

Media that corresponds to an 'm' line in the SDP payload of the Signaling Plane.

Discussion:

The format of the media is determined by the SDP attributes for the corresponding 'm' line.

Measurement Units:

N/A.

Issues:

None.

3.1.5. Overload

Definition:

Overload is defined as the state where a SIP server does not have sufficient resources to process all incoming SIP messages [RFC6357].

Discussion:

The distinction between an overload condition and other failure scenarios is outside the scope of black box testing and of this document. Under overload conditions, all or a percentage of Session Attempts will fail due to lack of resources. In black box testing the cause of the failure is not explored. The fact that a failure occurred for whatever reason, will trigger the tester to reduce the offered load, as described in the companion methodology document, [I-D.ietf-bmwg-sip-bench-meth]. SIP server resources may include CPU processing capacity, network bandwidth, input/output queues, or disk resources. Any combination of resources may be fully utilized when a SIP server (the DUT) is in the overload condition. For proxy-only (or intermediary) devices, it is expected that the proxy will be driven into overload based on the delivery rate of signaling requests.

Measurement Units:

N/A.

3.1.6. Session Attempt

Definition:

A SIP INVITE or REGISTER request sent by the EA that has not received a final response.

Discussion:

The attempted session may be either an invitation to an audio/video communication or a registration attempt. When counting the number of session attempts we include all requests that are rejected for lack of authentication information. The EA needs to record the total number of session attempts including those attempts that are routinely rejected by a proxy that requires the UA to authenticate itself. The EA is provisioned to deliver a specific number of session attempts per second. But the EA must also count the actual number of session attempts per given time interval.

Measurement Units:

N/A.

Issues:

None.

See Also:

Session

Session Attempt Rate

3.1.7. Established Session

Definition:

A SIP session for which the EA acting as the UE/UA has received a 200 OK message.

Discussion:

An Established Session may be either an invitation to an audio/video communication or a registration attempt. Early dialogues for INVITE requests are out of scope for this work.

Measurement Units:

N/A.

Issues:

None.

See Also:

None.

3.1.8. Session Attempt Failure

Definition:

A session attempt that does not result in an Established Session.

Discussion:

The session attempt failure may be indicated by the following observations at the EA:

1. Receipt of a SIP 3xx-, 4xx-, 5xx-, or 6xx-class response to a Session Attempt.
2. The lack of any received SIP response to a Session Attempt within the Establishment Threshold Time (c.f. Section 3.3.2).

Measurement Units:

N/A.

Issues:

None.

See Also:

Session Attempt

3.2. Test Components

3.2.1. Emulated Agent

Definition:

A device in the test topology that initiates/responds to SIP messages as one or more session endpoints and, wherever applicable, sources/receives Associated Media for Established Sessions.

Discussion:

The EA functions in the Signaling and Media Planes. The Tester may act as multiple EAs.

Measurement Units:

N/A

Issues:

None.

See Also:

Media Plane
Signaling Plane
Established Session
Associated Media

3.2.2. Signaling Server

Definition:

Device in the test topology that facilitates the creation of sessions between EAs. This device is the DUT.

Discussion:

The DUT is a RFC3261-capable network intermediary such as a Registrar, Redirect Server, Stateful Proxy, B2BUA or SBC.

Measurement Units:

NA

Issues:

None.

See Also:

Signaling Plane

3.2.3. SIP Transport Protocol

Definition:

The protocol used for transport of the Signaling Plane messages.

Discussion:

Performance benchmarks may vary for the same SIP networking device depending upon whether TCP, UDP, TLS, SCTP, websockets [RFC7118] or any future transport layer protocol is used. For this reason it is necessary to measure the SIP Performance Benchmarks using these various transport protocols. Performance Benchmarks MUST report the SIP Transport Protocol used to obtain the benchmark results.

Measurement Units:

While these are not units of measure, they are attributes that are one of many factors that will contribute to the value of the measurements to be taken. TCP, UDP, SCTP, TLS over TCP, TLS over UDP, TLS over SCTP, and websockets are among the possible values to be recorded as part of the test.

Issues:

None.

See Also:

None.

3.3. Test Setup Parameters**3.3.1. Session Attempt Rate****Definition:**

Configuration of the EA for the number of sessions per second (sps) that the EA attempts to establish using the services of the DUT.

Discussion:

The Session Attempt Rate is the number of sessions per second that the EA sends toward the DUT. Some of the sessions attempted may not result in a session being established.

Measurement Units:

Session attempts per second

Issues:

None.

See Also:

Session
Session Attempt

3.3.2. Establishment Threshold Time**Definition:**

Configuration of the EA that represents the amount of time that an EA client will wait for a response from an EA server before declaring a Session Attempt Failure.

Discussion:

This time duration is test dependent.

It is RECOMMENDED that the Establishment Threshold Time value be set to Timer B or Timer F as specified in RFC 3261, Table 4 [RFC3261].

Measurement Units:

Seconds

Issues:

None.

See Also:

None.

3.3.3. Session Duration**Definition:**

Configuration of the EA that represents the amount of time that the SIP dialog is intended to exist between the two EAs associated with the test.

Discussion:

The time at which the BYE is sent will control the Session Duration.

Measurement Units:

seconds

Issues:

None.

See Also:

None.

3.3.4. Media Packet Size**Definition:**

Configuration on the EA for a fixed number of frames or samples to be sent in each RTP packet of the media stream when the test involves Associated Media.

Discussion:

This document describes a method to measure SIP performance. If the DUT is processing media as well as SIP messages the media processing will potentially slow down the SIP processing and lower the SIP performance metric. The tests with associated media are designed for audio codecs and the assumption was made that larger media packets would require more processor time. This document does not define parameters applicable to video codecs.

For a single benchmark test, media sessions use a defined number of samples or frames per RTP packet. If two SBCs, for example, used the same codec but one puts more frames into the RTP packet, this might cause variation in the performance benchmark results.

Measurement Units:

An integer number of frames or samples, depending on whether hybrid- or sample-based codec are used, respectively.

Issues:

None.

See Also:

None.

3.3.5. Codec Type

Definition:

The name of the codec used to generate the media session.

Discussion

For a single benchmark test, all sessions use the same size packet for media streams. The size of packets can cause a variation in the performance benchmark measurements.

Measurement Units:

This is a textual name (alphanumeric) assigned to uniquely identify the codec.

Issues:

None.

See Also:

None.

3.4. Benchmarks

3.4.1. Session Establishment Rate

Definition:

The maximum value of the Session Attempt Rate that the DUT can handle for an extended, pre-defined, period with zero failures.

Discussion:

This benchmark is obtained with zero failure. The session attempt rate provisioned on the EA is raised and lowered as described in the algorithm in the accompanying methodology document [I-D.ietf-bmwg-sip-bench-meth], until a traffic load over the period of time necessary to attempt N sessions completes without failure, where N is a parameter specified in the algorithm and recorded in the Test Setup Report.

Measurement Units:

sessions per second (sps)

Issues:

None.

See Also:

Invite-Initiated Sessions
Non-Invite-Initiated Sessions
Session Attempt Rate

3.4.2. Registration Rate

Definition:

The maximum value of the Registration Attempt Rate that the DUT can handle for an extended, pre-defined, period with zero failures.

Discussion:

This benchmark is obtained with zero failures. The registration rate provisioned on the Emulated Agent is raised and lowered as described in the algorithm in the companion methodology draft [I-D.ietf-bmwg-sip-bench-meth], until a traffic load consisting of registration attempts at the given attempt rate over the period of time necessary to attempt N registrations completes without failure, where N is a parameter specified in the algorithm and recorded in the Test Setup Report.

This benchmark is described separately from the Session Establishment Rate (Section 3.4.1), although it could be considered a special case of that benchmark, since a REGISTER request is a request for a Non-Invite-Initiated session. It is defined separately because it is a very important benchmark for most SIP installations. An example demonstrating its use is an

avalanche restart, where hundreds of thousands of end points register simultaneously following a power outage. In such a case, an authoritative measurement of the capacity of the device to register endpoints is useful to the network designer. Additionally, in certain controlled networks, there appears to be a difference between the registration rate of new endpoints and the registering rate of existing endpoints (register refreshes). This benchmark can capture these differences as well.

Measurement Units:

registrations per second (rps)

Issues:

None.

See Also:

None.

3.4.3. Registration Attempt Rate

Definition:

Configuration of the EA for the number of registrations per second that the EA attempts to send to the DUT.

Discussion:

The Registration Attempt Rate is the number of registration requests per second that the EA sends toward the DUT.

Measurement Units:

Registrations per second (rps)

Issues:

None.

See Also: Non-Invite-Initiated Session

4. IANA Considerations

This document requires no IANA considerations.

5. Security Considerations

Documents of this type do not directly affect the security of Internet or corporate networks as long as benchmarking is not performed on devices or systems connected to production networks. Security threats and how to counter these in SIP and the media layer

is discussed in RFC3261 [RFC3261], RFC 3550 [RFC3550] and RFC3711 [RFC3711]. This document attempts to formalize a set of common terminology for benchmarking SIP networks. Packets with unintended and/or unauthorized DSCP or IP precedence values may present security issues. Determining the security consequences of such packets is out of scope for this document.

6. Acknowledgments

The authors would like to thank Keith Drage, Cullen Jennings, Daryl Malas, Al Morton, and Henning Schulzrinne for invaluable contributions to this document. Dale Worley provided an extensive review that lead to improvements in the documents. We are grateful to Barry Constantine, William Cervený and Robert Sparks for providing valuable comments during the document's last calls and expert reviews. Al Morton and Sarah Banks have been exemplary working group chairs, we thank them for tracking this work to completion.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [RFC5393] Sparks, R., Lawrence, S., Hawrylyshen, A., and B. Campen, "Addressing an Amplification Vulnerability in Session Initiation Protocol (SIP) Forking Proxies", RFC 5393, December 2008.
- [RFC4320] Sparks, R., "Actions Addressing Identified Issues with the Session Initiation Protocol's (SIP) Non-INVITE Transaction", RFC 4320, January 2006.
- [RFC6026] Sparks, R. and T. Zourzouvillys, "Correct Transaction Handling for 2xx Responses to Session Initiation Protocol (SIP) INVITE Requests", RFC 6026, September 2010.

[I-D.ietf-bmwg-sip-bench-meth]

Dauids, C., Gurbani, V., and S. Poretsky, "SIP Performance Benchmarking Methodology",
draft-ietf-bmwg-sip-bench-meth-10 (work in progress),
May 2014.

7.2. Informational References

- [RFC2285] Mandeville, R., "Benchmarking Terminology for LAN Switching Devices", RFC 2285, February 1998.
- [RFC1242] Bradner, S., "Benchmarking terminology for network interconnection devices", RFC 1242, July 1991.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.
- [RFC6357] Hilt, V., Noel, E., Shen, C., and A. Abdelal, "Design Considerations for Session Initiation Protocol (SIP) Overload Control", RFC 6357, August 2011.
- [RFC7118] Baz Castillo, I., Millan Villegas, J., and V. Pascual, "The WebSocket Protocol as a Transport for the Session Initiation Protocol (SIP)", RFC 7118, January 2014.

Authors' Addresses

Carol Davids
Illinois Institute of Technology
201 East Loop Road
Wheaton, IL 60187
USA

Phone: +1 630 682 6024
Email: davids@iit.edu

Vijay K. Gurbani
Bell Laboratories, Alcatel-Lucent
1960 Lucent Lane
Rm 9C-533
Naperville, IL 60566
USA

Phone: +1 630 224 0216
Email: vkg@bell-labs.com

Scott Poretsky
Allot Communications
300 TradeCenter, Suite 4680
Woburn, MA 08101
USA

Phone: +1 508 309 2179
Email: sporetsky@allot.com

Network Working Group
Internet Draft
Intended status: Informational
Expires: January 2015
August 10, 2014

B. Constantine
JDSU
T. Copley
Level-3
R. Krishnan
Brocade Communications

Traffic Management Benchmarking
draft-ietf-bmwg-traffic-management-00.txt

Status of this Memo

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

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

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

This Internet-Draft will expire on January 10, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Abstract

This framework describes a practical methodology for benchmarking the traffic management capabilities of networking devices (i.e. policing, shaping, etc.). The goal is to provide a repeatable test method that objectively compares performance of the device's traffic management capabilities and to specify the means to benchmark traffic management with representative application traffic.

Table of Contents

1. Introduction.....	4
1.1. Traffic Management Overview.....	4
1.2. DUT Lab Configuration and Testing Overview.....	5
2. Conventions used in this document.....	7
3. Scope and Goals.....	8
4. Traffic Benchmarking Metrics.....	9
4.1. Metrics for Stateless Traffic Tests.....	9
4.2. Metrics for Stateful Traffic Tests.....	11
5. Tester Capabilities.....	11
5.1. Stateless Test Traffic Generation.....	11
5.2. Stateful Test Pattern Generation.....	12
5.2.1. TCP Test Pattern Definitions.....	13
6. Traffic Benchmarking Methodology.....	15
6.1. Policing Tests.....	15
6.1.1 Policer Individual Tests.....	15
6.1.2 Policer Capacity Tests.....	16
6.1.2.1 Maximum Policers on Single Physical Port.....	16
6.1.2.2 Single Policer on All Physical Ports.....	17
6.1.2.3 Maximum Policers on All Physical Ports.....	17
6.2. Queue/Scheduler Tests.....	17
6.2.1 Queue/Scheduler Individual Tests.....	17
6.2.1.1 Testing Queue/Scheduler with Stateless Traffic....	17
6.2.1.2 Testing Queue/Scheduler with Stateful Traffic....	18
6.2.2 Queue / Scheduler Capacity Tests.....	19
6.2.2.1 Multiple Queues / Single Port Active.....	19
6.2.2.1.1 Strict Priority on Egress Port.....	19
6.2.2.1.2 Strict Priority + Weighted Fair Queue (WFQ)....	19
6.2.2.2 Single Queue per Port / All Ports Active.....	19
6.2.2.3 Multiple Queues per Port, All Ports Active.....	20
6.3. Shaper tests.....	20
6.3.1 Shaper Individual Tests.....	20
6.3.1.1 Testing Shaper with Stateless Traffic.....	20
6.3.1.2 Testing Shaper with Stateful Traffic.....	21
6.3.2 Shaper Capacity Tests.....	22
6.3.2.1 Single Queue Shaped, All Physical Ports Active....	22
6.3.2.2 All Queues Shaped, Single Port Active.....	22
6.3.2.3 All Queues Shaped, All Ports Active.....	22
6.4. Concurrent Capacity Load Tests.....	24
7. Security Considerations.....	24
8. IANA Considerations.....	24
9. Conclusions.....	24
10. References.....	24
10.1. Normative References.....	25
10.2. Informative References.....	25
11. Acknowledgments.....	25

1. Introduction

Traffic management (i.e. policing, shaping, etc.) is an increasingly important component when implementing network Quality of Service (QoS). There is currently no framework to benchmark these features although some standards address specific areas. This draft provides a framework to conduct repeatable traffic management benchmarks for devices and systems in a lab environment.

Specifically, this framework defines the methods to characterize the capacity of the following traffic management features in network devices; classification, policing, queuing / scheduling, and traffic shaping.

This benchmarking framework can also be used as a test procedure to assist in the tuning of traffic management parameters before service activation. In addition to Layer 2/3 benchmarking, Layer 4 test patterns are proposed by this draft in order to more realistically benchmark end-user traffic.

1.1. Traffic Management Overview

In general, a device with traffic management capabilities performs the following functions:

- Traffic classification: identifies traffic according to various configuration rules (i.e. VLAN, DSCP, etc.) and marks this traffic internally to the network device. Multiple external priorities (DSCP, 802.1p, etc.) can map to the same priority in the device.
- Traffic policing: limits the rate of traffic that enters a network device according to the traffic classification. If the traffic exceeds the contracted limits, the traffic is either dropped or remarked and sent onto to the next network device
- Traffic Scheduling: provides traffic classification within the network device by directing packets to various types of queues and applies a dispatching algorithm to assign the forwarding sequence of packets
- Traffic shaping: a traffic control technique that actively buffers and meters the output rate in an attempt to adapt bursty traffic to the configured limits
- Active Queue Management (AQM): monitors the status of internal queues and actively drops (or re-marks) packets, which causes hosts using congestion-aware protocols to back-off and in turn can alleviate queue congestion. Note that AQM is outside of the scope of this testing framework.

The following diagram is a generic model of the traffic management capabilities within a network device. It is not intended to represent all variations of manufacturer traffic management capabilities, but provide context to this test framework.

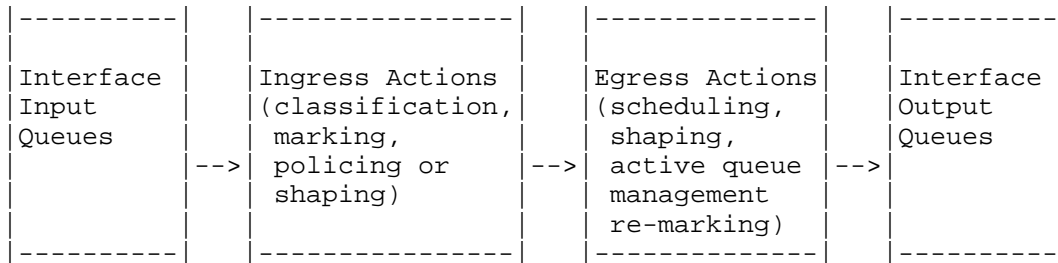


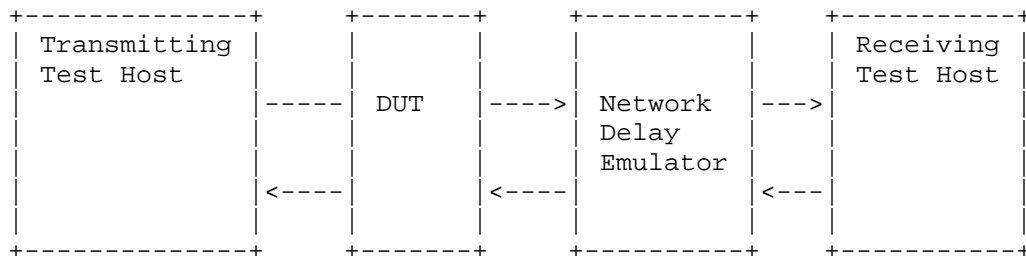
Figure 1: Generic Traffic Management capabilities of a Network Device

Ingress actions such as classification are defined in RFC 4689 and include IP addresses, port numbers, DSCP, etc. In terms of marking, RFC 2697 and RFC 2698 define a single rate and dual rate, three color marker, respectively.

The MEF specifies policing and shaping in terms of Ingress and Egress Subscriber/Provider Conditioning Functions in MEF12.1; Ingress and Bandwidth Profile attributes in MEF 10.2 and MEF 26.

1.2 DUT Lab Configuration and Testing Overview

The following is the description of the lab set-up for the traffic management tests:



As shown in the test diagram, the framework supports uni-directional and bi-directional traffic management tests.

This testing framework describes the tests and metrics for each of the following traffic management functions:

- Policing
- Queuing / Scheduling
- Shaping

The tests are divided into individual tests and rated capacity tests. The individual tests are intended to benchmark the traffic management functions according to the metrics defined in Section 4. The capacity tests verify traffic management functions under full load. This involves concurrent testing of multiple interfaces with the specific traffic management function enabled, and doing so to the capacity limit of each interface.

As an example: a device is specified to be capable of shaping on all of its egress ports. The individual test would first be conducted to benchmark the advertised shaping function against the metrics defined in section 4. Then the capacity test would be executed to test the shaping function concurrently on all interfaces and with maximum traffic load.

The Network Delay Emulator (NDE) is a requirement for the TCP stateful tests, which require network delay to allow TCP to fully open the TCP window. Also note that the Network Delay Emulator (NDE) should be passive in nature such as a fiber spool. This is recommended to eliminate the potential effects that an active delay element (i.e. test impairment generator) may have on the test flows. In the case that a fiber spool is not practical due to the desired latency, an active NDE must be independently verified to be capable of adding the configured delay without loss. In other words, the DUT would be removed and the NDE performance benchmarked independently.

Note the NDE should be used in "full pipe" delay mode. Most NDEs allow for per flow delay actions, emulating QoS prioritization. For this framework, the NDE's sole purpose is simply to add delay to all packets (emulate network latency). So to benchmark the performance of the NDE, maximum offered load should be tested against the following frame sizes: 128, 256, 512, 768, 1024, 1500, and 9600 bytes. The delay accuracy at each of these packet sizes can then be used to calibrate the range of expected BDPs for the TCP stateful tests.

2. Conventions used in this document

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

The following acronyms are used:

BB: Bottleneck Bandwidth

BDP: Bandwidth Delay Product

BSA: Burst Size Achieved

CBS: Committed Burst Size

CIR: Committed Information Rate

DUT: Device Under Test

EBS: Excess Burst Size

EIR: Excess Information Rate

NDE: Network Delay Emulator

SP: Strict Priority Queuing

QL: Queue Length

QoS: Quality of Service

RED: Random Early Discard

RTT: Round Trip Time

SBB: Shaper Burst Bytes

SBI: Shaper Burst Interval

SR: Shaper Rate

SSB: Send Socket Buffer

Tc: CBS Time Interval

Te: EBS Time Interval

Ti Transmission Interval

TTP: TCP Test Pattern

TTPET: TCP Test Pattern Execution Time

WRED: Weighted Random Early Discard

3. Scope and Goals

The scope of this work is to develop a framework for benchmarking and testing the traffic management capabilities of network devices in the lab environment. These network devices may include but are not limited to:

- Switches (including Layer 2/3 devices)
- Routers
- Firewalls
- General Layer 4-7 appliances (Proxies, WAN Accelerators, etc.)

Essentially, any network device that performs traffic management as defined in section 1.1 can be benchmarked or tested with this framework.

The primary goal is to assess the maximum forwarding performance that a network device can sustain without dropping or impairing packets, or compromising the accuracy of multiple instances of traffic management functions. This is the benchmark for comparison between devices.

Within this framework, the metrics are defined for each traffic management test but do not include pass / fail criterion, which is not within the charter of BMWG. This framework provides the test methods and metrics to conduct repeatable testing, which will provide the means to compare measured performance between DUTs.

As mentioned in section 1.2, this framework describes the individual tests and metrics for several management functions. It is also within scope that this framework will benchmark each function in terms of overall rated capacity. This involves concurrent testing of multiple interfaces with the specific traffic management function enabled, up to the capacity limit of each interface.

It is not within scope of this framework to specify the procedure for testing multiple traffic management functions concurrently. The multitudes of possible combinations is almost unbounded and the ability to identify functional "break points" would be most times impossible.

However, section 6.4 provides suggestions for some profiles of concurrent functions that would be useful to benchmark. The key requirement for any concurrent test function is that tests must produce reliable and repeatable results.

Also, it is not within scope to perform conformance testing. Tests defined in this framework benchmark the traffic management functions according to the metrics defined in section 4 and do not address any conformance to standards related to traffic management. Traffic management specifications largely do not exist and this is a prime driver for this framework; to provide an objective means to compare vendor traffic management functions.

Another goal is to devise methods that utilize flows with congestion-aware transport (TCP) as part of the traffic load and still produce repeatable results in the isolated test environment. This framework will derive stateful test patterns (TCP or application layer) that can also be used to further benchmark the performance of applicable traffic management techniques such as queuing / scheduling and traffic shaping. In cases where the network device is stateful in nature (i.e. firewall, etc.), stateful test pattern traffic is important to test along with stateless, UDP traffic in specific test scenarios (i.e. applications using TCP transport and UDP VoIP, etc.)

And finally, this framework will provide references to open source tools that can be used to provide stateless and/or stateful traffic generation emulation.

4. Traffic Benchmarking Metrics

The metrics to be measured during the benchmarks are divided into two (2) sections: packet layer metrics used for the stateless traffic testing and segment layer metrics used for the stateful traffic testing.

4.1. Metrics for Stateless Traffic Tests

For the stateless traffic tests, the metrics are defined at the layer 3 packet level versus layer 2 packet level for consistency.

Stateless traffic measurements require that sequence number and time-stamp be inserted into the payload for lost packet analysis. Delay analysis may be achieved by insertion of timestamps directly into the packets or timestamps stored elsewhere (packet captures). This framework does not specify the packet format to carry sequence number or timing information. However, RFC 4689 provides recommendations for sequence tracking along with definitions of in-sequence and out-of-order packets.

The following are the metrics to be used during the stateless traffic benchmarking components of the tests:

- Burst Size Achieved (BSA): for the traffic policing and network queue tests, the tester will be configured to send bursts to test either the Committed Burst Size (CBS) or Excess Burst Size (EBS) of a policer or the queue / buffer size configured in the DUT. The Burst Size Achieved metric is a measure of the actual burst size received at the egress port of the DUT with no lost packets. As an example, the configured CBS of a DUT is 64KB and after the burst test, only a 63 KB can be achieved without packet loss. Then 63KB is the BSA. Also, the average Packet Delay Variation (PDV see below) as experienced by the packets sent at the BSA burst size should be recorded.

- Lost Packets (LP): For all traffic management tests, the tester will transmit the test packets into the DUT ingress port and the number of packets received at the egress port will be measured. The difference between packets transmitted into the ingress port and received at the egress port is the number of lost packets as measured at the egress port. These packets must have unique identifiers such that only the test packets are measured. RFC 4737 and RFC 2680 describe the need to establish the time threshold to wait before a packet is declared as lost. packet as lost, and this threshold MUST be reported with the results.

- Out of Sequence (OOS): in additions to the LP metric, the test packets must be monitored for sequence and the out-of-sequence (OOS) packets. RFC 4689 defines the general function of sequence tracking, as well as definitions for in-sequence and out-of-order packets. Out-of-order packets will be counted per RFC 4737 and RFC 2680.

- Packet Delay (PD): the Packet Delay metric is the difference between the timestamp of the received egress port packets and the packets transmitted into the ingress port and specified in RFC 2285.

- Packet Delay Variation (PDV): the Packet Delay Variation metric is the variation between the timestamp of the received egress port packets and specified in RFC 5481.
- Shaper Rate (SR): the Shaper Rate is only applicable to the traffic shaping tests. The SR represents the average egress output rate (bps) over the test interval.
- Shaper Burst Bytes (SBB): the Shaper Burst Bytes is only applicable to the traffic shaping tests. A traffic shaper will emit packets in different size "trains" (bytes back-to-back). This metric characterizes the method by which the shaper emits traffic. Some shapers transmit larger bursts per interval, while other shapers may transmit a single frame at the CIR rate (two extreme examples).
- Shaper Burst Interval(SBI): the interval is only applicable to the traffic shaping tests and again is the time between a shaper emitted bursts.

4.2. Metrics for Stateful Traffic Tests

The stateful metrics will be based on RFC 6349 TCP metrics and will include:

- TCP Test Pattern Execution Time (TTPET): RFC 6349 defined the TCP Transfer Time for bulk transfers, which is simply the measured time to transfer bytes across single or concurrent TCP connections. The TCP test patterns used in traffic management tests will include bulk transfer and interactive applications. The interactive patterns include instances such as HTTP business applications, database applications, etc. The TTPET will be the measure of the time for a single execution of a TCP Test Pattern (TTP). Average, minimum, and maximum times will be measured or calculated.

An example would be an interactive HTTP TTP session which should take 5 seconds on a Gige network with 0.5 millisecond latency. During ten (10) executions of this TTP, the TTPET results might be: average of 6.5 seconds, minimum of 5.0 seconds, and maximum of 7.9 seconds.

- TCP Efficiency: after the execution of the TCP Test Pattern, TCP Efficiency represents the percentage of Bytes that were not retransmitted.

Transmitted Bytes - Retransmitted Bytes

TCP Efficiency % = ----- X 100

Transmitted Bytes

Transmitted Bytes are the total number of TCP Bytes to be transmitted including the original and the retransmitted Bytes. These retransmitted bytes should be recorded from the sender's TCP/IP stack perspective, to avoid any misinterpretation that a reordered packet is a retransmitted packet (as may be the case with packet decode interpretation).

- Buffer Delay: represents the increase in RTT during a TCP test versus the baseline DUT RTT (non congested, inherent latency). RTT and the technique to measure RTT (average versus baseline) are defined in RFC 6349. Referencing RFC 6349, the average RTT is derived from the total of all measured RTTs during the actual test sampled at every second divided by the test duration in seconds.

$$\text{Average RTT during transfer} = \frac{\text{Total RTTs during transfer}}{\text{Transfer duration in seconds}}$$

$$\text{Buffer Delay \%} = \frac{\text{Average RTT during Transfer} - \text{Baseline RTT}}{\text{Baseline RTT}} \times 100$$

Note that even though this was not explicitly stated in RFC 6349, retransmitted packets should not be used in RTT measurements.

Also, the test results should record the average RTT in millisecond across the entire test duration and number of samples.

5. Tester Capabilities

The testing capabilities of the traffic management test environment are divided into two (2) sections: stateless traffic testing and stateful traffic testing

5.1. Stateless Test Traffic Generation

The test set must be capable of generating traffic at up to the link speed of the DUT. The test set must be calibrated to verify that it will not drop any packets. The test set's inherent PD and PDV must also be calibrated and subtracted from the PD and PDV metrics. The test set must support the encapsulation to be tested such as VLAN, Q-in-Q, MPLS, etc. Also, the test set must allow control of the classification techniques defined in RFC 4689 (i.e. IP address, DSCP, TOS, etc classification).

The open source tool "iperf" can be used to generate stateless UDP traffic and is discussed in Appendix A. Since iperf is a software based tool, there will be performance limitations at higher link speeds (e.g. GigE, 10 GigE, etc.). Careful calibration of any test environment using iperf is important. At higher link speeds, it is recommended to use hardware based packet test equipment.

5.1.1 Burst Hunt with Stateless Traffic

A central theme for the traffic management tests is to benchmark the specified burst parameter of traffic management function, since burst parameters of SLAs are specified in bytes. For testing efficiency, it is recommended to include a burst hunt feature, which automates the manual process of determining the maximum burst size which can be supported by a traffic management function.

The burst hunt algorithm should start at the target burst size (maximum burst size supported by the traffic management function) and will send single bursts until it can determine the largest burst that can pass without loss. If the target burst size passes, then the test is complete. The hunt aspect occurs when the target burst size is not achieved; the algorithm will drop down to a configured minimum burst size and incrementally increase the burst until the maximum burst supported by the DUT is discovered. The recommended granularity of the incremental burst size increase is 1 KB.

Optionally for a policer function and if the burst size passes, the burst should be increased by increments of 1 KB to verify that the policer is truly configured properly (or enabled at all).

5.2. Stateful Test Pattern Generation

The TCP test host will have many of the same attributes as the TCP test host defined in RFC 6349. The TCP test device may be a standard computer or a dedicated communications test instrument. In both cases, it must be capable of emulating both a client and a server.

For any test using stateful TCP test traffic, the Network Delay Emulator (NDE function from the lab set-up diagram) must be used in order to provide a meaningful BDP. As referenced in section 2, the target traffic rate and configured RTT must be verified independently using just the NDE for all stateful tests (to ensure the NDE can delay without loss).

The TCP test host must be capable to generate and receive stateful TCP test traffic at the full link speed of the DUT. As a general rule of thumb, testing TCP Throughput at rates greater than 500 Mbps may require high performance server hardware or dedicated hardware based test tools.

The TCP test host must allow adjusting both Send and Receive Socket Buffer sizes. The Socket Buffers must be large enough to fill the BDP for bulk transfer TCP test application traffic.

Measuring RTT and retransmissions per connection will generally require a dedicated communications test instrument. In the absence of dedicated hardware based test tools, these measurements may need to be conducted with packet capture tools, i.e. conduct TCP Throughput tests and analyze RTT and retransmissions in packet captures.

The TCP implementation used by the test host must be specified in the test results (i.e. OS version, i.e. LINUX OS kernel using TCP New Reno, TCP options supported, etc.).

While RFC 6349 defined the means to conduct throughput tests of TCP bulk transfers, the traffic management framework will extend TCP test execution into interactive TCP application traffic. Examples include email, HTTP, business applications, etc. This interactive traffic is bi-directional and can be chatty.

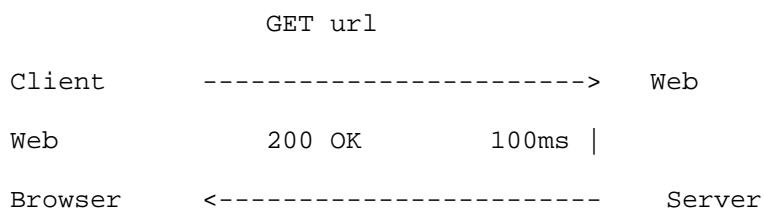
The test device must not only support bulk TCP transfer application traffic but also chatty traffic. A valid stress test SHOULD include both traffic types. This is due to the non-uniform, bursty nature of chatty applications versus the relatively uniform nature of bulk transfers (the bulk transfer smoothly stabilizes to equilibrium state under lossless conditions).

While iperf is an excellent choice for TCP bulk transfer testing, the open source tool "Flowgrind" (referenced in Appendix A) is client-server based and emulates interactive applications at the TCP layer. As with any software based tool, the performance must be qualified to the link speed to be tested. Hardware-based test equipment should be considered for reliable results at higher links speeds (e.g. 1 GigE, 10 GigE).

5.2.1. TCP Test Pattern Definitions

As mentioned in the goals of this framework, techniques are defined to specify TCP traffic test patterns to benchmark traffic management technique(s) and produce repeatable results. Some network devices such as firewalls, will not process stateless test traffic which is another reason why stateful TCP test traffic must be used.

An application could be fully emulated up to Layer 7, however this framework proposes that stateful TCP test patterns be used in order to provide granular and repeatable control for the benchmarks. The following diagram illustrates a simple Web Browsing application (HTTP).



In this example, the Client Web Browser (Client) requests a URL and then the Web Server delivers the web page content to the Client (after a Server delay of 100 millisecond). This asynchronous, "request/response" behavior is intrinsic to most TCP based applications such as Email (SMTP), File Transfers (FTP and SMB), Database (SQL), Web Applications (SOAP), REST, etc. The impact to the network elements is due to the multitudes of Clients and the variety of bursty traffic, which stresses traffic management functions. The actual emulation of the specific application protocols is not required and TCP test patterns can be defined to mimic the application network traffic flows and produce repeatable results.

There are two (2) techniques recommended by this framework to develop standard TCP test patterns for traffic management benchmarking.

The first technique involves modeling, which have been described in "3GPP2 C.R1002-0 v1.0" and describe the behavior of HTTP, FTP, and WAP applications at the TCP layer. The models have been defined with various mathematical distributions for the Request/Response bytes and inter-request gap times. The Flowgrind tool (Appendix A) supports many of the distributions and is a good choice as long as the processing limits of the server platform are taken into consideration.

The second technique is to conduct packet captures of the applications to test and then to statefully play the application back at the TCP layer. The TCP playback includes the request byte size, response byte size, and inter-message gaps at both the client and the server. The advantage of this method is that very realistic test patterns can be defined based on real world application traffic.

This framework does not specify a fixed set of TCP test patterns, but does provide recommended test cases in Appendix B. Some of these examples reflect those specified in "draft-ietf-bmwg-ca-bench-meth-04" which suggests traffic mixes for a variety of representative application profiles. Other examples are simply well known application traffic types.

6. Traffic Benchmarking Methodology

The traffic benchmarking methodology uses the test set-up from section 2 and metrics defined in section 4. Each test should be run for a minimum test time of 5 minutes.

Each test should compare the network device's internal statistics (available via command line management interface, SNMP, etc.) to the measured metrics defined in section 4. This evaluates the accuracy of the internal traffic management counters under individual test conditions and capacity test conditions that are defined in each subsection.

6.1. Policing Tests

The intent of the policing tests is to verify the policer performance (i.e. CIR-CBS and EIR-EBS parameters). The tests will verify that the network device can handle the CIR with CBS and the EIR with EBS and will use back-back packet testing concepts from RFC 2544 (but adapted to burst size algorithms and terminology). Also MEF-14,19,37 provide some basis for specific components of this test. The burst hunt algorithm defined in section 5.1.1 can also be used to automate the measurement of the CBS value.

The tests are divided into two (2) sections; individual policer tests and then full capacity policing tests. It is important to benchmark the basic functionality of the individual policer then proceed into the fully rated capacity of the device. This capacity may include the number of policing policies per device and the number of policers simultaneously active across all ports.

6.1.1 Policer Individual Tests

Policing tests should use stateless traffic. Stateful TCP test traffic will generally be adversely affected by a policer in the absence of traffic shaping. So while TCP traffic could be used, it is more accurate to benchmark a policer with stateless traffic.

The policer test shall test a policer as defined by RFC 4115 or MEF 10.2, depending upon the equipment's specification. As an example for RFC 4115, consider a CBS and EBS of 64KB and CIR and EIR of 100 Mbps on a 1GigE physical link (in color-blind mode). A stateless traffic burst of 64KB would be sent into the policer at the GigE rate. This equates to approximately a 0.512 millisecond burst time (64 KB at 1 GigE). The traffic generator must space these bursts to ensure that the aggregate throughput does not exceed the CIR. The Ti between the bursts would equal $CBS * 8 / CIR = 5.12$ millisecond in this example.

The metrics defined in section 4.1 shall be measured at the egress port and recorded.

In addition to verifying that the policer allows the specified CBS and EBS bursts to pass, the policer test must verify that the policer will police at the specified CBS/EBS values.

For this portion of the test, the CBS/EBS value should be incremented by 1000 bytes higher than the configured CBS and that the egress port measurements must show that the excess packets are dropped.

Additional tests beyond the simple color-blind example might include: color-aware mode, configurations where EIR is greater than CIR, etc.

6.1.2 Policer Capacity Tests

The intent of the capacity tests is to verify the policer performance in a scaled environment with multiple ingress customer policers on multiple physical ports. This test will benchmark the maximum number of active policers as specified by the device manufacturer.

As an example, a Layer 2 switching device may specify that each of the 32 physical ports can be policed using a pool of policing service policies. The device may carry a single customer's traffic on each physical port and a single policer is instantiated per physical port. Another possibility is that a single physical port may carry multiple customers, in which case many customer flows would be policed concurrently on an individual physical port (separate policers per customer on an individual port).

The specified policing function capacity is generally expressed in terms of the number of policers active on each individual physical port as well as the number of unique policer rates that are utilized. For all of the capacity tests, the benchmarking methodology described in Section 6.1.1 for a single policer should be applied to each of the physical port policers.

6.1.2.1 Maximum Policers on Single Physical Port

The first policer capacity test will benchmark a single physical port, maximum policers on that physical port.

Assume multiple categories of ingress policers at rates r_1, r_2, \dots, r_n . There are multiple customers on a single physical port. Each customer could be represented by a single tagged vlan, double tagged vlan, VPLS instance etc. Each customer is mapped to a different policer. Each of the policers can be of rates r_1, r_2, \dots, r_n .

An example configuration would be

- Y1 customers, policer rate r_1
- Y2 customers, policer rate r_2
- Y3 customers, policer rate r_3
- ...
- Yn customers, policer rate r_n

Some bandwidth on the physical port is dedicated for other traffic (non customer traffic); this includes network control protocol traffic. There is a separate policer for the other traffic. Typical deployments have 3 categories of policers; there may be some deployments with more or less than 3 categories of ingress policers.

6.1.2.2 Single Policer on All Physical Ports

The second policer capacity test involves a single Policer function per physical port with all physical ports active. In this test, there is a single policer per physical port. The policer can have one of the rates r_1, r_2, \dots, r_n . All the physical ports in the networking device are active.

6.1.2.3 Maximum Policers on All Physical Ports

Finally the third policer capacity test involves a combination of the first and second capacity test, namely maximum policers active per physical port and all physical ports are active .

6.2. Queue and Scheduler Tests

Queues and traffic Scheduling are closely related in that a queue's priority dictates the manner in which the traffic scheduler's transmits packets out of the egress port.

Since device queues / buffers are generally an egress function, this test framework will discuss testing at the egress (although the technique can be applied to ingress side queues).

Similar to the policing tests, the tests are divided into two sections; individual queue/scheduler function tests and then full capacity tests.

6.2.1 Queue/Scheduler Individual Tests

The various types of scheduling techniques include FIFO, Strict Priority (SP), Weighted Fair Queueing (WFQ) along with other variations. This test framework recommends to test at a minimum of three techniques although it is the discretion of the tester to benchmark other device scheduling algorithms.

6.2.1.1 Testing Queue/Scheduler with Stateless Traffic

A network device queue is memory based unlike a policing function, which is token or credit based. However, the same concepts from section 6.1 can be applied to testing network device queues.

The device's network queue should be configured to the desired size in KB (queue length, QL) and then stateless traffic should be transmitted to test this QL.

A queue should be able to handle repetitive bursts with the transmission gaps proportional to the bottleneck bandwidth. This gap is referred to as the transmission interval (Ti). Ti can be defined for the traffic bursts and is based off of the QL and Bottleneck Bandwidth (BB) of the egress interface.

$$Ti = QL * 8 / BB$$

Note that this equation is similar to the Ti required for transmission into a policer (QL = CBS, BB = CIR). Also note that the burst hunt algorithm defined in section 5.1.1 can also be used to automate the measurement of the queue value.

The stateless traffic burst shall be transmitted at the link speed and spaced within the T_i time interval. The metrics defined in section 4.1 shall be measured at the egress port and recorded; the primary result is to verify the BSA and that no packets are dropped.

The scheduling function must also be characterized to benchmark the device's ability to schedule the queues according to the priority. An example would be 2 levels of priority including SP and FIFO queueing. Under a flow load greater the egress port speed, the higher priority packets should be transmitted without drops (and also maintain low latency), while the lower priority (or best effort) queue may be dropped.

6.2.1.2 Testing Queue/Scheduler with Stateful Traffic

To provide a more realistic benchmark and to test queues in layer 4 devices such as firewalls, stateful traffic testing is recommended for the queue tests. Stateful traffic tests will also utilize the Network Delay Emulator (NDE) from the network set-up configuration in section 2.

The BDP of the TCP test traffic must be calibrated to the QL of the device queue. Referencing RFC 6349, the BDP is equal to:

$BB * RTT / 8$ (in bytes)

The NDE must be configured to an RTT value which is large enough to allow the BDP to be greater than QL. An example test scenario is defined below:

- Ingress link = GigE
- Egress link = 100 Mbps (BB)
- QL = 32KB

$RTT(min) = QL * 8 / BB$ and would equal 2.56 millisecond (and the BDP = 32KB)

In this example, one (1) TCP connection with window size / SSB of 32KB would be required to test the QL of 32KB. This Bulk Transfer Test can be accomplished using iperf as described in Appendix A.

Two types of TCP tests must be performed: Bulk Transfer test and Micro Burst Test Pattern as documented in Appendix B. The Bulk Transfer Test only bursts during the TCP Slow Start (or Congestion Avoidance) state, while the Micro Burst test emulates application layer bursting which may occur any time during the TCP connection.

Other tests types should include: Simple Web Site, Complex Web Site, Business Applications, Email, SMB/CIFS File Copy (which are also documented in Appendix B).

The test results will be recorded per the stateful metrics defined in section 4.2, primarily the TCP Test Pattern Execution Time (TTPET), TCP Efficiency, and Buffer Delay.

6.2.2 Queue / Scheduler Capacity Tests

The intent of these capacity tests is to benchmark queue/scheduler performance in a scaled environment with multiple queues/schedulers active on multiple egress physical ports. This test will benchmark the maximum number of queues and schedulers as specified by the device manufacturer. Each priority in the system will map to a separate queue.

6.2.2.1 Multiple Queues / Single Port Active

For the first scheduler / queue capacity test, multiple queues per port will be tested on a single physical port. In this case, all the queues (typically 8) are active on a single physical port. Traffic from multiple ingress physical ports are directed to the same egress physical port which will cause oversubscription on the egress physical port.

There are many types of priority schemes and combinations of priorities that are managed by the scheduler. The following sections specify the priority schemes that should be tested.

6.2.2.1.1 Strict Priority on Egress Port

For this test, Strict Priority (SP) scheduling on the egress physical port should be tested and the benchmarking methodology specified in section 6.2.1 should be applied here. For a given priority, each ingress physical port should get a fair share of the egress physical port bandwidth.

6.2.2.1.2 Strict Priority + Weighted Fair Queue (WFQ) on Egress Port

For this test, Strict Priority (SP) and Weighted Fair Queue (WFQ) should be enabled simultaneously in the scheduler but on a single egress port. The benchmarking methodology specified in Section 6.2.1 should be applied here. Additionally, the egress port bandwidth sharing among weighted queues should be proportional to the assigned weights. For a given priority, each ingress physical port should get a fair share of the egress physical port bandwidth.

6.2.2.2 Single Queue per Port / All Ports Active

Traffic from multiple ingress physical ports are directed to the same egress physical port, which will cause oversubscription on the egress physical port. Also, the same amount of traffic is directed to each egress physical port.

The benchmarking methodology specified in Section 6.2.1 should be applied here. Each ingress physical port should get a fair share of the egress physical port bandwidth. Additionally, each egress physical port should receive the same amount of traffic.

6.2.2.3 Multiple Queues per Port, All Ports Active

Traffic from multiple ingress physical ports are directed to all queues of each egress physical port, which will cause oversubscription on the egress physical ports. Also, the same amount of traffic is directed to each egress physical port.

The benchmarking methodology specified in Section 6.2.1 should be applied here. For a given priority, each ingress physical port should get a fair share of the egress physical port bandwidth. Additionally, each egress physical port should receive the same amount of traffic.

6.3. Shaper tests

A traffic shaper is memory based like a queue, but with the added intelligence of an active shaping element. The same concepts from section 6.2 (Queue testing) can be applied to testing network device shaper.

Again, the tests are divided into two sections; individual shaper benchmark tests and then full capacity shaper benchmark tests.

6.3.1 Shaper Individual Tests

A traffic shaper generally has three (3) components that can be configured:

- Ingress Queue bytes
- Shaper Rate, bps
- Burst Committed (Bc) and Burst Excess (Be), bytes

The Ingress Queue holds burst traffic and the shaper then meters traffic out of the egress port according to the Shaper Rate and Bc/Be parameters. Shapers generally transmit into policers, so the idea is for the emitted traffic to conform to the policer's limits.

The stateless and stateful traffic test sections describe the techniques to transmit bursts into the DUT's ingress port and the metrics to benchmark at the shaper egress port.

6.3.1.1 Testing Shaper with Stateless Traffic

The stateless traffic must be burst into the DUT ingress port and not exceed the Ingress Queue. The burst can be a single burst or multiple bursts. If multiple bursts are transmitted, then the Ti (Time interval) must be large enough so that the Shaper Rate is not exceeded. An example will clarify single and multiple burst test cases.

In the example, the shaper's ingress and egress ports are both full duplex Gigabit Ethernet. The Ingress Queue is configured to be 512,000 bytes, the Shaper Rate = 50 Mbps, and both Bc/Be configured to be 32,000 bytes. For a single burst test, the transmitting test device would burst 512,000 bytes maximum into the ingress port and then stop transmitting. The egress port metrics from section 4.1 will be recorded with particular emphasis on the LP, PDV, SBB, and SBI metrics.

If a multiple burst test is to be conducted, then the burst bytes divided by the time interval between the 512,000 byte bursts must not exceed the Shaper Rate. The time interval (Ti) must adhere to a similar formula as described in section 6.2.1.1 for queues, namely:

$$Ti = \text{Ingress Queue} \times 8 / \text{Shaper Rate}$$

So for the example from the previous paragraph, Ti between bursts must be greater than 82 millisecond (512,000 bytes x 8 / 50,000,000 bps). This yields an average rate of 50 Mbps so that an Input Queue would not overflow.

6.3.1.2 Testing Shaper with Stateful Traffic

To provide a more realistic benchmark and to test queues in layer 4 devices such as firewalls, stateful traffic testing is also recommended for the shaper tests. Stateful traffic tests will also utilize the Network Delay Emulator (NDE) from the network set-up configuration in section 2.

The BDP of the TCP test traffic must be calculated as described in section 6.2.2. To properly stress network buffers and the traffic shaping function, the cumulative TCP window should exceed the BDP which will stress the shaper. BDP factors of 1.1 to 1.5 are recommended, but the values are the discretion of the tester and should be documented.

The cumulative TCP Window Sizes* (RWND at the receiving end & CWND at the transmitting end) equates to:

TCP window size* for each connection x number of connections

* as described in section 3 of RFC6349, the SSB MUST be large enough to fill the BDP

Example, if the BDP is equal to 256 Kbytes and a connection size of 64Kbytes is used for each connection, then it would require four (4) connections to fill the BDP and 5-6 connections (over subscribe the BDP) to stress test the traffic shaping function.

Two types of TCP tests must be performed: Bulk Transfer test and Micro Burst Test Pattern as documented in Appendix B. The Bulk Transfer Test only bursts during the TCP Slow Start (or Congestion Avoidance) state, while the Micro Burst test emulates application layer bursting which may any time during the TCP connection.

Other tests types should include: Simple Web Site, Complex Web Site, Business Applications, Email, SMB/CIFS File Copy (which are also documented in Appendix B).

The test results will be recorded per the stateful metrics defined in section 4.2, primarily the TCP Test Pattern Execution Time (TTPET), TCP Efficiency, and Buffer Delay.

6.3.2 Shaper Capacity Tests

The intent of these scalability tests is to verify shaper performance in a scaled environment with shapers active on multiple queues on multiple egress physical ports. This test will benchmark the maximum number of shapers as specified by the device manufacturer.

For all of the capacity tests, the benchmarking methodology described in Section 6.3.1 for a single shaper should be applied to each of the physical port and/or queue shapers.

6.3.2.1 Single Queue Shaped, All Physical Ports Active

The first shaper capacity test involves per port shaping, all physical ports active. Traffic from multiple ingress physical ports are directed to the same egress physical port and this will cause oversubscription on the egress physical port. Also, the same amount of traffic is directed to each egress physical port.

The benchmarking methodology described in Section 6.3.1 should be applied to each of the physical ports. Each ingress physical port should get a fair share of the egress physical port bandwidth.

6.3.2.2 All Queues Shaped, Single Port Active

The second shaper capacity test is conducted with all queues actively shaping on a single physical port. The benchmarking methodology described in per port shaping test (previous section) serves as the foundation for this. Additionally, each of the SP queues on the egress physical port is configured with a shaper. For the highest priority queue, the maximum amount of bandwidth available is limited by the bandwidth of the shaper. For the lower priority queues, the maximum amount of bandwidth available is limited by the bandwidth of the shaper and traffic in higher priority queues.

6.3.2.3 All Queues Shaped, All Ports Active

And for the third shaper capacity test (which is a combination of the tests in the previous two sections), all queues will be actively shaping and all physical ports active.

6.4 Concurrent Capacity Load Tests

As mentioned in the scope of this document, it is impossible to specify the various permutations of concurrent traffic management functions that should be tested in a device for capacity testing. However, some profiles are listed below which may be useful to test under capacity as well:

- Policers on ingress and queuing on egress
- Policers on ingress and shapers on egress (not intended for a flow to be policed then shaped, these would be two different flows tested at the same time)
- etc.

Appendix A: Open Source Tools for Traffic Management Testing

This framework specifies that stateless and stateful behaviors should both be tested. Two (2) open source tools that can be used are iperf and Flowgrind to accomplish many of the tests proposed in this framework.

Iperf can generate UDP or TCP based traffic; a client and server must both run the iperf software in the same traffic mode. The server is set up to listen and then the test traffic is controlled from the client. Both uni-directional and bi-directional concurrent testing are supported.

The UDP mode can be used for the stateless traffic testing. The target bandwidth, packet size, UDP port, and test duration can be controlled. A report of bytes transmitted, packets lost, and delay variation are provided by the iperf receiver.

The TCP mode can be used for stateful traffic testing to test bulk transfer traffic. The TCP Window size (which is actually the SSB), the number of connections, the packet size, TCP port and the test duration can be controlled. A report of bytes transmitted and throughput achieved are provided by the iperf sender.

Flowgrind is a distributed network performance measurement tool. Using the flowgrind controller, tests can be setup between hosts running flowgrind. For the purposes of this traffic management testing framework, the key benefit of Flowgrind is that it can emulate non-bulk transfer applications such as HTTP, Email, etc. This is due to fact that Flowgrind supports the concept of request and response behavior while iperf does not.

Traffic generation options include the request size, response size, inter-request gap, and response time gap. Additionally, various distribution types are supported including constant, normal, exponential, pareto, etc. These traffic generation parameters facilitate the emulation of some of the TCP test patterns which are discussed in Appendix B.

Since these tools are software based, the host hardware must be qualified as capable of generating the target traffic loads without packet loss and within the packet delay variation threshold.

Appendix B: Stateful TCP Test Patterns

This framework recommends at a minimum the following TCP test patterns since they are representative of real world application traffic (section 5.2.1 describes some methods to derive other application-based TCP test patterns).

- Bulk Transfer: generate concurrent TCP connections whose aggregate number of in-flight data bytes would fill the BDP. Guidelines from RFC 6349 are used to create this TCP traffic pattern.
- Micro Burst: generate precise burst patterns within a single or multiple TCP connections(s). The idea is for TCP to establish equilibrium and then burst application bytes at defined sizes. The test tool must allow the burst size and burst time interval to be configurable.
- Web Site Patterns: The HTTP traffic model from "3GPP2 C.R1002-0 v1.0" is referenced (Table 4.1.3.2-1) to develop these TCP test patterns. In summary, the HTTP traffic model consists of the following parameters:
 - Main object size (Sm)
 - Embedded object size (Se)
 - Number of embedded objects per page (Nd)
 - Client processing time (T_{cp})
 - Server processing time (T_{sp})

Web site test patterns are illustrated with the following examples:

- Simple Web Site: mimic the request / response and object download behavior of a basic web site (small company).
- Complex Web Site: mimic the request / response and object download behavior of a complex web site (ecommerce site).

Referencing the HTTP traffic model parameters , the following table was derived (by analysis and experimentation) for Simple and Complex Web site TCP test patterns:

Parameter	Simple Web Site	Complex Web Site

Main object size (Sm)	Ave. = 10KB Min. = 100B Max. = 500KB	Ave. = 300KB Min. = 50KB Max. = 2MB
Embedded object size (Se)	Ave. = 7KB Min. = 50B Max. = 350KB	Ave. = 10KB Min. = 100B Max. = 1MB
Number of embedded objects per page (Nd)	Ave. = 5 Min. = 2 Max. = 10	Ave. = 25 Min. = 10 Max. = 50
Client processing time (T _{cp})*	Ave. = 3s Min. = 1s Max. = 10s	Ave. = 10s Min. = 3s Max. = 30s
Server processing time (T _{sp})*	Ave. = 5s Min. = 1s Max. = 15s	Ave. = 8s Min. = 2s Max. = 30s

* The client and server processing time is distributed across the

transmission / receipt of all of the main and embedded objects

Constantine

August 10, 2014

[Page 24]

To be clear, the parameters in this table are reasonable guidelines for the TCP test pattern traffic generation. The test tool can use fixed parameters for simpler tests and mathematical distributions for more complex tests. However, the test pattern must be repeatable to ensure that the benchmark results can be reliably compared.

- Inter-active Patterns: While Web site patterns are inter-active to a degree, they mainly emulate the downloading of various complexity web sites. Inter-active patterns are more chatty in nature since there is a lot of user interaction with the servers. Examples include business applications such as Peoplesoft, Oracle and consumer applications such as Facebook, IM, etc. For the inter-active patterns, the packet capture technique was used to characterize some business applications and also the email application.

In summary, an inter-active application can be described by the following parameters:

- Client message size (Scm)
- Number of Client messages (Nc)
- Server response size (Srs)
- Number of server messages (Ns)
- Client processing time (Tcp)
- Server processing Time (Tsp)
- File size upload (Su)*
- File size download (Sd)*

* The file size parameters account for attachments uploaded or downloaded and may not be present in all inter-active applications

Again using packet capture as a means to characterize, the following table reflects the guidelines for Simple Business Application, Complex Business Application, eCommerce, and Email Send / Receive:

Parameter	Simple Biz. App.	Complex Biz. App.	eCommerce*	Email
Client message size (Scm)	Ave. = 450B Min. = 100B Max. = 1.5KB	Ave. = 2KB Min. = 500B Max. = 100KB	Ave. = 1KB Min. = 100B Max. = 50KB	Ave. = 200B Min. = 100B Max. = 1KB
Number of client messages (Nc)	Ave. = 10 Min. = 5 Max. = 25	Ave. = 100 Min. = 50 Max. = 250	Ave. = 20 Min. = 10 Max. = 100	Ave. = 10 Min. = 5 Max. = 25
Client processing time (Tcp)**	Ave. = 10s Min. = 3s Max. = 30s	Ave. = 30s Min. = 3s Max. = 60s	Ave. = 15s Min. = 5s Max. = 120s	Ave. = 5s Min. = 3s Max. = 45s
Server response size (Srs)	Ave. = 2KB Min. = 500B Max. = 100KB	Ave. = 5KB Min. = 1KB Max. = 1MB	Ave. = 8KB Min. = 100B Max. = 50KB	Ave. = 200B Min. = 150B Max. = 750B
Number of server messages (Ns)	Ave. = 50 Min. = 10 Max. = 200	Ave. = 200 Min. = 25 Max. = 1000	Ave. = 100 Min. = 15 Max. = 500	Ave. = 15 Min. = 5 Max. = 40
Server processing time (Tsp)**	Ave. = 0.5s Min. = 0.1s Max. = 5s	Ave. = 1s Min. = 0.5s Max. = 20s	Ave. = 2s Min. = 1s Max. = 10s	Ave. = 4s Min. = 0.5s Max. = 15s
File size upload (Su)	Ave. = 50KB Min. = 2KB Max. = 200KB	Ave. = 100KB Min. = 10KB Max. = 2MB	Ave. = N/A Min. = N/A Max. = N/A	Ave. = 100KB Min. = 20KB Max. = 10MB

File size	Ave. = 50KB	Ave. = 100KB	Ave. = N/A	Ave. = 100KB
download (Sd)	Min. = 2KB	Min. = 10KB	Min. = N/A	Min. = 20KB
	Max. = 200KB	Max. = 2MB	Max. = N/A	Max. = 10MB

* eCommerce used a combination of packet capture techniques and reference traffic flows from "SPECweb2009" (need proper reference)

** The client and server processing time is distributed across the transmission / receipt of all of messages. Client processing time consists mainly of the delay between user interactions (not machine processing).

And again, the parameters in this table are the guidelines for the TCP test pattern traffic generation. The test tool can use fixed parameters for simpler tests and mathematical distributions for more complex tests. However, the test pattern must be repeatable to ensure that the benchmark results can be reliably compared.

- SMB/CIFS File Copy: mimic a network file copy, both read and write. As opposed to FTP which is a bulk transfer and is only flow controlled via TCP, SMB/CIFS divides a file into application blocks and utilizes application level handshaking in addition to TCP flow control.

In summary, an SMB/CIFS file copy can be described by the following parameters:

- Client message size (Scm)
- Number of client messages (Nc)
- Server response size (Srs)
- Number of Server messages (Ns)
- Client processing time (Tcp)
- Server processing time (Tsp)
- Block size (Sb)

The client and server messages are SMB control messages. The Block size is the data portion of the file transfer.

Again using packet capture as a means to characterize the following table reflects the guidelines for SMB/CIFS file copy:

Parameter	SMB File Copy
Client message size (Scm)	Ave. = 450B Min. = 100B Max. = 1.5KB
Number of client messages (Nc)	Ave. = 10 Min. = 5 Max. = 25
Client processing time (Tcp)	Ave. = 1ms Min. = 0.5ms Max. = 2
Server response size (Srs)	Ave. = 2KB Min. = 500B Max. = 100KB
Number of server messages (Ns)	Ave. = 10 Min. = 10 Max. = 200
Server processing time (Tsp)	Ave. = 1ms Min. = 0.5ms Max. = 2ms
Block Size (Sb)*	Ave. = N/A Min. = 16KB Max. = 128KB

*Depending upon the tested file size, the block size will be transferred n number of times to complete the example. An example would be a 10 MB file test and 64KB block size. In this case 160 blocks would be transferred after the control channel is opened between the client and server.

7. Security Considerations

8. IANA Considerations

9. Conclusions

10. References

10.1. Normative References

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

[2] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

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

[RFC2234] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

10.2. Informative References

11. Acknowledgments

Authors' Addresses

Barry Constantine

JDSU, Test and Measurement Division

Germantown, MD 20876-7100, USA

Phone: +1 240 404 2227

Email: barry.constantine@jdsu.com

Timothy Copley

Level 3 Communications

14605 S 50th Street

Phoenix, AZ 85044

Email: Timothy.copley@level3.com

Ram Krishnan

Brocade Communications

San Jose, 95134, USA

Phone: +001-408-406-7890

Email: ramk@brocade.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 29, 2015

A. Morton
AT&T Labs
October 26, 2014

Considerations for Benchmarking Virtual Network Functions and Their
Infrastructure
draft-morton-bmwg-virtual-net-02

Abstract

Benchmarking Methodology Working Group has traditionally conducted laboratory characterization of dedicated physical implementations of internetworking functions. This memo investigates additional considerations when network functions are virtualized and performed in commodity off-the-shelf hardware.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 29, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Scope	3
3. Considerations for Hardware and Testing	3
3.1. Hardware Components	3
3.2. Configuration Parameters	4
3.3. Testing Strategies	4
4. Benchmarking Considerations	5
4.1. Comparison with Physical Network Functions	5
4.2. Continued Emphasis on Black-Box Benchmarks	5
4.3. New Benchmarks	6
4.4. Assessment of Benchmark Coverage	7
5. Security Considerations	8
6. IANA Considerations	8
7. Acknowledgements	8
8. References	9
8.1. Normative References	9
8.2. Informative References	10
Author's Address	10

1. Introduction

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

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

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

2. Scope

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

This memo investigates additional methodological considerations necessary when benchmarking VNF instantiated and hosted in commodity off-the-shelf (COTS) hardware. An essential consideration is benchmarking both physical and virtual network functions, thereby allowing direct comparison.

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

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

A colossal non-goal is any form of architecture development related to NFV and associated technologies in BMWG, as has been the case since BMWG began work in 1989.

3. Considerations for Hardware and Testing

This section lists the new considerations which must be addressed to benchmark VNF(s) and their supporting infrastructure.

3.1. Hardware Components

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

1. High volume server platforms (COTS, possibly with virtual technology enhancements).
2. Storage systems with large capacity, high speed, and high reliability.

3. Network Interface ports specially designed for efficient service of many virtual NICs.
4. High capacity Ethernet Switches.

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

3.2. Configuration Parameters

It will be necessary to configure and document the settings for the entire COTS platform, including:

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

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

- o Hypervisor
- o Virtual Machine
- o Infrastructure Virtual Network

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

- o specific function being implemented in VNF
- o number of VNF components in the service function chain
- o number of physical interfaces and links transited in the service function chain

3.3. Testing Strategies

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

1. It may be more representative of system capacity to characterize the case where Virtual Machines (VM, hosting the VNF) are operating at 50% Utilization, and therefore sharing the "real" processing power across many VMs.
2. Another important case stems from the need for partitioning functions. A noisy neighbor (VM hosting a VNF in an infinite loop) would ideally be isolated and the performance of other VMs would continue according to their specifications.
3. System errors will likely occur as transients, implying a distribution of performance characteristics with a long tail (like latency), leading to the need for longer-term tests of each set of configuration and test parameters.
4. The desire for Elasticity and flexibility among network functions will include tests where there is constant flux in the VM instances. Requests for new VMs and Releases for VMs hosting VNFs no longer needed would be a normal operational condition.
5. All physical things can fail, and benchmarking efforts can also examine recovery aided by the virtual architecture with different approaches to resiliency.

4. Benchmarking Considerations

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

4.1. Comparison with Physical Network Functions

In order to compare the performance of virtual designs and implementations with their physical counterparts, identical benchmarks must be used. Since BMWG has developed specifications for many network functions already, there will be re-use of existing benchmarks through references, while allowing for the possibility of benchmark curation during development of new methodologies. Consideration should be given to quantifying the number of parallel VNFs required to achieve comparable performance with a given physical device, or whether some limit of scale was reached before the VNFs could achieve the comparable level.

4.2. Continued Emphasis on Black-Box Benchmarks

When the network functions under test are based on Open Source code, there may be a tendency to rely on internal measurements to some extent, especially when the externally-observable phenomena only support an inference of internal events (such as routing protocol

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

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

4.3. New Benchmarks

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

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

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

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

4.4. Assessment of Benchmark Coverage

It can be useful to organize benchmarks according to their applicable lifecycle stage and the performance criteria they intend to assess. The table below provides a way to organize benchmarks such that there is a clear indication of coverage for the intersection of lifecycle stages and performance criteria.

	SPEED	ACCURACY	RELIABILITY
Activation			
Operation			
De-activation			

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

Comment/Discussion:

In one of the first applications of the 3x3 matrix on BMWG, we discovered that metrics on measured size, capacity, or scale do not easily match one of the three columns above. There are three alternatives to resolve this:

1. Add a column, Scaleability, but then it would be expected to have metrics in most of the Activation, Operation, and De-activation functions (which may not be the case).

2. Include Scalability under Reliability: This fits the user perspective of the 3x3 matrix because the size or capacity of a device contributes to the likelihood that a request will be blocked, or that operation will be un-reliable when operating in an overload state.
3. Keep size, capacity, and scale metrics separate from the 3x3 matrix, and present the results for key benchmarks in different versions of the matrix, and the titles of each matrix provide the details of configuration and scale.

Alternative 3 would address a discussion comment from IETF-90, so it seems to cover a range of wanted features.

5. Security Considerations

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

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

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

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

6. IANA Considerations

No IANA Action is requested at this time.

7. Acknowledgements

The author acknowledges an encouraging conversation on this topic with Mukhtiar Shaikh and Ramki Krishnan in November 2013. Bhuvaneshwaran Vengainathan, Bhavani Parise, and Ilya Varlashkin have provided useful suggestions to expand these considerations.

8. References

8.1. Normative References

- [NFV.PER001] "Network Function Virtualization: Performance and Portability Best Practices", Group Specification ETSI GS NFV-PER 001 V1.1.1 (2014-06), June 2014.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.
- [RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, September 1999.
- [RFC2680] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, September 1999.
- [RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.
- [RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, November 2002.
- [RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, November 2002.
- [RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, November 2006.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.
- [RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

8.2. Informative References

- [RFC1242] Bradner, S., "Benchmarking terminology for network interconnection devices", RFC 1242, July 1991.
- [RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, March 2009.
- [RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, April 2011.
- [RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, October 2011.

Author's Address

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

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com
URI: <http://home.comcast.net/~acmacm/>