Network Working Group
INTERNET-DRAFT
Intended Status: Standards Track
Expires: September 5, 2011

S. Baillargeon
C. Flinta
A. Johnsson
S. Ekelin
Ericsson

March 4, 2011

TWAMP Value-Added Octets
draft-baillargeon-ippm-twamp-value-added-octets-01.txt

Abstract

This memo describes the optional extensions to the standard TWAMP test protocol for identifying test sessions and packet trains, and for measuring capacity metrics like the available path capacity, tight section capacity and UDP throughput in the forward and reverse path directions.

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), 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/1id-abstracts.html

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

Table of Contents

1 Introduction

The notion of embedding a number of meaningful fields in the padding octets has been established as a viable methodology for carrying additional information within the TWAMP-Test protocol running between a Session-Sender and a Session-Reflector [RFC5357] [RFC6038].

This memo describes an OPTIONAL feature for the Two-Way Active Measurement Protocol [RFC5357]. It is called the Value-Added Octets feature.

This feature enables the controller host to measure capacity metrics like the IP-type-P available path capacity (APC) [RFC5136], IP-layer tight section capacity (TSC) [Y1540] and UDP throughput [RFC1242] on both forward and reverse paths. With this feature, it is also possible to improve the demultiplexing of test packets to the correct test sessions running on the controller and responder hosts when methods solely based on IP and UDP header information is not desirable or insufficient.

The Valued-Added Octets feature consists of new behaviors for the Session-Sender and Session-Reflector, and a set of value-added octets of information that are placed at the beginning of the Packet Padding field [RFC5357] or at the beginning of the Packet Padding (to be reflected) field [RFC6038] by the Session-Sender, and are reflected or returned by the Session-Reflector. The length of the value-added octets varies in size between 6, 10 and 14 octets depending on the setting of the flag bits specified at the beginning of the value-added octets.

This memo is an update to the TWAMP core protocol specified in [RFC5357]. Measurement systems are not required to implement the feature described in this memo to claim compliance with [RFC5357].

UDP throughput is defined in the Benchmarking Terminology for Network Interconnection Devices [RFC1242]. IP-Type-P APC metric is defined in Defining Network Capacity [RFC5136]. IP-layer TSC metric is defined in IP Packet Transfer and Availability Performance Parameters [Y1540]. The actual method to calculate the available path capacity, the tight section capacity or the UDP throughput from packet-level data performance data is not discussed in this memo.

1.1 Requirements Language

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

2 Purpose and scope

The purpose of this memo is to define the OPTIONAL Valued-Added Octets feature for TWAMP [RFC5357].

The scope of the memo is limited to specifications of the following enhancements:

- The extension of the modes of operation through assignment of a new value in the Mode field to communicate feature capability and use,
- The definition of a structure for embedding a sequence of value-added fields at the beginning of the Packet Padding field [RFC5037] or Packet Padding (to be reflected) field [RFC6038] in the TWAMP test packets and,
- The definition of new Session-Sender and Session-Reflector behaviors

The motivation for this feature is to enable the measurements of capacity metrics on both the forward and reverse paths, and to improve the demultiplexing of test packets to the correct test session at both endpoints.

This memo extends the modes of operation through assignment one new value in the Modes field (see Section 3.1 of [RFC4656] for the format of the Server Greeting message), while retaining backward compatibility with the core TWAMP [RFC5357] implementations. The new value correspond to the Valued-Added Octets Version 1 feature defined in this memo.

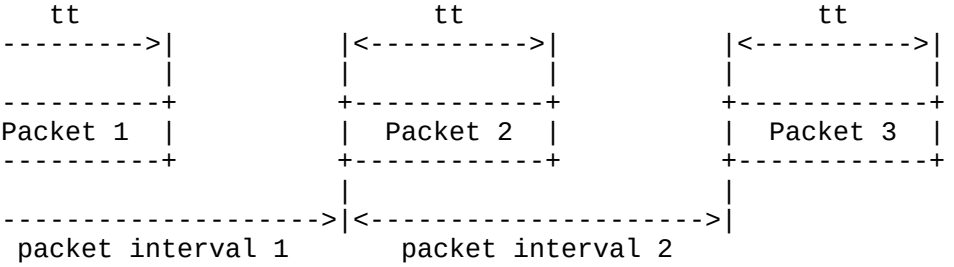
When the Server and Control-Client have agreed to use the Value-Added Octets Version 1 mode during control connection setup, then the Control-Client, the Server, the Session-Sender, and the Session-Reflector MUST all conform to the requirements of that mode, as identified below.

The OPTIONAL packet padding octets are designed to retain backward compatibility with the original TWAMP test protocol [RFC5357].

3 Capacity Measurement Principles

Most capacity estimation methods for available path capacity [RRBNC][PDM][ENHJMMB][SBW] and for UDP throughput [RFC2544] need to send and receive packets in groups, called packet trains or simply trains. Each train is sent at a specific transmission rate in a given direction. These trains must be identified within each bi-directional test session stream.

The first measurement principle is to send multiple trains within a test session stream from one IP node to another IP node in order to estimate the available path capacity, tight section capacity or UDP throughput in the forward direction. Each train consists of a group of test packets which are separated from each other by a packet interval, as shown in the picture below.



The packet interval between consecutive packets for each train sent by the Session-Sender and reflected by the Session-Reflector MUST be calculated and determined by the controller or an application or entity communicating with the controller. The packet interval MAY be constant within a train. Determination of the packet interval within a train as well as for consecutive trains for a given test session is implementation-specific.

The transmission time tt to send one packet (i.e. determined by the interface speed and the IP packet size) is also shown in the picture. Observe that the packet interval MUST be larger than or equal to tt.

At the Session-Reflector, each received test packet within a forward train is time stamped. This provides a second set of packet interval values. Methods for measuring the available path capacity, tight section capacity and UDP throughput use the packet intervals obtained from both end points in the estimation process. The method to measuring the UDP throughput may also require the packet loss at the receiving end. The estimation process itself as well as any requirements on software or hardware is implementation-specific.

The second measurement principle is referred to as self-induced congestion. According to this principle, in order to measure the available path capacity, tight section capacity and UDP throughput, some trains MUST cause momentary congestion on the network path. In essence this means that some trains MUST be sent at a higher rate than what is available on the network path. The congestion is only transient, for the duration of the train which is typically short.

In order to fulfill the above measurement principles and to measure the available path capacity, tight section capacity and UDP throughput in the reverse direction, the reflected test packets MUST be re-grouped into trains at the Session-Reflector.

4 Test packet Demultiplexing Principles

The controller (or the Session-Sender) requires a method for demultiplexing the received test packets to the correct test session especially when it manages multiple active test sessions. The responder also requires a method for demultiplexing the received test packets from multiple active test sessions originating from the same controller or from different controllers.

The purpose of this section is to provide some basic principles for identifying the test packets and to clarify the optional usage of the Sender Discriminator (SD) field described in this memo. It is important to note the actual method for identifying a test packet and the process for mapping it to the correct test session are implementation-specific. They may differ between various controllers and responders.

In general, the methods are based on fields available in the various headers of the TWAMP test packet (e.g. Ethernet, IP, UDP and TWAMP headers). Note the SID [RFC4656] is generally not used for identification purpose since it does not normally appear in the TWAMP test packets. As an example, a measurement system (controller or responder) may use the source IP address of the incoming test packet in order to associate it to the correct test session. This method is valid but has a number of limitations. It is simple and effective when each measurement system only requires a single test session for each peer but fails when multiple test sessions (with different characteristics) are running between the same pair of controller and responder.

Another approach is to use a combination of the source IP address, destination IP address, source UDP port and destination UDP port. This method is also valid but to work effectively, it requires that the controller allocates multiple UDP ports (one for each test session for instance) and/or the responder listens on multiple ports.

Ideally, a measurement system should limit the number of UDP ports for sending and receiving test packets. This approach may be improved by using a combination of the IP addresses, UDP ports and DSCP codepoint. This method also has its limitations. For instance, it cannot identify test packets from different test sessions running between the same pair of controller and responder if they are using the same UDP endpoints and the same DSCP codepoint.

This memo introduces a new field, the Sender Discriminator (SD) field intended to simplify the identification of the test packets at the controller and responder. It is especially useful when multiple test sessions with different DSCP codepoints and/or test packet sizes are expected to be running between the same pair of UDP endpoints. As described in 6.1.2, the SD is a number generated by the Session-Sender that uniquely identifies a test session on its system. With this field, the controller can explicitly identify the test packets belonging to a test session. When provided, the responder MAY use the SD field in combination of the source IP address for instance to identify the test packets belonging to a test session.

5 TWAMP Control Extensions

TWAMP-Control protocol [RFC5357] uses the Modes field to identify and select specific communication capabilities, and this field is a recognized extension mechanism.

TWAMP connection establishment follows the procedure defined in Section 3.1 of [RFC4656] and Section 3.1 of [RFC5357]. The new feature require one new bit position (and value) to identify the ability of the Server/Session-Reflector to read and act upon the new fields in the value-added octets. See the IANA section for details on the assigned value and bit position.

The Server sets the new bit position in the Modes field of the Server Greeting message to indicate its capability to operate in this new mode.

Both the Reflect Octets mode and Symmetrical Size mode SHOULD be selected to ensure the reflection of the value-added padding octets by the Session-Reflector and symmetrical size TWAMP-Test packets in the forward and reverse directions of transmission.

The forward and reverse APC, TSC and UDP throughput measurement characteristics depend on the size of the test packets. All test packets (forward and reverse test packets) belonging to a specific test session responsible to measure the available path capacity, tight section capacity and/or UDP throughput MUST have the same IP

packet size.

6 Extended TWAMP Test

The TWAMP-test protocol carrying the value-added padding octets is identical to TWAMP [RFC5357] except for the definition of first 6, 10 or 14 octets in the Padding Octet field that the Session-Sender expects to be reflected.

The Session-Sender and Session-Reflector behaviors are also modified.

6.1 Sender Behavior

This section describes the extensions to the behavior of the TWAMP Session-Sender.

When the Value-Added Octets Version 1 mode is selected, the Session-Sender MAY set the Sender Discriminator Present bit to 1. If it is set to 1, the Session-Sender MUST generate and transmit a unique nonzero discriminator value in the Sender Discriminator field.

When the Value-Added Octets Version 1 mode is selected, the Session-Sender MAY set the Last Seqno in Train Present bit to 1. If it is set to 1, the Session-Sender MUST generate and transmit a valid sequence number in the Last Seqno in Train field. The Session-Sender MUST also group the test packets in trains and send the trains towards the Session-Reflector at the desired forward packet intervals.

When the Value-Added Octets Version 1 mode is selected, the Session-Sender MAY set the the Desired Reverse Packet Interval Present bit to 1. If it is set to 1, the Session-Sender MUST generate and transmit a valid inter-packet time interval in the Desired Reverse Packet Interval field.

The desired forward and reverse rate interval parameters are usually provided by a measurement method, tool or algorithm. This measurement algorithm is outside the scope of this specification.

6.1.1 Packet Timings

The Send Schedule is not utilized in TWAMP and this is unchanged in this memo.

6.1.2 Session-Sender Packet Format

The Session-Sender packet format follows the same procedure and guidelines as defined in TWAMP [RFC5357] and TWAMP Reflect Octets and

Symmetrical Size Features [RFC6038].

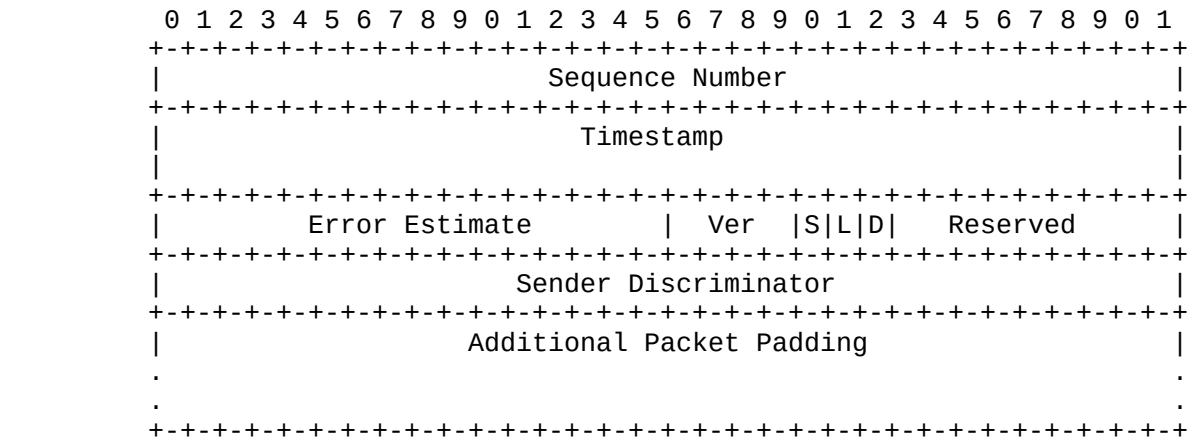
This feature allows the Session-Sender to set the first few octets in the TWAMP-Test Packet Padding field with information to communicate value-added padding version number, flag bits, sender discriminator, sequence number of the last packet in a train and desired inter-packet time interval (or per-packet waiting time) for the reverse path direction of transmission.

A version number and a sequence of flag bits are defined at the very beginning of the value-added padding octets. The version number identifies the version of the value-added padding octets and meaning of the flag bits. Each flag bit indicates if a specific field of a specific size is present in the valued-added padding octets. The flag bits are designed to accommodate different combinations of fields while reducing padding overhead when certain fields are not needed. The version number and flag bits also provide an effective method for parsing information at Session-Reflector and Session-Sender. This document defines version 1 with three flag bits: S, L and D.

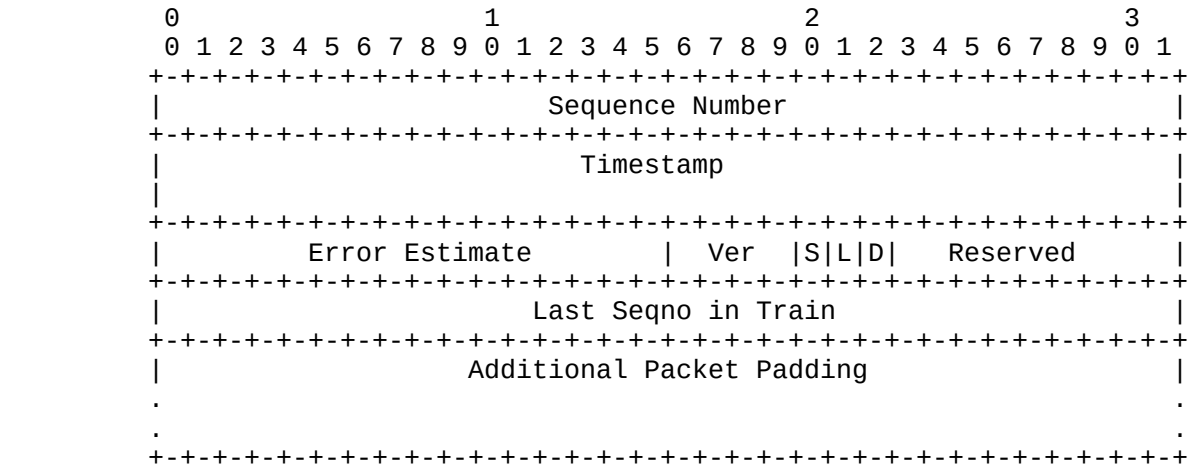
The format of the test packet depends on the TWAMP mode and flag bits being used. The Value-Added Octets Version 1 mode is intended to work with any TWAMP modes.

When the Value-Added Octets Version 1 is selected with S=1, L=0 and D=0, the Session-Sender SHALL use the following TWAMP test packet format in unauthenticated mode:

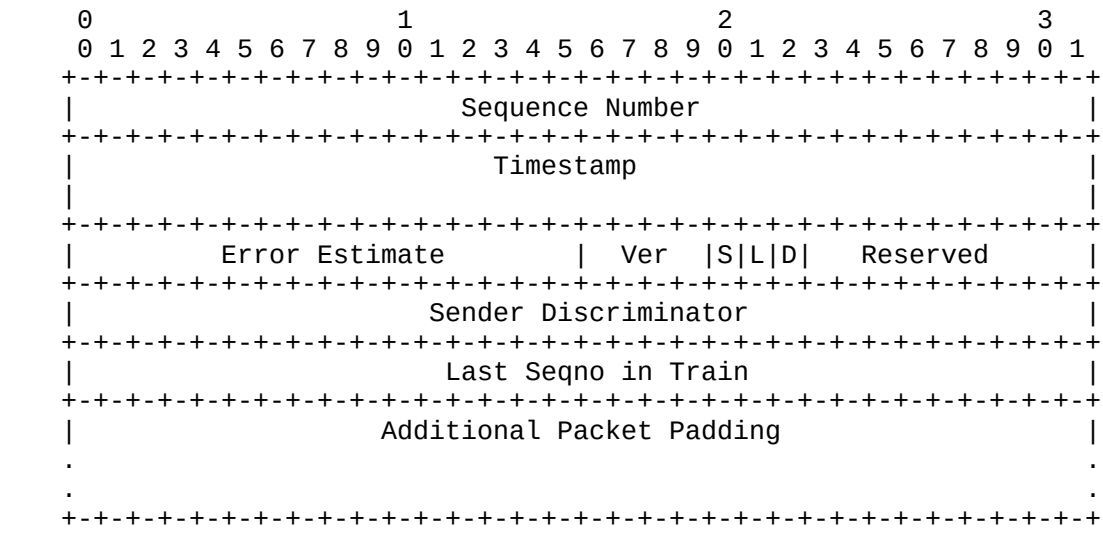
0 1 2 3



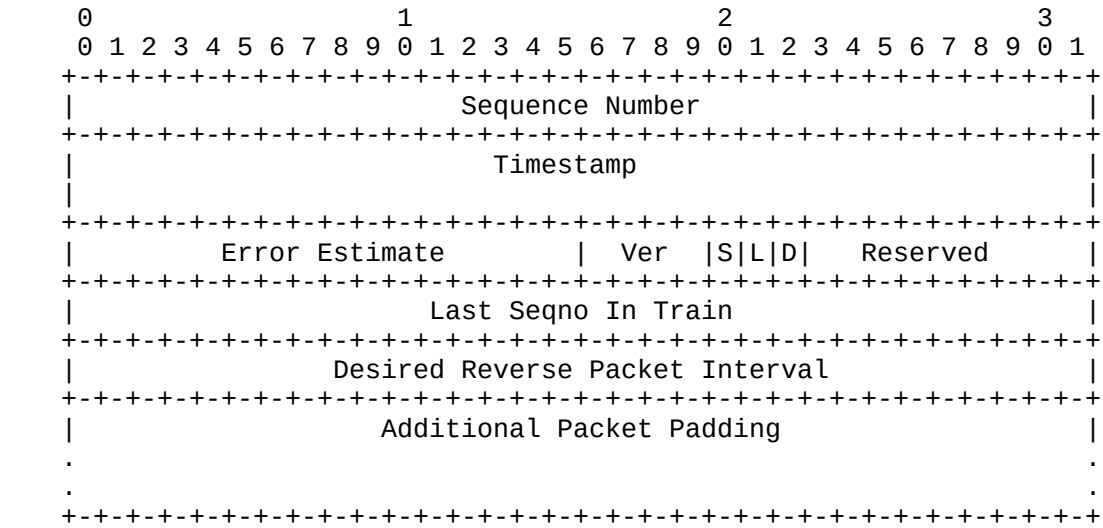
When the Value-Added Octets Version 1 is selected with S=0, L=1 and D=0, the Session-Sender SHALL use the following TWAMP test packet format in unauthenticated mode:



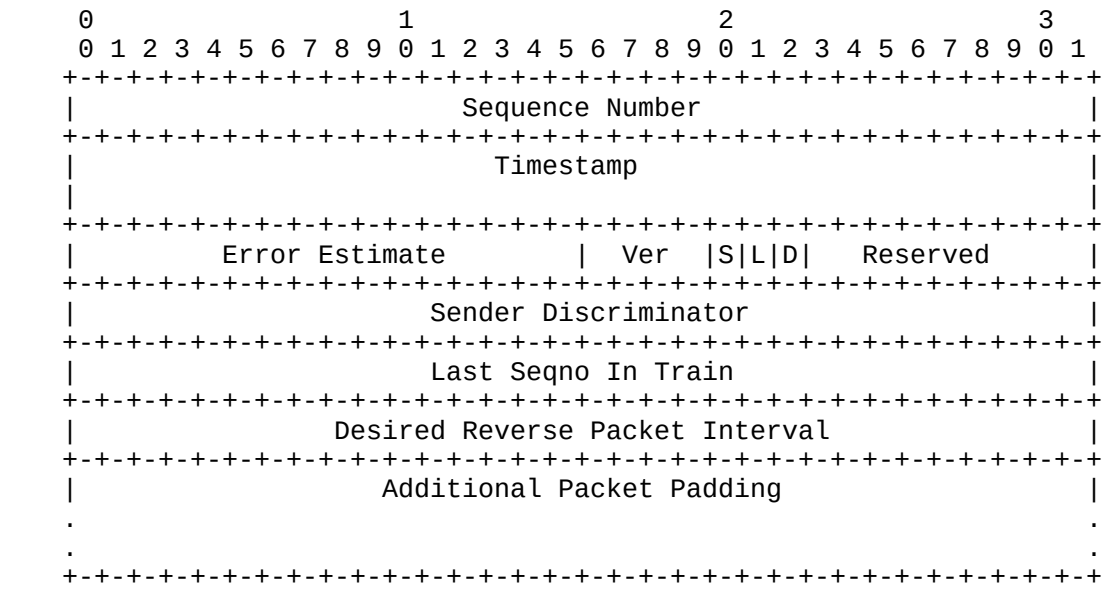
When the Value-Added Octets Version 1 is selected with S=1, L=1 and D=0, the Session-Sender SHALL use the following TWAMP test packet format in unauthenticated mode:



When the Value-Added Octets Version 1 is selected with S=0, L=1 and D=1, the Session-Sender SHALL use the following TWAMP test packet format in unauthenticated mode:

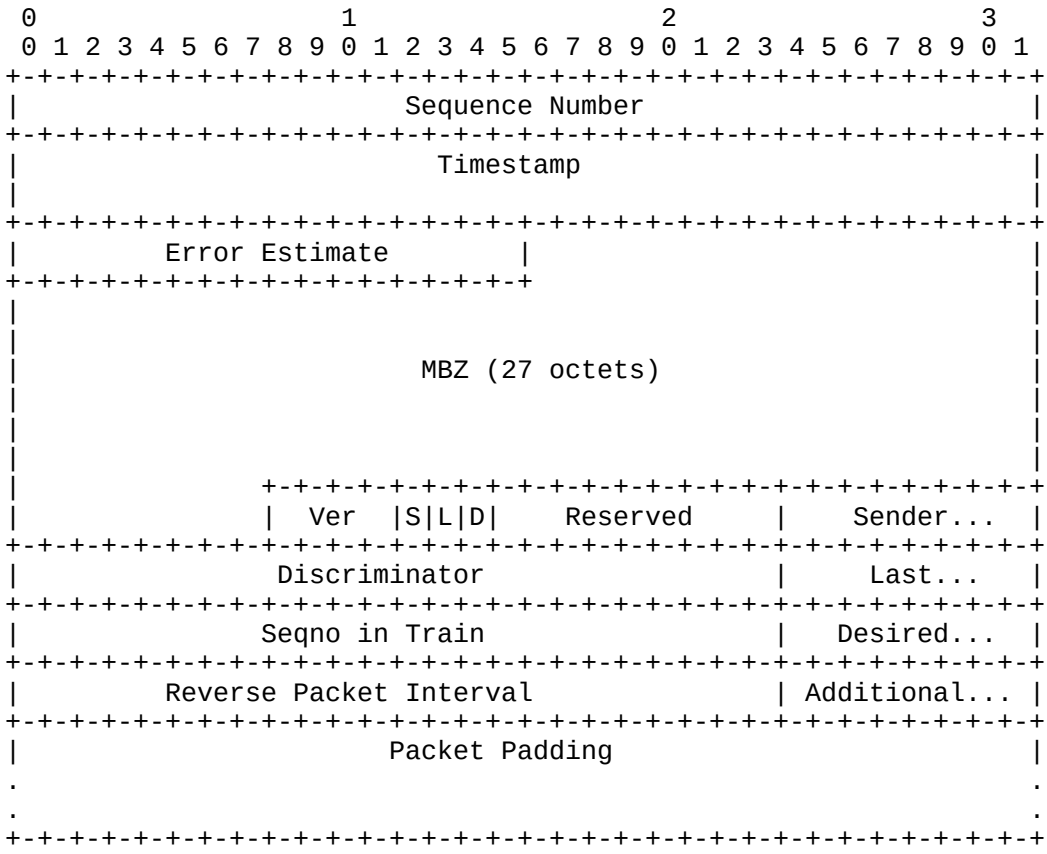


When the Value-Added Octets Version 1 is selected with S=1, L=1 and D=1, the Session-Sender SHALL use the following TWAMP test packet format in unauthenticated mode:



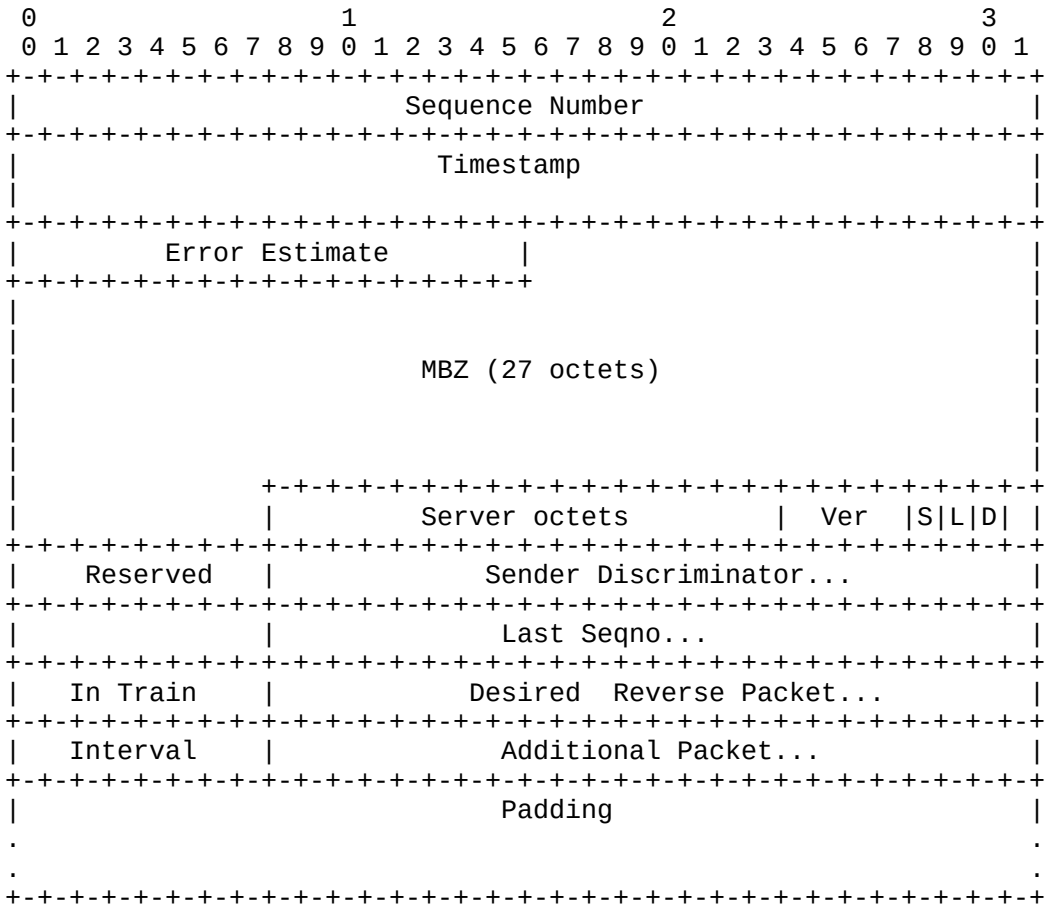
When the Value-Added Octets Version 1 is selected with S=1, L=1 and

D=1, the Session-Sender SHALL use the following TWAMP test packet format in conjunction with the unauthenticated mode, Symmetrical Size mode and Reflect Octets mode:



When the Value-Added Octets Version 1 is selected with S=1, L=1 and

D=1, the Session-Sender SHALL use the following TWAMP test packet format in conjunction with the unauthenticated mode, Symmetrical Size mode and Reflect Octets mode with a non-zero value in the Server octets field:



In the combined mode including Reflect Octets, the value-added padding octets are embedded in the Packet Padding (to be reflected) field.

The Version (Ver) field MUST be encoded in the first 4 bits. It identifies the version number of the value-added padding octets and meaning of the flag bits. This document defines version 1.

The Sender Discriminator Present bit (S) MUST be the first flag. If the Sender Discriminator Present bit is set to 1, then a Sender Discriminator field MUST be present and MUST contain valid information.

The Last Seqno in Train Present bit (L) MUST be the second flag. If the Last Seqno in Train Present bit is set to 1, then the Last Seqno in Train field MUST be present and MUST contain valid information.

The Desired Reverse Packet Interval Present bit (D) MUST be the third flag. If the Desired Reverse Packet Interval Present bit is set to 1, then Desired Reverse Packet Interval Present field MUST be present and MUST contain valid information.

The Reserved field is reserved for future use. All 9 bits of the Reserved field MUST be transmitted as zero by the Session-Sender.

The Sender Discriminator (SD) field MUST contain an unsigned 32 bit integer generated by the Session-Sender. It is used by the Session-Reflector and/or Session-Sender to identify packets belonging to a test session. The Session-Sender MUST choose a nonzero discriminator value that is unique among all test sessions on its system. This field is present only if the Sender Discriminator Present bit is set to one.

The Last Seqno in Train MUST contain an unsigned 32 bit integer generated by the Session-Sender. It MUST indicate the expected sequence number of the last packet in the train. It SHOULD be used by the Session-Sender and Session-reflector to identify the train a test packet belongs to. The packets belonging to a train are determined by observing the test packet sequence number in relation to the Last Seqno for a train. The sequence number of a packet in a train MUST be lower than or equal to the Last Seqno for that train. The sequence number MUST also be larger than the Last Seqno for the previous train. This field is present only if the Last Seqno in Train Present bit is set to one.

The Desired Reverse Packet Interval (DRPI) MUST contain an unsigned 32 bit integer generated by the Session-Sender. It MUST indicate the desired inter-packet time interval (or the waiting time) that the Session-Reflector SHOULD use when transmitting the reflected test packets towards the Session-Sender. The value 0 means the The Session-Reflector SHOULD return the test packet to the Session-Sender as quickly as possible. The format of this field MUST be a fractional

part of a second as defined in OWAMP [RFC4656]. This field is present only if the Desired Reverse Packet Interval Present bit is set to one.

The method by which the Sender Discriminator and Desired Reverse Packet Interval values are obtained is outside of the scope of this document.

6.2 Reflector behavior

The TWAMP Session-Reflector follows the procedures and guidelines in Section 4.2 of [RFC5357], with some changes and additional functions.

When the Value-Added Octets Version 1 is selected, the behavior of the Session-Reflector SHALL be as follows:

- The Session-Reflector MUST read the Version field. If Ver=1, the Session-Reflector MUST read the S, L and D flag bits. If Ver is not equal 1, the Session-Reflector MUST ignore the rest of the value-added padding octets and MUST follow the procedures and guidelines described in section 4.2 of [RFC5357]. The Session-Reflector SHOULD transmit the packet as quickly as possible including the test packets that are currently stored for the test session.

- If S=0, L=0 and D=0, the Session-Reflector MUST ignore the rest of the value-added padding octets and MUST follow the procedures and guidelines described in section 4.2 of [RFC5357]. The Session-Reflector SHOULD transmit the packet as quickly as possible including the test packets that are currently stored for the test session.

- If S=1, the Session-Reflector MUST continue reading and extracting the information from the Sender Discriminator field in the value-added padding octets.

- After reading and extracting the information from the Sender Discriminator field, the Session-Reflector SHOULD associate the test packets to the correct test session based on the value specified in the Sender Discriminator field and the source IP address specified in the IP header of the test packet. The actual method for demultiplexing the received test packets to the correct test session based on the Sender Discriminator and source IP address is outside the scope of this specification. The Session-Reflector MAY also use additional packet fields to demultiplex test packets to a test session.

- If L=1, the Session-Reflector MUST continue reading and extracting the information from the Last Seqno in Train field in the value-added padding octets.

- After reading and extracting the information from the Last Seqno in Train field, Last Seqno in Train field MUST be compared to Sequence number in the same packet in order to determine when a complete train has been collected. The Session-Reflector SHOULD buffer the packets belonging to the current train (or store the packet-level performance data) and SHOULD transmit them as immediately as possible after the last packet of the train has been received. The last packet within a train has Sender Sequence Number = Last Seqno in Train.

- The Last Seqno in Train of a packet MUST also be compared to the Last Seqno in Train of the previous packet in order to determine if a new train needs to be collected. In case of packet loss, the Session-Reflector MUST transmit the incomplete train when it receives a packet with a Last SeqNo in Train belonging to the another train (e.g. next train) of the test session, or after a timeout. The timeout MAY be the REFWAIT timer specified in section 4.2 of [RFC5357].

- Packets arriving out-of-order within a train MUST be buffered at the Session-Reflector if the train is not yet transmitted to the Session-Sender. If the train is already transmitted, the test packet SHOULD be returned to the Session-Sender as quickly as possible. The Session-Reflector MUST not reorder the test packets if they happen to arrive out-of-sequence.

- Duplicate packets within a train MUST be buffered at the Session-Reflector if the train is not yet transmitted to the Session-Sender. If the train is already transmitted, the duplicate test packet SHOULD be returned to the Session-Sender as quickly as possible. The Session-Reflector MUST not discard duplicate test packets.

- If D=1, the Session-Reflector MUST continue reading and extracting the information from the Desired Reverse Packet Interval field in the value-added padding octets.

- After reading and extracting the information from the Desired Reverse Packet Interval field, the Session-Reflector SHOULD transmit the packets belonging to a reverse train with a waiting time (packet interval) for each packet indicated in the Desired Reverse Packet Interval field. If the Desired Reverse Packet Interval field is set to zero, then the Session-Reflector SHOULD transmit the packets as quickly as possible.

The Session-Reflector MUST implement the changes described above when the Value-Added Octets Version 1 mode is selected.

6.2.1 Session-Reflector Packet Format

The Session-Reflector packet format follows the same procedure and guidelines as defined in TWAMP [RFC5357] and TWAMP Reflect Octets and Symmetrical Size Features [RFC6038], with the following changes:

- The Session-Reflector MUST re-use (reflect) the value-added padding octets (6, 10 or 14 octets) provided in the Sender's Packet Padding.
- The Session-Reflector MAY re-use the rest of the padding octets in the Sender's Packet Padding.

When using the recommended truncation process [RFC5357], the Session-Reflector MUST truncate exactly 27 octets of padding in Unauthenticated mode,and exactly 56 octets in Authenticated and Encrypted modes.

6.3 Additional Considerations

It is not required to use the Sender Discriminator field for calculating the capacity metrics. The Sender Discriminator Present bit can be set to zero. However, the Session-Sender and Session-Reflector MUST implement a local policy to identify the test packets belonging to a specific test session. The method for demultiplexing the received test packets to the correct test session based on other packet fields (e.g. fields in the IP header) is outside the scope of this specification.

Capacity measurements introduce an additional consideration when the test sessions operate in TWAMP Light. When the Session-Reflector does not have knowledge of the session state, the measurement system will only be capable to estimate or calculate the capacity metrics in the forward path direction of transmission. Capacity measurements in the reverse path direction requires the Session-Reflector to have knowledge of the session state and be capable to identify the test packets belonging to a specific test session. The method for creating a session state from the initial test packets on the TWAMP Light Session-Reflector is outside the scope of this specification.

7 Security Considerations

The value-added padding octets permit new attacks on the responder host communicating with core TWAMP [RFC5357]. The responder host MUST provide a mechanism to protect or limit the use of its local memory or buffer space.

The security considerations that apply to any active measurement of live networks are relevant here as well. See [RFC4656] and [RFC5357].

8 IANA Considerations

This memo adds one mode to the IANA registry for the TWAMP Modes field, and describes behavior when the new modes are used. This field is a recognized extension mechanism for TWAMP.

8.1. Registry Specification

IANA has created a TWAMP-Modes registry (as requested in [RFC5618]). TWAMP-Modes are specified in TWAMP Server Greeting messages and Setup Response messages, as described in Section 3.1 of [RFC5357], consistent with Section 3.1 of [RFC4656], and extended by this memo. Modes are indicated by setting bits in the 32-bit Modes field that correspond to values in the Modes registry. For the TWAMP-Modes registry, we expect that new features will be assigned increasing registry values that correspond to single bit positions, unless there is a good reason to do otherwise (more complex encoding than single-bit positions may be used in the future to access the 2^32 value space).

8.2. Registry Contents

The TWAMP-Modes registry has been augmented as follows:

Value	Description	Semantics Definition
128	Valued-Added Octets Ver 1	This memo, Section 2 new bit position (7)

9 References

9.1 Normative References

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

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol(OWAMP)", RFC 4656, September 2006.

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

[RFC5136] Chimento, P. and Ishac,J., "Defining Network Capacity", RFC 5136, February 2008.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.

[RFC6038] Morton, A., Ciavattone, L., TWAMP Reflect Octets and Symmetrical Size Features, RFC6038 , October 2010.

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

9.2 Informative References

[RRBNC] Ribeiro, V., Riedi, R., Baraniuk, R., Navratil, J., Cottrel, L., Pathchirp: Efficient available bandwidth estimation for network paths, Passive and Active Measurement Workshop, 2003.

[PDM] Dovrolis, C., Ramanathan, P., and Moore D., Packet Dispersion Techniques and a Capacity Estimation Methodology, IEEE/ACM Transactions on Networking, December 2004.

[ENHJMMB] Ekelin, S., Nilsson, M., Hartikainen, E., Johnsson, A., Mangs, J., Melander, B., Bjorkman, M., Real-time measurement of end-to-end available bandwidth using kalman filtering, Proceedings to the IEEE IFIP Network Operations and Management Symposium, 2006.

[SBW] Sommers, J., Barford, P., Willinger, W., Laboratory-based calibration of available bandwidth estimation tools, Microprocess Microsyst., 2007.

[Y1540] ITU-T Y.1540, Internet protocol data communication service - IP packet transfer and availability performance parameters, 2011.

[MRM] Morton, A., Ramachandran, G., Maguluri, G., Reporting

Metrics Different Points of View, draft-ietf-ippm-reporting-metrics-03, June 2010.

Author's Addresses

Steve Baillargeon
Ericsson
3500 Carling Avenue
Ottawa, Ontario K2H 8E9
Canada
EMail: steve.baillargeon@ericsson.com

Christofer Flinta
Ericsson
Farogatan 6
Stockholm, 164 80
Sweden
EMail: christofer.flinta@ericsson.com

Andreas Johnsson
Ericsson
Farogatan 6
Stockholm, 164 80
Sweden
EMail: andreas.a.johnsson@ericsson.com

Svante Ekelin
Ericsson
Farogatan 6
Stockholm, 164 80
Sweden
EMail: svante.ekelin@ericsson.com

Internet Engineering Task Force
Internet-Draft
Obsoletes: 5136 (if approved)
Intended status: Informational
Expires: April 19, 2011

X. Cui
Huawei

October 16, 2010

Defining Network Capacity
draft-cui-ippm-rfc5136bis-00

Abstract

This document defines the metric of network capacity, including link capacity aspect, router capacity aspect and path capacity aspect. RFC5136 has defined link capacity and path capacity, where the router impact is implicitly considered in link capacity. However, in this document, router capacity is considerred as a separate factor of path capacity, no longer a factor of link capacity. This document explicitly describes the router capacity and its impact to network capacity, e.g. how to evaluate path capacity.

This document is derived from RFC5136 and obsoletes RFC5136.

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 19, 2011.

Copyright Notice

Table of Contents

1. Introduction

The IPPM working group has defined a framework for IP Performance Metrics [RFC2330] and a set of IP Performance Metrics, such as One-way Delay Metric [RFC2679], Packet Delay Variation Metric [RFC3393] and Network Capacity Metric [RFC5136].

Network capacity, which is defined in [RFC5136], is one of the most important IP Performance Metrics in internet. In [RFC5136], network capacity consists of link capacity, path capacity, link usage, link utilization, available link capacity and available path capacity. [RFC5136] also introduces the definitions, measurement and calculation methods and some important formulas.

As stated in [RFC5136], "measuring the capacity of a link or network path is a task that sounds simple, but in reality can be quite complex". There are so many factors and so complicated coupling (between these factors) that the factor of router capacity is not explicitly stated in [RFC5136]. Router is an important element of internet and it is also an essential component of path. In [RFC5136] router impact is implicitly considered in link capacity, but it should be considered in path and path capacity instead, because router is a part of path while not a part of link.

This memo explicitly presents that the router factor should be considered in path, path capacity and related metrics (e.g. available path capacity). For the integrity of network capacity metrics, this memo additionally defines router capacity, router usage, router utilization and available router capacity.

This memo is the latest development based on [RFC5136] and draws heavily from it.

The remainder of this memo is structured as follows.

Section 2.1 contains component definitions and explanations (node, host, router, link, path, etc.)

Section 2.2 contains nominal physical capacity and explanations of link and router.

Section 2.3 give IP-layer capacity definitions and explanstions. It is structured in 11 subsections:

- IP-layer Bits (section 2.3.1)
- IP-type-P Link Capacity (section 2.3.2)
- IP-type-P Link Usage (section 2.3.3)
- IP-type-P Link Utilization (section 2.3.4)

- IP-type-P Available Link Capacity (section 2.3.5)
- IP-type-P Router Capacity (section 2.3.6)
- IP-type-P Router Usage (section 2.3.7)
- IP-type-P Router Utilization (section 2.3.8)
- IP-type-P Available Router Capacity (section 2.3.9)
- IP-type-P Path Capacity (section 2.3.10)
- IP-type-P Available Path Capacity (section 2.3.11)

Section 3 describes changes from [RFC5136]. Section 4 gives some complementary discussion. Section 5 gives discussion conclusion.

1.1. Overview of Capacity

Any physical medium requires that information be encoded and, depending on the medium, there are various schemes to convert information into a sequence of signals that are transmitted physically from one location to another.

While on some media, the maximum frequency of these signals can be thought of as "capacity", on other media, the signal transmission frequency and the information capacity of the medium (channel) may be quite different. For example, a satellite channel may have a carrier frequency of a few gigahertz, but an information-carrying capacity of only a few hundred kilobits per second. Often similar or identical terms are used to refer to these different applications of capacity, adding to the ambiguity and confusion, and the lack of a unified nomenclature makes it difficult to properly build, test, and use various techniques and tools.

We are interested in information-carrying capacity, but even this is not straightforward. Each of the layers, depending on the medium, adds overhead to the task of carrying information. The wired Ethernet uses Manchester coding or 4/5 coding, which cuts down considerably on the "theoretical" capacity. Similarly, RF (radio frequency) communications will often add redundancy to the coding scheme to implement forward error correction because the physical medium (air) is lossy. This can further decrease the information capacity.

In addition to coding schemes, usually the physical layer and the link layer add framing bits for multiplexing and control purposes. For example, on SONET there is physical-layer framing and typically also some layer-2 framing such as High-Level Data Link Control (HDLC), PPP, or ATM.

Aside from questions of coding efficiency, there are issues of how access to the channel is controlled, which also may affect the

capacity. For example, a multiple-access medium with collision detection, avoidance, and recovery mechanisms has a varying capacity from the point of view of the users. This varying capacity depends upon the total number of users contending for the medium, how busy the users are, and bounds resulting from the mechanisms themselves. RF channels may also vary in capacity, depending on range, environmental conditions, mobility, shadowing, etc.

The important points to derive from this discussion are these: First, capacity is only meaningful when defined relative to a given protocol layer in the network. It is meaningless to speak of "link" capacity without qualifying exactly what is meant. Second, capacity is not necessarily fixed, and consequently, a single measure of capacity at any layer may in fact provide a skewed picture (either optimistic or pessimistic) of what is actually available.

1.2. Requirements Language

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

2. Definitions

In this section, we specify component definitions and capacity definitions.

2.1. Component Definitions

In this section, we specify component definitions for network. We define "node", "Non-IP-Node", "host", "router", "link" and "path" clearly in this section, then we define capacity of network in next section.

2.1.1. Node

IPv6 Specification [RFC2460] defines node is a device that implements IPv6. Framework for IP Performance Metrics [RFC2330] defines host is a computer capable of communicating using the Internet protocols; includes "routers". The notion of host from [RFC2330] is equal to the notion of node from RFC2460. In this document, a node is a computer that implements IP protocol.

Note in this document any node without specicial statement is an IP node.

2.1.2. Non-IP-Node

In this document, a Non-IP-Node is a device that can transmit, receive or forward bit flow, but doesn't implement IP protocol. The examples of Non-IP-Node are ethernet switch and hub.

Note the Non-IP-Node may be part of link and impact the link capacity, for example, consider an ethernet switch that can operate ports at different speeds.

2.1.3. Host

IPv6 Specification [RFC2460] defines a host as any node that is not a router. This document adopts this definition, and the notion of host in this document dosn't includes "routers".

2.1.4. Router

[RFC2460] defines a router is a node that forwards IP packets not explicitly addressed to itself. This document adopts this definition.

2.1.5. Link

[RFC2460] defines link is a communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IPv6. Examples are Ethernets (simple or bridged); PPP links; X.25, Frame Relay, or ATM networks; and internet (or higher) layer "tunnels", such as tunnels over IPv4 or IPv6 itself. [RFC2330] defines link is a single link-level connection between two (or more) hosts; includes leased lines, ethernets, frame relay clouds, etc. This document adopts the definition from [RFC2460].

Note that link is a bidirectional concept, link terminal and link-layer middle-box are included in link.

2.1.6. Path

As defined in [RFC2330], a path of length n is a sequence of the form (N0, L1, N1, ..., Ln, Nn), where n >= 0, each Ni is a node, each Li is a link between Ni-1 and Ni, each N1...Nn-1 is a router. A pair (Li, Ni) is termed a 'hop'. In an appropriate operational configuration, the links and routers in the path facilitate network-layer communication of packets from N0 to Nn.

Note that path is a unidirectional concept and a path of length one is not equal to the corresponding link. In this case, the Link (i.e., L1) is a part of the path, i.e., the sequence of (N0, L1, N1).

2.2. Definition: Nominal Physical Capacity

Nominal physical link capacity, NomCap(L), is the theoretical maximum amount of data that the link L can support. For example, an OC-3 link would be capable of 155.520 Mbit/s. We stress that this is a measurement at the physical layer and not the network IP layer, which we will define separately. While NomCap(L) is typically constant over time, there are links whose characteristics may allow otherwise, such as the dynamic activation of additional transponders for a satellite link.

Note when we define nominal physical capacity of link, link terminals are considered while the nodes (host or router) which are connected by the link are not gathered. This is because the link terminal (e.g., network interface card) is not integrant of computer, it is only an accessory of the computer. However, there may be some Non-IP-Node in the link, such as an ethereal switch. The physical link capacity is affected by the switch’s ability to process and forward information bits for the given link.

The nominal physical link capacity is provided as a means to help distinguish between the commonly used link-layer capacities and the remaining definitions for IP-layer capacity. The nominal physical capacity provides an upper bound on link capaciy of both IP-layer and link-layer.

However, it is difficult to define the nominal physical capacity of a router. The routers are designed under many limitation, such as physical bound of CPU, memory and system bus. We usually use a pair of common principle to estimate a router: packet per second and bit per second. These two principles are coupled together, and in general, we almost can not correctly estimate a router by either of them. So we don’t define nominal physical router capacity in this document.

2.3. Capacity at the IP Layer

There are many factors that can reduce the IP information carrying capacity of the link. However, the goal of this document is not to become an exhaustive list of such factors. Rather, we outline some of the major examples in the following section, thus providing food for thought to those implementing the algorithms or tools that attempt to measure capacity accurately.

The remaining definitions are all given in terms of "IP-layer bits" in order to distinguish these definitions from the nominal physical capacity of the link.

2.3.1. Definition: IP-layer Bits

IP-layer bits are defined as eight (8) times the number of octets in all IP packets received, from the first octet of the IP header to the last octet of the IP packet payload, inclusive.

IP-layer bits are recorded at the destination D beginning at time T and ending at a time T+I. Since the definitions are based on averages, the two time parameters, T and I, must accompany any report or estimate of the following values in order for them to remain meaningful. It is not required that the interval boundary points fall between packet arrivals at D. However, boundaries that fall within a packet will invalidate the packets on which they fall. Specifically, the data from the partial packet that is contained within the interval will not be counted. This may artificially bias some of the values, depending on the length of the interval and the amount of data received during that interval. We elaborate on what constitutes correctly received data in the next section.

2.3.1.1. Standard or Correctly Formed Packets

The definitions in this document specify that IP packets must be received correctly. The IPPM framework recommends a set of criteria for such standard-formed packets in Section 15 of [RFC2330]. However, it is inadequate for use with this document. Thus, we outline our own criteria below while pointing out any variations or similarities to [RFC2330].

First, data that is in error at layers below IP and cannot be properly passed to the IP layer must not be counted. For example, wireless media often have a considerably larger error rate than wired media, resulting in a reduction in IP link capacity. In accordance with the IPPM framework, packets that fail validation of the IP header must be discarded. Specifically, the requirements in [RFC1812], Section 5.2.2, on IP header validation must be checked, which includes a valid length, checksum, and version field.

The IPPM framework specifies further restrictions, requiring that any transport header be checked for correctness and that any packets with IP options be ignored. However, the definitions in this document are concerned with the traversal of IP-layer bits. As a result, data from the higher layers is not required to be valid or understood as that data is simply regarded as part of the IP packet. The same holds true for IP options. Valid IP fragments must also be counted as they expend the resources of a link even though assembly of the full packet may not be possible. The IPPM framework differs in this area, discarding IP fragments.

For a discussion of duplicates, please see Section 4.2.

In summary, any IP packet that can be properly processed must be included in these calculations.

2.3.1.2. Type P Packets

The definitions in this document refer to "Type P" packets to designate a particular type of flow or sets of flows. As defined in [RFC2330], Section 13, "Type P" is a placeholder for what may be an explicit specification of the packet flows referenced by the metric, or it may be a very loose specification encompassing aggregates. We use the "Type P" designation in these definitions in order to emphasize two things: First, that the value of the capacity measurement depends on the types of flows referenced in the definition. This is because networks may treat packets differently (in terms of queuing and scheduling) based on their markings and classification. Networks may also arbitrarily decide to flow-balance based on the packet type or flow type and thereby affect capacity measurements. Second, the measurement of capacity depends not only on the type of the reference packets, but also on the types of the packets in the "population" with which the flows of interest share the links in the path.

All of this indicates two different approaches to measuring: One is to measure capacity using a broad spectrum of packet types, suggesting that "Type P" should be set as generic as possible. The second is to focus narrowly on the types of flows of particular interest, which suggests that "Type P" should be very specific and narrowly defined. The first approach is likely to be of interest to providers, the second to application users.

As a practical matter, it should be noted that some providers may treat packets with certain characteristics differently than other packets. For example, access control lists, routing policies, and other mechanisms may be used to filter ICMP packets or forward packets with certain IP options through different routes. If a capacity-measurement tool uses these special packets and they are included in the "Type P" designation, the tool may not be measuring the path that it was intended to measure. Tool authors, as well as users, may wish to check this point with their service providers.

2.3.2. Definition: IP-type-P Link Capacity

We define the IP-layer link capacity, C(L,T,I), to be the maximum number of IP-layer bits that can be transmitted from the source S and correctly received by the destination D over the link L during the interval [T, T+I], divided by I. The "maximum" means that IP-type-P

link capacity is the capacity representation when the link is fully utilized (i.e., nominal physical link capacity is fully used.)

In theory, IP-layer link capacity may be calculated out from nominal physical link capacity. Usually, for any link whose link protocol is given, we can know well the encapsulation, overhead and overtail of the link layer protocol. In these cases, for Type P Packets, whose length is Lp, we can get IP-layer link capacity as:

C(L,T,I)
= [Lp/(Lh + Lp + Lt)] * [1 - BER(T, T+I)] * [1 - BDR(T, T+I)] * P(L)

In this formula,
- Lp denotes type P packet length (in IP layer),
- Lh denotes link layer protocol overhead length,
- Lt denotes link layer protocol overtail length,
- BER(T, T+I) denotes Block Error or Lost Rate during the interval [T, T+I],
- BDR(T, T+I) denotes Block Duplication Rate during the interval [T, T+I],
- P(L) denotes nominal physical link capacity of the given link.

Like nominal physical link capacity, IP-type-P link capacity is also a theoretical maximum value. But IP-type-P link capacity is not constant over time, becauese there are many types of link layer protocol and BER and BDR (e.g., BER/BDR of radio channel) may vary in different period.

As defined in section 2.1.5, link is the layer 2 connection between nodes, so the nodes which are connected by the link are not part of the given link. However, there may be some Non-IP-Node in the link, such as an ethereal switch. The IP-type-P link capacity is affected by the switch’s ability to process and forward IP packets for the given link.

IP-type-P link capacity is affected by on-way Non-IP-Node but not affected by the nodes which are connected by the IP link. This means, the injecting node may affect how many packets are transpored between the source S and the destination D during the interval [T, T+I], and the incepting node may also affect how many packets are correctly received in the destination D, but these factors do not affect IP-type-P link capacity, because the capacity is the maximum value can be represented by the link.

IP-type-P link capacity is similar to IP-type-P link usage in some percent. The comparison is described in the next section.

2.3.3. Definition: IP-type-P Link Usage

The average usage of a link L, Used(L,T,I), is the actual number of IP-layer bits from any source, correctly received over link L during the interval [T, T+I], divided by I.

An important distinction between usage and capacity is that the capacity is a theoretical value (constant number) while the usage is a factually represented value (variable number). This is to say, Used(L,T,I) is not the maximum number, but rather, the actual average rate that IP bits are correctly received.

The information transmitted across the link can be generated by any source, including those sources that may not be directly attached to either side of the link. In addition, each information flow from these sources may share any number (from one to n) of links in the overall path between S and D.

2.3.4. Definition: IP-type-P Link Utilization

We express usage as a fraction of the overall IP-layer link capacity.

Util(L,T,I) = (Used(L,T,I) / C(L,T,I))

Thus, the utilization now represents the fraction of the capacity that is being used and is a value between zero (meaning nothing is used) and one (meaning the link is fully saturated). Multiplying the utilization by 100 yields the percent utilization of the link. By using the above, we can now define the available capacity over the link.

2.3.5. Definition: IP-type-P Available Link Capacity

We can now determine the amount of available capacity on a congested link by multiplying the IP-layer link capacity with the complement of the IP-layer link utilization. Thus, the IP-layer available link capacity becomes:

AvailCap(L,T,I) = C(L,T,I) * (1 - Util(L,T,I))

2.3.6. Definition: IP-type-P Router Capacity

As mentioned in section 2.2, we don’t define nominal physical router capacity in this document, we only discuss the router IP capaicy for given packet type.

We define the IP-type-P router capacity, C(R,T,I), to be the maximum number of IP-layer bits (in the formation of type P packet) that can

be correctly transfered from the ingress interfaces to the egress interfaces during the interval [T, T+I], divided by I. Like nominal physical link capacity and IP-layer link capacity, IP-type-P router capacity is also a theoretical maximum value and typically constant over time.

Note this is only a nominal value or an approximation, because the accurate IP layer router capacity depends on many factors. Any router faces the common challenge, its capacity representation depends on its atchitect design, memory (e.g. queueing) management, interface deployment and other implementation issues. for example, a router can support 1000 interfaces and the capacity of 1T bps for IP type P at best. When we configure this router with 100 interfaces/ links, we can get this capacity value (i.e., 1T bps). But if the router is configured with only one ingress interface/link and one egress interface/link, maybe the maximum capacity value this router can present is less than 1T bps, because of its internal bus structure factors, even each link has the IP layer capacity of 2T bps.

On the other hand, as link capacity is node-independent, router capacity is not dependent on bits injection. The ingress link (i.e., the link which is attached to the ingress interface) may affect how many packets are injected to the router and the egress link (i.e., the link which is attached to the egress interface) may affect how many packets are forwarded to the next hop, but note the router capacity is the maximum number that we can get in all cases, for the given type P packets.

2.3.7. Definition: IP-type-P Router Usage

The average usage of a Router R, Used(R,T,I), is the actual number of IP-layer bits (in the formation of type P packet) correctly transfered from any ingress interface to the right engree interface during the interval [T, T+I], divided by I.

An important distinction between usage and capacity is that Used(R,T,I) is not the maximum number, but rather, the actual number of IP bits that are correctly transfered.

The information forwarded through the router can be generated by any source, including those sources that are not directly attached to the router. In addition, each information flow from these sources may share the router in their respective path.

2.3.8. Definition: IP-type-P Router Utilization

We express usage as a fraction of the overall IP-layer router capacity.

Util(R,T,I) = (Used(R,T,I) / C(R,T,I))

Thus, the utilization now represents the fraction of the capacity that is being used and is a value between zero (meaning nothing is used) and one (meaning the router is fully saturated). Multiplying the utilization by 100 yields the percent utilization of the router. By using the above, we can now define the capacity available throuth the router.

2.3.9. Definition: IP-type-P Available Router Capacity

We can now determine the amount of available capacity on a congested router by multiplying the IP-layer router capacity with the complement of the IP-layer router utilization. Thus, the IP-layer available router capacity becomes:

AvailCap(R,T,I) = C(R,T,I) * (1 - Util(R,T,I))

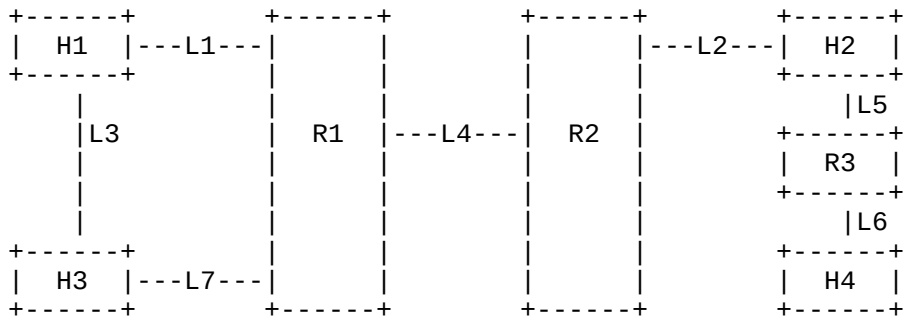
As mentioned in router capacity section, AvailCap(R,T,I) is only an approximation, because the accurate available router capacity depends on many internal factors.

2.3.10. Definition: IP-type-P Path Capacity

Using our definition for IP-layer link capacity and IP-layer router capacity, we can then extend these notions to an entire path.

We define the IP-type-P IP-layer path capacity, C(P,T,I), to be the maximum number of IP-layer bits (in the formation of type P packet) that can be correctly transfered from the source to the destination during the interval [T, T+I], divided by I. Like link capacity and router capacity, path capacity is also a theoretical number.

As mentioned earlier, the path of length n is a sequence of the form (N0, L1, N1, ..., Ln, Nn) and N1, N2, ..., Nn-1 are all routers and part of the path. But these links and routers may be part of one or multiple paths, for example, in the following scenario:



Host, Router, Link and Path

Figure 1

There are multiple paths in this network, such as:

Path P1 (from H1 to H2) -- (H1, L1, R1, L4, R2, L2, H2);
Path P2 (from H1 to H3) -- (H1, L3, H3);
Path P3 (from H1 to H3) -- (H1, L1, R1, L7, H3);
Path P4 (from H2 to H1) -- (H2, L2, R2, L4, R1, L1, H1);
Path P5 (from H2 to H3) -- (H2, L2, R2, L4, R1, L7, H3); and,
Path P6 (from H2 to H4) -- (H2, L5, R3, L6, H4).

Note this is not an exhaustive list. There are many other paths in this network, e.g., (R1, L4, R2).

In this scenario, the path (H1, L3, H4) and the path (H2, L5, R3, L6, H4) are exclusive path. The IP-layer capacity of an exclusive path may be calculated by:

C(P,T,I) = min {1..n} {C(Ln,T,I), C(Rn,T,I)}

we can also find that the link of L1, L2, L4 are all shared by multiple paths and the router of R1 and R2 are the same. Because of the capacity sharing, path capacity rather depends on the capacity contribution from the links and the routers than the IP-layer capacity of themselves. So for any given path whose link or router overlaps with other path, the IP-layer path capacity becomes more complex, it depends on not only the IP-layer capacity of the links and the routers but also the "competitive" traffic (also in formation of type P packet) of other paths, which have overlap segment with the given path. This means the capacity of non-exclusive path is a variable, is external situation dependent.

It is very difficult to calculate IP-type-P path capacity of non-exclusive path in general but we can get out the maximum number of path capacity from links and routers, to indicate the upper bound on path capacity.

The maximum number of IP-layer capacity of non-exclusive path may be calculated by:

Cmax(P,T,I) = min {1..n} {C(Ln,T,I), C(Rn,T,I)}

2.3.11. Definition: IP-type-P Available Path Capacity

Using our definition for IP-layer available link capacity and IP-layer available router capacity, we can then extend these notions to an entire path, such that the IP-layer available path capacity simply becomes that of the link and router with the smallest available capacity along that path.

AvailCap(P,T,I) = min {1..n} {AvailCap(Ln,T,I), AvailCap(Rn,T,I)}

Since measurements of available capacity are more volatile than that of link capacity, we stress the importance that both the time and interval be specified as their values have a great deal of influence on the results. In addition, a sequence of measurements may be beneficial in offsetting the volatility when attempting to characterize available capacity.

3. Changes from RFC5136

In general, this document clarifies some definitions (e.g., path) and expounds that the capacity metrics (e.g., IP-type-P link capacity) are theoretical number. In addition, usage metrics (e.g., IP-type-P link usage) are very different from capacity metrics because they are actual number represented in measurement cases.

3.1. Node Definition

Section 2.1 from [RFC5136] has been changed from:

We define nodes as hosts, routers, Ethernet switches, or any other device where the input and output links can have different characteristics.

to Section 2.1.1 of this memo:

Node is a computer that implements IP protocol.

Reason/summarization:

The reason for this modification is to follow the most idiomatic definition. Non-IP device is excluded in the notion of node.

3.2. Link Definition

Section 2.1 from [RFC5136] has been changed from:

A link is a connection between two of these network devices or nodes.

to Section 2.1.5 of this memo:

Link is a communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IPv6. Examples are Ethernets (simple or bridged); PPP links; X.25, Frame Relay, or ATM networks; and internet (or higher) layer "tunnels", such as tunnels over IPv4 or IPv6 itself.

Reason/summarization:

The reason for this modification is to clarify the notion. The connection between layer1 or layer2 devices is not an absolute link, but only a segment of link.

3.3. Path Definition

Section 2.1 from [RFC5136] has been changed from:

We then define a path P of length n as a series of links (L1, L2, ..., Ln) connecting a sequence of nodes (N1, N2, ..., Nn+1). A source S and destination D reside at N1 and Nn+1, respectively.

to Section 2.1.6 of this memo:

A path of length n is a sequence of the form (N0, L1, N1, ..., Ln, Nn), where n >= 0, each Ni is a node, each Li is a link between Ni-1 and Ni, each N1...Nn-1 is a router. A pair (Li, Ni) is termed a 'hop'. In an appropriate operational configuration, the links and routers in the path facilitate network-layer communication of packets from N0 to Nn.

Reason/summarization:

The reason for this modification is to emphasize that routers in the path are essential component of the path.

3.4. Definition: Nominal Physical Capacity

Section 2.2 from [RFC5136] has been changed from "Definition: Nominal Physical Link Capacity" to "Definition: Nominal Physical Capacity". And some statement are added, including:

Note when we define nominal physical capacity of link, link terminals are considered while the nodes (host or router) which are connected by the link are not gathered. This is because the link terminal (e.g., network interface card) is not integrant of computer, it is only an accessory of the computer. However, there may be some non-IP-node in the link, such as the ethereal switch. The physical link capacity is affected by the switch's ability to process and forward information bits for the given link.

and,

However, it is difficult to define the nominal physical capacity of a router. The routers are designed under many limitation, such as physical bound of CPU, memory and system bus. We usually use a pair of common principle to estimate a router: packet per second and bit per second. These two principles are coupled together, and in general, we almost can not correctly estimate a router by either of them. So we don't define nominal physical router capacity in this document.

3.5. IP-type-P Link Capacity

Section 2.3.2 from [RFC5136] has been changed from:

We define the IP-layer link capacity, C(L,T,I), to be the maximum number of IP-layer bits that can be transmitted from the source S and correctly received by the destination D over the link L during the interval [T, T+I], divided by I.

As mentioned earlier, this definition is affected by many factors that may change over time. For example, a device's ability to process and forward IP packets for a particular link may have varying effect on capacity, depending on the amount or type of traffic being processed.

to Section 2.3.2 of this memo:

We define the IP-layer link capacity, C(L,T,I), to be the maximum number of IP-layer bits that can be transmitted from the source S and correctly received by the destination D over the link L during the interval [T, T+I], divided by I. The "maximum" means that IP-type-P link capacity is the capacity representation when the link is fully

utilized (i.e., nominal physical link capacity is fully used.)

In theory, IP-layer link capacity may be calculated out from nominal physical link capacity. Usually, for any link whose link protocol is given, we can know well the encapsulation, overhead and overtail of the link layer protocol. In these cases, for Type P Packets, whose length is Lp, we can get IP-layer link capacity as:

C(L,T,I)
= [Lp/(Lh + Lp + Lt)] * [1 - BER(T, T+I)] * [1 - BDR(T, T+I)] * P(L)

In this formula,
- Lp denotes type P packet length (in IP layer),
- Lh denotes link layer protocol overhead length,
- Lt denotes link layer protocol overtail length,
- BER(T, T+I) denotes Block Error or Lost Rate during the interval [T, T+I],
- BDR(T, T+I) denotes Block Duplication Rate during the interval [T, T+I],
- P(L) denotes nominal physical link capacity of the given link.

Like nominal physical link capacity, IP-type-P link capacity is also a theoretical maximum value. But IP-type-P link capacity is not constant over time, becauese there are many types of link layer protocol and BER and BDR (e.g., BER/BDR of radio channel) may vary in different period.

As defined in section 2.1.5, link is the layer 2 connection between nodes, so the nodes which are connected by the link are not part of the given link. However, there may be some Non-IP-Node in the link, such as the ethereal switch. The IP-type-P link capacity is affected by the switch's ability to process and forward IP packets for the given link.

IP-type-P link capacity is affected by on-way Non-IP-Node but not affected by the nodes which are connected by the IP link. This means, the injecting node may affect how many packets are transpored between the source S and the destination D during the interval [T, T+I], and the incepting node may also affect how many packets are correctly received in the destination D, but these factors do not affect IP-type-P link capacity, because the capacity is the maximum value can be represented by the link.

IP-type-P link capacity is similar to IP-type-P link usage in some percent. The comparison is described in the next section.

Reason/summarization:

This modification clarifies the definition and calculation of link capacity and explicitly indicates that node doesn't affect link capacity but the Non-IP-Node which is part of link does.

3.6. IP-type-P Router Capacity

Section 2.3.6 is newly added in this memo, as:

As mentioned in section 2.2, we don't define nominal physical router capacity in this document, we only discuss the router IP capaicy for given packet type.

We define the IP-type-P router capacity, C(R,T,I), to be the maximum number of IP-layer bits (in the formation of type P packet) that can be correctly transfered from the ingress interfaces to the egress interfaces during the interval [T, T+I], divided by I. Like nominal physical link capacity and IP-layer link capacity, IP-type-P router capacity is also a theoretical maximum value and typically constant over time

Note this is only a nominal value or an approximation, because the accurate IP layer router capacity depends on many factors. Any router faces the common challenge, its capacity representation depends on its atchitect design, memory (e.g. queueing) management, interface deployment and other implementation issues. for example, a router can support 1000 interfaces and the capacity of 1T bps for IP type P at best. When we configure this router with 100 interfaces/ links, we can get this capacity value (i.e., 1T bps). But if the router is configured with only one ingress interface/link and one egress interface/link, maybe the maximum capacity value this router can present is less than 1T bps, because of its internal bus structure factors, even each link has the IP layer capacity of 2T bps.

On the other hand, as link capacity is node-independent, router capacity is not dependent on bits injection. The ingress link (i.e., the link which is attached to the ingress interface) may affect how many packets are injected to the router and the egress link (i.e., the link which is attached to the egress interface) may affect how many packets are forwarded to the next hop, but note the router capacity is the maximum number that we can get in all cases, for the given type P packets.

Reason/summarization:

This modification defines IP-layer router capacity aspect from network capacity.

3.7. IP-type-P Router Usage

Section 2.3.7 is newly added in this memo, as:

The average usage of a Router R, Used(R,T,I), is the actual number of IP-layer bits (in the formation of type P packet) correctly transfered from any ingress interface to the right engree interface during the interval [T, T+I], divided by I.

An important distinction between usage and capacity is that Used(R,T,I) is not the maximum number, but rather, the actual number of IP bits that are correctly transfered.

The information forwarded through the router can be generated by any source, including those sources that are not directly attached to the router. In addition, each information flow from these sources may share the router in their respective path.

Reason/summarization:

This modification defines IP-layer router usage aspect from network capacity.

3.8. IP-type-P Router Utilization

Section 2.3.8 is newly added in this memo, as:

We express usage as a fraction of the overall IP-layer router capacity.

Util(R,T,I) = (Used(R,T,I) / C(R,T,I))

Thus, the utilization now represents the fraction of the capacity that is being used and is a value between zero (meaning nothing is used) and one (meaning the router is fully saturated). Multiplying the utilization by 100 yields the percent utilization of the router. By using the above, we can now define the capacity available throuth the router.

Reason/summarization:

This modification defines IP-layer router utilization aspect from network capacity.

3.9. IP-type-P Available Router Capacity

Section 2.3.9 is newly added in this memo, as:

We can now determine the amount of available capacity on a congested router by multiplying the IP-layer router capacity with the complement of the IP-layer router utilization. Thus, the IP-layer available router capacity becomes:

AvailCap(R,T,I) = C(R,T,I) * (1 - Util(R,T,I))

As mentioned in router capacity section, AvailCap(R,T,I) is only an approximation, because the accurate available router capacity depends on many internal factors.

Reason/summarization:

This modification defines IP-layer available router capacity aspect from network capacity.

3.10. IP-type-P Path Capacity

Section 2.3.3 from [RFC5136] has been changed from:

Using our definition for IP-layer link capacity, we can then extend this notion to an entire path, such that the IP-layer path capacity simply becomes that of the link with the smallest capacity along that path.

C(P,T,I) = min {1..n} {C(Ln,T,I)}

The previous definitions specify the number of IP-layer bits that can be transmitted across a link or path should the resource be free of any congestion. It represents the full capacity available for traffic between the source and destination. Determining how much capacity is available for use on a congested link is potentially much more useful. However, in order to define the available capacity, we must first specify how much is being used.

to Section 2.3.10 of this memo:

We define the IP-type-P IP-layer path capacity, C(P,T,I), to be the maximum number of IP-layer bits (in the formation of type P packet) that can be correctly transfered from the source to the destination during the interval [T, T+I], divided by I. Like link capacity and router capacity, path capacity is also a theoretical number.

The IP-layer capacity of an exclusive path may be calculated by:

C(P,T,I) = min {1..n} {C(Ln,T,I), C(Rn,T,I)}

It is very difficult to calculate IP-type-P path capacity of non-

exclusive path in general but we can get out the maximum number of path capacity from links and routers, to indicate the upper bound on path capacity.

The maximum number of IP-layer capacity of non-exclusive path may be calculated by:

Cmax(P,T,I) = min {1..n} {C(Ln,T,I), C(Rn,T,I)}

Reason/summarization:

This modification clarifies how to correctly evaluate path capacity. Router capacity is considered for path capacity.

3.11. IP-type-P Available Path Capacity

Section 2.3.7 from [RFC5136] has been changed from:

Using our definition for IP-layer available link capacity, we can then extend this notion to an entire path, such that the IP-layer available path capacity simply becomes that of the link with the smallest available capacity along that path.

AvailCap(P,T,I) = min {1..n} {AvailCap(Ln,T,I)}

to Section 2.3.11 of this memo:

Using our definition for IP-layer available link capacity and IP-layer available router capacity, we can then extend these notions to an entire path, such that the IP-layer available path capacity simply becomes that of the link and router with the smallest available capacity along that path.

AvailCap(P,T,I) = min {1..n} {AvailCap(Ln,T,I), AvailCap(Rn,T,I)}

Reason/summarization:

This modification clarifies how to correctly evaluate available path capacity. Available router capacity is considered for available path capacity.

4. Discussion

4.1. Time and Sampling

We must emphasize the importance of time in the basic definitions of these quantities. We know that traffic on the Internet is highly

variable across all time scales. This argues that the time and length of measurements are critical variables in reporting available capacity measurements and must be reported when using these definitions.

The closer to "instantaneous" a metric is, the more important it is to have a plan for sampling the metric over a time period that is sufficiently large. By doing so, we allow valid statistical inferences to be made from the measurements. An obvious pitfall here is sampling in a way that causes bias. For example, a situation where the sampling frequency is a multiple of the frequency of an underlying condition.

4.2. Hardware Duplicates

We briefly consider the effects of paths where hardware duplication of packets may occur. In such an environment, a node in the network path may duplicate packets, and the destination may receive multiple, identical copies of these packets. Both the original packet and the duplicates can be properly received and appear to be originating from the sender. Thus, in the most generic form, duplicate IP packets are counted in these definitions. However, hardware duplication can affect these definitions depending on the use of "Type P" to add additional restrictions on packet reception. For instance, a restriction only to count uniquely-sent packets may be more useful to users concerned with capacity for meaningful data. In contrast, the more general, unrestricted metric may be suitable for a user who is concerned with raw capacity. Thus, it is up to the user to properly scope and interpret results in situations where hardware duplicates may be prevalent.

4.3. Other Potential Factors

IP encapsulation does not affect the definitions as all IP header and payload bits must be counted regardless of content. However, IP packets of different sizes can lead to a variation in the amount of overhead needed at the lower layers to transmit the data, thus altering the overall IP link-layer capacity.

Should the link happen to employ a compression scheme such as RObust Header Compression (ROHC) [RFC3095] or V.44 [V44], some of the original bits are not transmitted across the link. However, the inflated (not compressed) number of IP-layer bits should be counted.

4.4. Common Terminology in Literature

Certain terms are often used to characterize specific aspects of the presented definitions. The link with the smallest capacity is

commonly referred to as the "narrow link" of a path. Also, the link with the smallest available capacity is often referred to as the "tight link" within a path. So, while a given link may have a very large capacity, the overall congestion level on the link makes it the likely bottleneck of a connection. Conversely, a link that has the smallest capacity may not be the bottleneck should it be lightly loaded in relation to the rest of the path.

Also, literature often overloads the term "bandwidth" to refer to what we have described as capacity in this document. For example, when inquiring about the bandwidth of a 802.11b link, a network engineer will likely answer with 11 Mbit/s. However, an electrical engineer may answer with 25 MHz, and an end user may tell you that his observed bandwidth is 8 Mbit/s. In contrast, the term "capacity" is not quite as overloaded and is an appropriate term that better reflects what is actually being measured.

4.5. Comparison to Bulk Transfer Capacity (BTC)

Bulk Transfer Capacity (BTC) [RFC3148] provides a distinct perspective on path capacity that differs from the definitions in this document in several fundamental ways. First, BTC operates at the transport layer, gauging the amount of capacity available to an application that wishes to send data. Only unique data is measured, meaning header and retransmitted data are not included in the calculation. In contrast, IP-layer link capacity includes the IP header and is indifferent to the uniqueness of the data contained within the packet payload. (Hardware duplication of packets is an anomaly addressed in a previous section.) Second, BTC utilizes a single congestion-aware transport connection, such as TCP, to obtain measurements. As a result, BTC implementations react strongly to different path characteristics, topologies, and distances. Since these differences can affect the control loop (propagation delays, segment reordering, etc.), the reaction is further dependent on the algorithms being employed for the measurements. For example, consider a single event where a link suffers a large duration of bit errors. The event could cause IP-layer packets to be discarded, and the lost packets would reduce the IP-layer link capacity. However, the same event and subsequent losses would trigger loss recovery for a BTC measurement resulting in the retransmission of data and a potentially reduced sending rate. Thus, a measurement of BTC does not correspond to any of the definitions in this document. Both techniques are useful in exploring the characteristics of a network path, but from different perspectives.

5. Conclusion

In this document, we have defined a set of quantities related to the capacity of links, routers and paths in an IP network. In these definitions, we have tried to be as clear as possible and take into account various characteristics that links, routers and paths can have. The goal of these definitions is to enable researchers who propose capacity metrics to relate those metrics to these definitions and to evaluate those metrics with respect to how well they approximate these quantities.

In addition, we have pointed out some key auxiliary parameters and opened a discussion of issues related to valid inferences from available capacity metrics.

6. Security Considerations

This document specifies definitions regarding IP traffic traveling between a source and destination in an IP network. These definitions do not raise any security issues and do not have a direct impact on the networking protocol suite.

Tools that attempt to implement these definitions may introduce security issues specific to each implementation. Both active and passive measurement techniques can be abused, impacting the security, privacy, and performance of the network. Any measurement techniques based upon these definitions must include a discussion of the techniques needed to protect the network on which the measurements are being performed.

7. IANA Considerations

This document has no actions for IANA.

8. Acknowledgments

The author would especially like to acknowledge Phil Chimento and Joseph Ishac for their great contribution on the item of network capacity. The author would like to acknowledge Mark Allman, Patrik Arlos, Matt Mathis, Al Morton, Stanislav Shalunov, and Matt Zekauskas for their contribution on [RFC5136], which is the basis of this document.

The author would also like to acknowledge Brian E Carpenter, Adrian Farrel, Spencer Dawkins, David Harrington and Barry Leiba for their

review and discussion in the early stage of this document.

9. References

9.1. Normative References

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[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.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

9.2. Informative References

[RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, September 1999.

[RFC3095] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[RFC3148] Mathis, M. and M. Allman, "A Framework for Defining Empirical Bulk Transfer Capacity Metrics", RFC 3148, July 2001.

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

[RFC5136] Chimento, P. and J. Ishac, "Defining Network Capacity", RFC 5136, February 2008.

[V44] ITU Telecommunication Standardization Sector (ITU-T) Recommendation V.44, "Data Compression Procedures", November 2000.

Author's Address

Xiangsong Cui (editor)
Huawei
KuiKe Bld., No.9 Xinxi Rd., Shang-Di Information Industry Base
Beijing, 100085
P.R. China

Phone:
Email: Xiangsong.Cui@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 4, 2011

N. Duffield
AT&T Labs-Research
A. Morton
AT&T Labs
J. Sommers
Colgate University
December 31, 2010

Loss Episode Metrics for IPPM
draft-ietf-ippm-loss-episode-metrics-01

Abstract

The IETF has developed a one way packet loss metric that measures the loss rate on a Poisson probe stream between two hosts. However, the impact of packet loss on applications is in general sensitive not just to the average loss rate, but also to the way in which packet losses are distributed in loss episodes (i.e., maximal sets of consecutively lost probe packets). This draft defines one-way packet loss episode metrics, specifically the frequency and average duration of loss episodes, and a probing methodology under which the loss episode metrics are to be measured.

Requirements Language

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

Status of this Memo

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

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

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

This Internet-Draft will expire on July 4, 2011.

Copyright Notice

Table of Contents

1. Introduction

1.1. Background and Motivation

Packet loss in the Internet is a complex phenomenon due to the bursty nature of traffic and congestion processes, influenced by both end-users and applications, and the operation of transport protocols such as TCP. For these reasons, the simplest model of packet loss--the single parameter Bernoulli (independent) loss model--does not represent the complexity of packet loss over periods of time. Correspondingly, a single loss metric--the average packet loss ratio over some period of time--arising, e.g., from a stream of Poisson probes as in [RFC2680] is not sufficient to determine the effect of packet loss on traffic in general.

Moving beyond single parameter loss models, Markovian and Markov-modulated loss models involving transitions between a good and bad state, each with an associated loss rate, have been proposed by Gilbert and more generally by Elliot. In principle, Markovian models can be formulated over state spaces involving patterns of loss of any desired number of packets. However further increase in the size of the state space makes such models cumbersome both for parameter estimation (accuracy decreases) and prediction in practice (due to computational complexity and sensitivity to parameter inaccuracy). In general, the relevance and importance of particular models can change in time, e.g. in response to the advent of new applications and services. For this reason we are drawn to empirical metrics that do not depend on a particular model for their interpretation.

An empirical measure of packet loss complexity, the index of dispersion of counts (IDC), comprise, for each t >0, the ratio v(t) \ a(t) of the variance v(t) and average a(t) of the number of losses over successive measurement windows of a duration t. However, a full characterization of packet loss over time requires specification of the IDC for each window size t>0.

In the standards arena, loss pattern sample metrics are defined in [RFC3357]. Following the Gilbert-Elliot model, burst metrics specific for VoIP that characterize complete episodes of lost, transmitted and discarded packets are defined in [RFC3611]

All these considerations motivate formulating empirical metrics of one-way packet loss that provide the simplest generalization of the successful [RFC2680] that can capture deviations from independent packet loss in a robust model-independent manner, and, to define efficient measurement methodologies for these metrics.

1.2. Loss Episode Metrics and Bi-Packet Probes

The losses experienced by the packet stream can be viewed as occurring in loss episodes, i.e., maximal set of consecutively lost packets. This memo describes one-way loss episode metrics: their frequency and average duration. Although the average loss ratio can be expressed in terms of these quantities, they go further in characterizing the statistics of the patterns of packet loss within the stream of probes. This is useful information in understanding the effect of packet losses on application performance, since different applications can have different sensitivities to patterns of loss, being sensitive not only to the long term average loss rate, but how losses are distributed in time. As an example: MPEG video traffic may be sensitive to loss involving the I-frame in a group of pictures, but further losses within an episode of sufficiently short duration have no further impact; the damage is already done.

The loss episode metrics presented here represent have the following useful properties:

1. the metrics are empirical and do not depend on an underlying model; e.g., the loss process is not assumed to be Markovian. On the other hand, it turns out that the metrics of this memo can be related to the special case of the Gilbert Model parameters; see Section 7.

2. the metric units can be directly compared with applications or user requirements or tolerance for network loss performance, in the frequency and duration of loss episodes, as well as the usual packet loss ratio, which can be recovered from the loss episode metrics upon dividing the average loss episode duration by the loss episode frequency.

3. the metrics provide the smallest possible increment in complexity beyond, but in the spirit of, the IPPM average packet loss ratio metrics [RFC2680] i.e., moving from a single metric (average packet loss ratio) to a pair of metrics (loss episode frequency and average loss episode duration).

The draft also describes a probing methodology under which loss episode metrics are to be measured. The methodology comprises sending probe packets in pairs, where packets within each probe pair have a fixed separation, and the time between pairs takes the form of a geometric distributed number multiplied by the same separation. This can be regarded a generalization of Poisson probing where the probes are pairs rather than single packets as in [RFC2680], and also of geometric probing described in [RFC2330]. However, it should be distinguished from back to back packet pairs whose change in

separation on traversing a link is used to probe bandwidth. In this draft, the separation between the packets in a pair is the temporal resolution at which different loss episodes are to be distinguished. One key feature of this methodology is its efficiency: it estimates the average length of loss episodes without directly measuring the complete episodes themselves. Instead, this information is encoded in the observed relative frequencies of the 4 possible outcomes arising from the loss or successful transmission of each of the two packets of the probe pairs. This is distinct from the approach of [RFC3611] that reports on directly measured episodes.

The metrics defined in this memo are "derived metrics", according to Section 6.1 of [RFC2330] the IPPM framework. They are based on the singleton loss metric defined in Section 2 of [RFC2680] .

1.3. Outline and Contents

- Section 2 defines the fundamental singleton metric for the possible outcomes of a probe pair: Type-P-One-way-Bi-Packet-Loss.

- Section 3 defines sample sets of this metric derived from a general probe stream: Type-P-One-way-Bi-Packet-Loss-Stream.

- Section 4 defines the prime example of the Bi-Packet-Loss-Stream metrics, specifically Type-P-One-way-Bi-Packet-Loss-Geometric-Stream arising from the geometric stream of packet-pair probes that was described informally in Section 1.

- Section 5 defines Loss episode proto-metrics that summarize the outcomes from a stream metrics as an intermediate step to forming the loss episode metrics; they need not be reported in general.

- Section 6 defines the final loss episode metrics that are the focus of this memo, the new metrics

  * Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Duration, the average duration, in seconds, of a loss episode

  * Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Frequency, the average frequency, per second, at which loss episodes start.

  * Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Ratio, which is the average packet loss ratio metric arising from the geometric stream probing methodology

- Section 7 details applications and relations to existing loss models.

2. Singleton Definition for Type-P-One-way Bi-Packet Loss

2.1. Metric Name

Type-P-One-way-Bi-Packet-Loss

2.2. Metric Parameters

- Src, the IP address of a source host
- Dst, the IP address of a destination host
- T1, a sending time of the first packet
- T2, a sending time of the second packet, with T2>T1
- F, a selection function defining unambiguously the two packets from the stream selected for the metric.
- P, the specification of the packet type, over and above the source and destination addresses

2.3. Metric Units

A Loss Pair is pair (l1, l2) where each of l1 and l2 is a binary value 0 or 1, where 0 signifies successful transmission of a packet and 1 signifies loss.

The metric unit for Type-P-One-way-Bi-Packet-Loss takes is a Loss Pair

2.4. Metric Definition

1. "The Type-P-One-way-Bi-Packet-Loss with parameters (Src, Dst, T1, T2, F, P) is (1,1)" means that Src sent the first bit of a Type-P packet to Dst at wire-time T1 and the first bit of a Type-P packet to Dst a wire-time T2>T1, and that neither packet was received at Dst.
2. The Type-P-One-way-Bi-Packet-Loss with parameters (Src, Dst, T1, T2, F, P) is (1,0)" means that Src sent the first bit of a Type-P packet to Dst at wire-time T1 and the first bit of a Type-P packet to Dst a wire-time T2>T1, and that the first packet was not received at Dst, and the second packet was received at Dst

3. The Type-P-One-way-Bi-Packet-Loss with parameters (Src, Dst, T1, T2, F, P) is (0,1)" means that Src sent the first bit of a Type-P packet to Dst at wire-time T1 and the first bit of a Type-P packet to Dst a wire-time T2>T1, and that the first packet was received at Dst, and the second packet was not received at Dst

4. The Type-P-One-way-Bi-Packet-Loss with parameters (Src, Dst, T1, T2, F, P) is (0,0)" means that Src sent the first bit of a Type-P packet to Dst at wire-time T1 and the first bit of a Type-P packet to Dst a wire-time T2>T1, and that both packet were received at Dst.

2.5. Discussion

The purpose of the selection function is to specify exactly which packets are to be used for measurement. The notion is taken from Section 2.5 of [RFC3393], where examples are discussed.

2.6. Methodologies

The methodologies related to the Type-P-One-way-Packet-Loss metric in Section 2.6 of [RFC2680] are similar for the Type-P-One-way-Bi-Packet-Loss metric described above. In particular, the methodologies described in RFC 2680 apply to both packets of the pair.

2.7. Errors and Uncertainties

Sources of error for the Type-P-One-way-Packet-Loss metric in Section 2.7 of [RFC2680] apply to each packet of the pair for the Type-P-One-way-Bi-Packet-Loss metric.

2.8. Reporting the Metric

Refer to Section 2.8 of [RFC2680].

3. General Definition of samples for Type-P-One-way-Bi-Packet-Loss

Given the singleton metric for Type-P-One-way-Bi-Packet-Loss, we now define examples of samples of singletons. The basic idea is as follows. We first specify a set of times T1 < T2 <...<Tn, each of which acts as the first time of a packet pair for a single Type-P-One-way-Bi-Packet-Loss measurement. This results is a set of n metric values of Type-P-One-way-Bi-Packet-Loss.

3.1. Metric Name

Type-P-One-way-Bi-Packet-Loss-Stream

3.2. Metric Parameters

- Src, the IP address of a source host
- Dst, the IP address of a destination host
- (T11,T12), (T21,T22).....,(Tn1,Tn2) a set of n times of sending times for packet pairs, with T11 < T12 <= T21 < T22 <=...<= Tn1 < Tn2
- F, a selection function defining unambiguously the two packets from the stream selected for the metric.
- P, the specification of the packet type, over and above the source and destination address

3.3. Metric Units

A set L1,L2,...,Ln of loss pairs

3.4. Metric Definition

Each loss pair Li for i-1,....n is the Type-P-One-way-Bi-Packet-Loss with parameters (Src, Dst, Ti1, Ti2, Fi, P) where Fi is the restriction of the selection function F to the packet pair at time Ti1, Ti2.

3.5. Discussion

The metric definition of Type-P-One-way-Bi-Packet-Loss-Stream is sufficiently general to describe the case where packets are sampled from a pre-existing stream. This is useful in the case that there is a general purpose measurement stream setup between two hosts, and we which to select a substream from it for the purposes of loss episode measurement. In the next section we specialize this somewhat to more concretely describe a purpose built packet stream for loss episode measurement.

3.6. Methodologies

3.7. Errors and Uncertainties

3.8. Reporting the Metric

4. An active probing methodology for Bi-Packet Loss

This section specializes the preceding section for an active probing methodology. The basic idea is a follows. We set up a sequence of evenly spaced times T1 < T2 < ... < Tn. Each time Ti is potentially the first packet time for a packet pair measurement. We make an independent random decision at each time, whether to initiate such a measurement. Hence the interval count between successive times at which a pair is initiated follows a geometric distribution. We also specify that the spacing between successive times Ti is the same as the spacing between packets in a given pair. Thus if pairs happen to be launched at the successive times Ti T(i+1), the second packet of the first pair is actually used as the first packet of the second pair.

4.1. Metric Name

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream

4.2. Metric Parameters

- Src, the IP address of a source host
- Dst, the IP address of a destination host
- T0, the randomly selected starting time [RFC3432] for periodic launch opportunities
- d, the time spacing between potential launch times, Ti and Ti+1
- n, a count of potential measurement instants
- q, a launch probability
- F, a selection function defining unambiguously the two packets from the stream selected for the metric.
- P, the specification of the packet type, over and above the source and destination address

4.3. Metric Units

A set of Loss Pairs L1, L2, ..., Lm for some m <= n

4.4. Metric Definition

for each i = 0, 1, ..., n-1 we form the potential measurement time Ti = T + i * d. With probability q, a packet pair measurement is launched at Ti, resulting in a Type-P-One-way-Bi-Packet-Loss with parameters (Src, Dst, Ti, Ti+1, Fi, P) where Fi is the restriction of the selection function F to the packet pair at times Ti, Ti+1. L1, L2,...Lm are the resulting Loss Pairs; m can be less than n since not all time Ti have an associated measurement.

4.5. Discussion

The above definition of Type-P-One-way-Bi-Packet-Loss-Geometric-Stream is equivalent to using Type-P-One-way-Bi-Packet-Loss-Stream with an appropriate statistical definition of the selection function F.

The number m of loss pairs in the metric can be less than the number of potential measurement instants because not all instants may generate a probe when the launch probability q is strictly less than 1.

4.6. Methodologies

The methodologies follow from:

- the specific time T0, from which all successive Ti follow, and
- the specific time spacing, and
- the methodologies discussion given above for the singleton Type-P-One-way-Bi-Packet-Loss metric.

The issue of choosing an appropriate time spacing (e.g., one that is matched to expected characteristics of loss episodes) is outside the scope of this document.

Note that as with any active measurement methodology, consideration must be made to handle out-of-order arrival of packets; see also Section 3.6. of [RFC2680].

4.7. Errors and Uncertainties

In addition to sources of errors and uncertainties related to methodologies for measuring the singleton Type-P-One-way-Bi-Packet-Loss metric, a key source of error when emitting packets for Bi-Packet Loss relates to resource limits on the host used to send the packets. In particular, the choice of T0, the choice of the time spacing, and the choice of the launch probability results in a schedule for sending packets. Insufficient CPU resources on the sending host may result in an inability to send packets according to schedule. Note that the choice of time spacing directly affects the ability of the host CPU to meet the required schedule (e.g., consider a 100 microsecond spacing versus a 100 millisecond spacing).

For other considerations, refer to Section 3.7. [RFC2680].

4.8. Reporting the Metric

Refer to Section 3.8. of [RFC2680].

5. Loss Epsiode Proto-Metrics

This section describes four generic proto-metric quantities associated with an arbitrary set of loss pairs. These are the Loss-Pair-Counts, Bi-Packet-Loss-Ratio, Bi-Packet-Loss-Episode-Duration-Number, Bi-Packet-Loss-Episode-Frequency-Number. Specific loss episode metrics can then be constructed when these proto metrics take as their input, sets of loss pairs samples generated by the Type-P-One-way-Bi-Packet-Loss-Stream and Type-P-One-way-Bi-Packet-Loss-Geometric Stream. The second of these is described in Section 4. It is not expected that these proto-metrics would be reported themselves. Rather they are intermediate quantities in the production of the final metrics of Section 6 below, and could be rolled up into them in implementations. The metrics report loss episode durations and frequencies in terms of packet counts, since they do not depend on the actual time between probe packets. The final metrics of Section 6 incorporate timescales and yield durations in seconds, and frequencies as per second.

5.1. Loss-Pair-Counts

Loss-Pair-Counts are the absolute frequencies of the 4 types of loss pair outcome in a sample. More precisely, the Loss-Pair-Counts associated with a set of loss pairs L1,,,,Ln are the numbers N(i,j) of such loss pairs that take each possible value (i,j) in the set ((0,0), (0,1), (1,0), (1,1)).

5.2. Bi-Packet-Loss-Ratio

The Bi-Packet-loss-ratio associated with a set of n loss pairs L1,,,,Ln is defined in terms of their Loss-Pair-Counts by the quantity (N(1,0) +N(1,1))/n.

Note this is formally equivalent to the loss metric Type-P-One-way-Packet-Loss-Average from[RFC2680] since it averages single packet losses.

5.3. Bi-Packet-Loss-Episode-Duration-Number

The Bi-Packet-Loss-Episode-Duration-Number associated with a set of n loss pairs L1,,,,Ln is defined in terms of their Loss-Pair-Counts in the following cases:

- 2*(N(0,1) + N(1,0) + N(1,1)/ (N(0,1)+N(1,0)) - 1 if N(0,1) + N(1,0) >1
- 0 if N(0,1) + N(1,0) + N(1,1) = 0 (no probe packets lost)
- Undefined if N(0,1) + N(1,0) + N(0,0) = 0 (all probe packets lost)

Note N(0,1) + N(1,0) is zero if there are no transitions between loss and no-loss outcomes.

5.4. Bi-Packet-Loss-Episode-Frequency-Number

The Bi-Packet-Loss-Episode-Frequency-Number associated with a set of n loss pairs L1,,,,Ln is defined in terms of their Loss-Pair-Counts as Bi-Packet-Loss-Ratio / Bi-Packet-Loss-Episode-Duration-Number, when this can be defined, specifically, it is:

- (N(1,0)+N(1,1)) * (N(0,1)+N(1,0)) / (2*N(1,1)+N(0,1)+N(1,0)) / n if N(0,1)+N(0,1) > 0
- 0 if N(0,1)+N(1,0) +N(1,1) = 0 (no probe packets lost)
- 1 if N(0,1) +N(1,0) +N(0,0) = 0 (all probe packets lost)

6. Loss Episode Metrics derived from Bi-Packet Loss Probing

Metrics for the time frequency and time duration of loss episodes are now defined as functions of set of n loss pairs L1,....,Ln. Although a loss episode is defined as a maximal set of successive lost packets, the loss episode metrics are not defined directly in terms of the sequential patterns of packet loss exhibited by loss pairs.

This is because samples, including Type-P-One-way-Bi-Packet-Loss-Geometric-Stream, generally do not report all lost packets in each episode. Instead, the metrics are defined as functions of the Loss-Pair-Counts of the sample, for reasons that are now described.

Consider an idealized Type-P-One-way-Bi-Packet-Loss-Geometric-Stream sample in which the launch probability q =1. It is shown in [SBDR08] that the average number of packets in a loss episode of this ideal sample is exactly the Bi-Packet-Loss-Episode-Duration derived from its set of loss pairs. Note this computation makes no reference to the position of lost packet in the sequence of probes.

A general Type-P-One-way-Bi-Packet-Loss-Geometric-Stream sample with launch probability q < 1, independently samples, with probability q, each loss pair of an idealized sample. On average, the Loss-Pair-Counts (if normalized by the total number of pairs) will be the same as in the idealized sample. The loss episode metrics in the general case are thus estimators of those for the idealized case; the statistical properties of this estimation, including a derivation of the estimation variance, is provided in [SBDR08].

6.1. Geometric Stream: Loss Ratio

6.1.1. Metric Name

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Ratio

6.1.2. Metric Parameters

- Src, the IP address of a source host
- Dst, the IP address of a destination host
- T0, the randomly selected starting time [RFC3432] for periodic launch opportunities
- d, the time spacing between potential launch times, Ti and Ti+1
- n, a count of potential measurement instants
- q, a launch probability
- F, a selection function defining unambiguously the two packets from the stream selected for the metric.
- P, the specification of the packet type, over and above the source and destination address

6.1.3. Metric Units

A number in the interval [0,1]

6.1.4. Metric Definition

The result obtained by computing the Bi-Packet-Loss-Ratio over a Type-P-One-way-Bi-Packet-Loss-Geometric-Stream sample with the metric parameters.

6.1.5. Discussion

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Ratio estimates the fraction of packets lost from the geometric stream of Bi-Packet probes.

6.1.6. Methodologies

Refer to Section 4.6

6.1.7. Errors and Uncertainties

Because Type-P-One-way-Bi-Packet-Loss-Geometric-Stream is sampled in general (when the launch probability q <1) the metrics described in this Section can be regarded as statistical estimators of the corresponding idealized version corresponding to q = 1. Estimation variance as it applies to Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Loss-Ratio is described in [SBDR08].

For other issues refer to Section 4.7

6.1.8. Reporting the Metric

Refer to Section 4.8

6.2. Geometric Steam: Loss Episode Duration

6.2.1. Metric Name

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Duration

6.2.2. Metric Parameters

- Src, the IP address of a source host
- Dst, the IP address of a destination host

- o T0, the randomly selected starting time [RFC3432] for periodic launch opportunities
- o d, the time spacing between potential launch times, Ti and Ti+1
- o n, a count of potential measurement instants
- o q, a launch probability
- o F, a selection function defining unambiguously the two packets from the stream selected for the metric.
- o P, the specification of the packet type, over and above the source and destination address

6.2.3. Metric Units

A non-negative number of seconds.

6.2.4. Metric Definition

The result obtained by computing the Bi-Packet-Loss-Episode-Duration-Number over a Type-P-One-way-Bi-Packet-Loss-Geometric-Stream sample with the metric parameters, then multiplying the result by the launch spacing parameter d.

6.2.5. Discussion

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Duration estimates the average duration of a loss episode, measured in seconds. The duration measured in packets is obtained by dividing the metric value by the packet launch spacing parameter d.

6.2.6. Methodologies

Refer to Section 4.6

6.2.7. Errors and Uncertainties

Because Type-P-One-way-Bi-Packet-Loss-Geometric-Stream is sampled in general (when the launch probability q <1) the metrics described in this Section can be regarded as statistical estimators of the corresponding idealized version corresponding to q = 1. Estimation variance as it applies to Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Duration is described in [SBDR08].

For other issues refer to Section 4.7

6.2.8. Reporting the Metric

Refer to Section 4.8

6.3. Geometric Stream: Loss Episode Frequency

6.3.1. Metric Name

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Frequency

6.3.2. Metric Parameters

- Src, the IP address of a source host
- Dst, the IP address of a destination host
- T0, the randomly selected starting time [RFC3432] for periodic launch opportunities
- d, the time spacing between potential launch times, Ti and Ti+1
- n, a count of potential measurement instants
- q, a launch probability
- F, a selection function defining unambiguously the two packets from the stream selected for the metric.
- P, the specification of the packet type, over and above the source and destination address

6.3.3. Metric Units

A positive number.

6.3.4. Metric Definition

The result obtained by computing the Bi-Packet-Loss-Episode-Frequency over a Type-P-One-way-Bi-Packet-Loss-Geometric-Stream sample with the metric parameters, then dividing he result by the launch spacing parameter d.

6.3.5. Discussion

Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Frequency estimates the average frequency per unit time with which loss episodes start (or finish). The frequency relative to the count of potential probe launches is obtained by multiplying the metric value

by the packet launch spacing parameter d.

6.3.6. Methodologies

Refer toSection 4.6

6.3.7. Errors and Uncertainties

Because Type-P-One-way-Bi-Packet-Loss-Geometric-Stream is sampled in general (when the launch probability q <1) the metrics described in this Section can be regarded as statistical estimators of the corresponding idealized version corresponding to q = 1. Estimation variance as it applies to Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Frequency is described in [SBDR08].

For other issues refer to Section 4.7

6.3.8. Reporting the Metric

Refer to Section 4.8

7. Applicability of Loss Episode Metrics

7.1. Relation to Gilbert Model

The general Gilbert-Elliot model is a discrete time Markov chain over two states, Good (g) and Bad (b), each with its own independent packet loss rate. In the simplest case, the Good loss rate is 0 while the Bad loss rate is 1. Correspondingly, there are two independent parameters, the Markov transition probabilities P(g|b) = 1- P(b|b) and P(b|g) = 1- P(g|g), where P(i|j) is the probability to transition from state j and step n to state i at step n+1. With these parameters, the fraction of steps spent in the bad state is P(b|g)/(P(b|g) + P(g|b)) while the average duration of a sojourn in the bad state is 1/P(g|b) steps.

Now identify the steps of the Markov chain with the possible sending times of packets for a Type-P-One-way-Bi-Packet-Loss-Geometric-Stream with launch spacing d. Suppose the loss episode metrics Type-P-One-way-Bi-Packet-Loss-Geometric-Stream-Ratio and ype-P-One-way-Bi-Packet-Loss-Geometric-Stream-Episode-Duration take the values r and m respectively. Then from the discussion in Section 6.2.5 the following can be equated:

r = P(b|g)/(P(b|g) + P(g|b)) and m/d = 1/P(g|b).

These relationships can be inverted in order to recover the Gilbert

model parameters:

P(g|b) = d/m and P(b|g)=d/m/(1/r - 1)

8. IPR Considerations

IPR disclosures concerning some of the material covered in this draft has been made to the IETF: see https://datatracker.ietf.org/ipr/1009/ , https://datatracker.ietf.org/ipr/1010/ , and https://datatracker.ietf.org/ipr/1126/

9. Security Considerations

Conducting Internet measurements raises both security and privacy concerns. This memo does not specify an implementation of the metrics, so it does not directly affect the security of the Internet nor of applications which run on the Internet. However,implementations of these metrics must be mindful of security and privacy concerns.

There are two types of security concerns: potential harm caused by the measurements, and potential harm to the measurements. The measurements could cause harm because they are active, and inject packets into the network. The measurement parameters MUST be carefully selected so that the measurements inject trivial amounts of additional traffic into the networks they measure. If they inject "too much" traffic, they can skew the results of the measurement, and in extreme cases cause congestion and denial of service. The measurements themselves could be harmed by routers giving measurement traffic a different priority than "normal" traffic, or by an attacker injecting artificial measurement traffic. If routers can recognize measurement traffic and treat it separately, the measurements may not reflect actual user traffic. If an attacker injects artificial traffic that is accepted as legitimate, the loss rate will be artificially lowered. Therefore, the measurement methodologies SHOULD include appropriate techniques to reduce the probability that measurement traffic can be distinguished from "normal" traffic. Authentication techniques, such as digital signatures, may be used where appropriate to guard against injected traffic attacks. The privacy concerns of network measurement are limited by the active measurements described in this memo: they involve no release of user data.

10. IANA Considerations

11. Acknowledgements

12. References

12.1. Normative References

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

[RFC2680] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, 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.

[RFC3611] Friedman, T., Caceres, R., and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, November 2003.

12.2. Informative References

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

[RFC3357] Koodli, R. and R. Ravikanth, "One-way Loss Pattern Sample Metrics", RFC 3357, August 2002.

[SBDR08] IEEE/ACM Transactions on Networking, 16(2): 307-320, "A Geometric Approach to Improving Active Packet Loss Measurement", 2008.

Authors' Addresses

Nick Duffield
AT&T Labs-Research
180 Park Avenue
Florham Park, NJ 07932
USA

Phone: +1 973 360 8726
Fax: +1 973 360 8871
Email: duffield@research.att.com
URI: http://www.research.att.com/people/Duffield_Nicholas_G

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/

Joel Sommers
Colgate University
304 McGregory Hall
Hamilton, NY 13346
USA

Phone: +1 315 228 7587
Fax:
Email: jsommers@colgate.edu
URI: http://cs.colgate.edu/faculty/jsommers

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: September 15, 2011

R. Geib, Ed.
Deutsche Telekom
A. Morton
AT&T Labs
R. Fardid
Cariden Technologies
A. Steinmitz
HS Fulda

March 14, 2011

IPPM standard advancement testing
draft-ietf-ippm-metrictest-02

Abstract

This document specifies tests to determine if multiple independent instantiations of a performance metric RFC have implemented the specifications in the same way. This is the performance metric equivalent of interoperability, required to advance RFCs along the standards track. Results from different implementations of metric RFCs will be collected under the same underlying network conditions and compared using state of the art statistical methods. The goal is an evaluation of the metric RFC itself, whether its definitions are clear and unambiguous to implementors and therefore a candidate for advancement on the IETF standards track.

Status of this Memo

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

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

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

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Table of Contents

1. Introduction

The Internet Standards Process RFC2026 [RFC2026] requires that for a IETF specification to advance beyond the Proposed Standard level, at least two genetically unrelated implementations must be shown to interoperate correctly with all features and options. This requirement can be met by supplying:

- o evidence that (at least a sub-set of) the specification has been implemented by multiple parties, thus indicating adoption by the IETF community and the extent of feature coverage.

- o evidence that each feature of the specification is sufficiently well-described to support interoperability, as demonstrated through testing and/or user experience with deployment.

In the case of a protocol specification, the notion of "interoperability" is reasonably intuitive - the implementations must successfully "talk to each other", while exercising all features and options. To achieve interoperability, two implementors need to interpret the protocol specifications in equivalent ways. In the case of IP Performance Metrics (IPPM), this definition of interoperability is only useful for test and control protocols like the One-Way Active Measurement Protocol, OWAMP [RFC4656], and the Two-Way Active Measurement Protocol, TWAMP [RFC5357].

A metric specification RFC describes one or more metric definitions, methods of measurement and a way to report the results of measurement. One example would be a way to test and report the One-way Delay that data packets incur while being sent from one network location to another, One-way Delay Metric.

In the case of metric specifications, the conditions that satisfy the "interoperability" requirement are less obvious, and there was a need for IETF agreement on practices to judge metric specification "interoperability" in the context of the IETF Standards Process. This memo provides methods which should be suitable to evaluate metric specifications for standards track advancement. The methods proposed here MAY be generally applicable to metric specification RFCs beyond those developed under the IPPM Framework [RFC2330].

Since many implementations of IP metrics are embedded in measurement systems that do not interact with one another (they were built before OWAMP and TWAMP), the interoperability evaluation called for in the IETF standards process cannot be determined by observing that independent implementations interact properly for various protocol exchanges. Instead, verifying that different implementations give statistically equivalent results under controlled measurement

conditions takes the place of interoperability observations. Even when evaluating OWAMP and TWAMP RFCs for standards track advancement, the methods described here are useful to evaluate the measurement results because their validity would not be ascertained in typical interoperability testing.

The standards advancement process aims at producing confidence that the metric definitions and supporting material are clearly worded and unambiguous, or reveals ways in which the metric definitions can be revised to achieve clarity. The process also permits identification of options that were not implemented, so that they can be removed from the advancing specification. Thus, the product of this process is information about the metric specification RFC itself: determination of the specifications or definitions that are clear and unambiguous and those that are not (as opposed to an evaluation of the implementations which assist in the process).

This document defines a process to verify that implementations (or practically, measurement systems) have interpreted the metric specifications in equivalent ways, and produce equivalent results.

Testing for statistical equivalence requires ensuring identical test setups (or awareness of differences) to the best possible extent. Thus, producing identical test conditions is a core goal of the memo. Another important aspect of this process is to test individual implementations against specific requirements in the metric specifications using customized tests for each requirement. These tests can distinguish equivalent interpretations of each specific requirement.

Conclusions on equivalence are reached by two measures.

First, implementations are compared against individual metric specifications to make sure that differences in implementation are minimised or at least known.

Second, a test setup is proposed ensuring identical networking conditions so that unknowns are minimized and comparisons are simplified. The resulting separate data sets may be seen as samples taken from the same underlying distribution. Using state of the art statistical methods, the equivalence of the results is verified. To illustrate application of the process and methods defined here, evaluation of the One-way Delay Metric [RFC2679] is provided in an Appendix. While test setups will vary with the metrics to be validated, the general methodology of determining equivalent results will not. Documents defining test setups to evaluate other metrics should be developed once the process proposed here has been agreed and approved.

The metric RFC advancement process begins with a request for protocol action accompanied by a memo that documents the supporting tests and results. The procedures of [RFC2026] are expanded in[RFC5657], including sample implementation and interoperability reports. Section 3 of [morton-advance-metrics-01] can serve as a template for a metric RFC report which accompanies the protocol action request to the Area Director, including description of the test set-up, procedures, results for each implementation and conclusions.

Changes from WG-01 to WG-02:

- Clarification of the number of test streams recommended in section 3.2.
- Clarifications on testing details in sections 3.3 and 3.4.
- Spelling corrections throughout.

Changes from WG -00 to WG -01 draft

- Discussion on merits and requirements of a distributed lab test using only local load generators.
- Proposal of metrics suitable for tests using the proposed measurement configuration.
- Hint on delay caused by software based L2TPv3 implementation.
- Added an appendix with a test configuration allowing remote tests comparing different implementations across the network.
- Proposal for maximum error of "equivalence", based on performance comparison of identical implementations. This may be useful for both ADK and non-ADK comparisons.

Changes from prior ID -02 to WG -00 draft

- Incorporation of aspects of reporting to support the protocol action request in the Introduction and section 3.5
- Overhaul of section 3.2 regarding tunneling: Added generic tunneling requirements and L2TPv3 as an example tunneling mechanism fulfilling the tunneling requirements. Removed and adapted some of the prior references to other tunneling protocols
- Softened a requirement within section 3.4 (MUST to SHOULD on precision) and removed some comments of the authors.

- Updated contact information of one author and added a new author.
- Added example C++ code of an Anderson-Darling two sample test implementation.

Changes from ID -01 to ID -02 version

- Major editorial review, rewording and clarifications on all contents.
- Additional text on parallel testing using VLANs and GRE or Pseudowire tunnels.
- Additional examples and a glossary.

Changes from ID -00 to ID -01 version

- Addition of a comparison of individual metric implementations against the metric specification (trying to pick up problems and solutions for metric advancement [morton-advance-metrics]).
- More emphasis on the requirement to carefully design and document the measurement setup of the metric comparison.
- Proposal of testing conditions under identical WAN network conditions using IP in IP tunneling or Pseudo Wires and parallel measurement streams.
- Proposing the requirement to document the smallest resolution at which an ADK test was passed by 95%. As no minimum resolution is specified, IPPM metric compliance is not linked to a particular performance of an implementation.
- Reference to RFC 2330 and RFC 2679 for the 95% confidence interval as preferred criterion to decide on statistical equivalence
- Reducing the proposed statistical test to ADK with 95% confidence.

1.1. Requirements Language

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

2. Basic idea

The implementation of a standard compliant metric is expected to meet

the requirements of the related metric specification. So before comparing two metric implementations, each metric implementation is individually compared against the metric specification.

Most metric specifications leave freedom to implementors on non-fundamental aspects of an individual metric (or options). Comparing different measurement results using a statistical test with the assumption of identical test path and testing conditions requires knowledge of all differences in the overall test setup. Metric specification options chosen by implementors have to be documented. It is REQUIRED to use identical implementation options wherever possible for any test proposed here. Calibrations proposed by metric standards should be performed to further identify (and possibly reduce) potential sources of errors in the test setup.

The Framework for IP Performance Metrics [RFC2330] expects that a "methodology for a metric should have the property that it is repeatable: if the methodology is used multiple times under identical conditions, it should result in consistent measurements." This means an implementation is expected to repeatedly measure a metric with consistent results (repeatability with the same result). Small deviations in the test setup are expected to lead to small deviations in results only. To characterise statistical equivalence in the case of small deviations, RFC 2330 and [RFC2679] suggest to apply a 95% confidence interval. Quoting RFC 2679, "95 percent was chosen because ... a particular confidence level should be specified so that the results of independent implementations can be compared."

Two different implementations are expected to produce statistically equivalent results if they both measure a metric under the same networking conditions. Formulating in statistical terms: separate metric implementations collect separate samples from the same underlying statistical process (the same network conditions). The statistical hypothesis to be tested is the expectation that both samples do not expose statistically different properties. This requires careful test design:

- The measurement test setup must be self-consistent to the largest possible extent. To minimize the influence of the test and measurement setup on the result, network conditions and paths MUST be identical for the compared implementations to the largest possible degree. This includes both the stability and non-ambiguity of routes taken by the measurement packets. See RFC 2330 for a discussion on self-consistency.

- The error induced by the sample size must be small enough to minimize its influence on the test result. This may have to be respected, especially if two implementations measure with

different average probing rates.

- Every comparison must be repeated several times based on different measurement data to avoid random indications of compatibility (or the lack of it).

- To minimize the influence of implementation options on the result, metric implementations SHOULD use identical options and parameters for the metric under evaluation.

- The implementation with the lowest probing frequency determines the smallest temporal interval for which samples can be compared.

The metric specifications themselves are the primary focus of evaluation, rather than the implementations of metrics. The documentation produced by the advancement process should identify which metric definitions and supporting material were found to be clearly worded and unambiguous, OR, it should identify ways in which the metric specification text should be revised to achieve clarity and unified interpretation.

The process should also permit identification of options that were not implemented, so that they can be removed from the advancing specification (this is an aspect more typical of protocol advancement along the standards track).

Note that this document does not propose to base interoperability indications of performance metric implementations on comparisons of individual singletons. Individual singletons may be impacted by many statistical effects while they are measured. Comparing two singletons of different implementations may result in failures with higher probability than comparing samples.

3. Verification of conformance to a metric specification

This section specifies how to verify compliance of two or more IPPM implementations against a metric specification. This document only proposes a general methodology. Compliance criteria to a specific metric implementation need to be defined for each individual metric specification. The only exception is the statistical test comparing two metric implementations which are simultaneously tested. This test is applicable without metric specific decision criteria.

Several testing options exist to compare two or more implementations:

- o Use a single test lab to compare the implementations and emulate the Internet with an impairment generator.

- o Use a single test lab to compare the implementations and measure across the Internet.

- o Use remotely separated test labs to compare the implementations and emulate the Internet with two "identically" configured impairment generators.

- o Use remotely separated test labs to compare the implementations and measure across the Internet.

- o Use remotely separated test labs to compare the implementations and measure across the Internet and include a single impairment generator to impact all measurement flows in non discriminatory way.

The first two approaches work, but cause higher expenses than the other ones (due to travel and/or shipping+installation). For the third option, ensuring two identically configured impairment generators requires well defined test cases and possibly identical hard- and software. >>>Comment: for some specific tests, impairment generator accuracy requirements are less-demanding than others, and in such cases there is more flexibility in impairment generator configuration. <<<

It is a fair question, whether the last two options can result in any applicable test set up at all. While an experimental approach is given in Appendix C, the trade off that measurement packets of different sites pass the path segments but always in a different order of segments probably can’t be avoided.

The question of which option above results in identical networking conditions and is broadly accepted can’t be answered without more practical experience in comparing implementations. The last proposal has the advantage that, while the measurement equipment is remotely distributed, a single network impairment generator and the Internet can be used in combination to impact all measurement flows.

3.1. Tests of an individual implementation against a metric specification

A metric implementation MUST support the requirements classified as "MUST" and "REQUIRED" of the related metric specification to be compliant to the latter.

Further, supported options of a metric implementation SHOULD be

documented in sufficient detail. The documentation of chosen options is RECOMMENDED to minimise (and recognise) differences in the test setup if two metric implementations are compared. Further, this documentation is used to validate and improve the underlying metric specification option, to remove options which saw no implementation or which are badly specified from the metric specification to be promoted to a standard. This documentation SHOULD be made for all implementation-relevant specifications of a metric picked for a comparison that are not explicitly marked as "MUST" or "REQUIRED" in the RFC text. This applies for the following sections of all metric specifications:

- Singleton Definition of the Metric.
- Sample Definition of the Metric.
- Statistics Definition of the Metric. As statistics are compared by the test specified here, this documentation is required even in the case, that the metric specification does not contain a Statistics Definition.
- Timing and Synchronisation related specification (if relevant for the Metric).
- Any other technical part present or missing in the metric specification, which is relevant for the implementation of the Metric.

RFC2330 and RFC2679 emphasise precision as an aim of IPPM metric implementations. A single IPPM conformant implementation MUST under otherwise identical network conditions produce precise results for repeated measurements of the same metric.

RFC 2330 prefers the "empirical distribution function" EDF to describe collections of measurements. RFC 2330 determines, that "unless otherwise stated, IPPM goodness-of-fit tests are done using 5% significance." The goodness of fit test determines by which precision two or more samples of a metric implementation belong to the same underlying distribution (of measured network performance events). The goodness of fit test to be applied is the Anderson-Darling K sample test (ADK sample test, K stands for the number of samples to be compared) [ADK]. Please note that RFC 2330 and RFC 2679 apply an Anderson Darling goodness of fit test too.

The results of a repeated test with a single implementation MUST pass an ADK sample test with confidence level of 95%. The resolution for which the ADK test has been passed with the specified confidence level MUST be documented. To formulate this differently: The

requirement is to document the smallest resolution, at which the results of the tested metric implementation pass an ADK test with a confidence level of 95%. The minimum resolution available in the reported results from each implementation MUST be taken into account in the ADK test.

3.2. Test setup resulting in identical live network testing conditions

Two major issues complicate tests for metric compliance across live networks under identical testing conditions. One is the general point that metric definition implementations cannot be conveniently examined in field measurement scenarios. The other one is more broadly described as "parallelism in devices and networks", including mechanisms like those that achieve load balancing (see [RFC4928]).

This section proposes two measures to deal with both issues. Tunneling mechanisms can be used to avoid parallel processing of different flows in the network. Measuring by separate parallel probe flows results in repeated collection of data. If both measures are combined, WAN network conditions are identical for a number of independent measurement flows, no matter what the network conditions are in detail.

Any measurement setup MUST be made to avoid the probing traffic itself to impede the metric measurement. The created measurement load MUST NOT result in congestion at the access link connecting the measurement implementation to the WAN. The created measurement load MUST NOT overload the measurement implementation itself, e.g., by causing a high CPU load or by creating imprecisions due to internal transmit (receive respectively) probe packet collisions.

Tunneling multiple flows reaching a network element on a single physical port may allow to transmit all packets of the tunnel via the same path. Applying tunnels to avoid undesired influence of standard routing for measurement purposes is a concept known from literature, see e.g. GRE encapsulated multicast probing [GU+Duffield]. An existing IP in IP tunnel protocol can be applied to avoid Equal-Cost Multi-Path (ECMP) routing of different measurement streams if it meets the following criteria:

- Inner IP packets from different measurement implementations are mapped into a single tunnel with single outer IP origin and destination address as well as origin and destination port numbers which are identical for all packets.

- An easily accessible commodity tunneling protocol allows to carry out a metric test from more test sites.

- o A low operational overhead may enable a broader audience to set up a metric test with the desired properties.

- o The tunneling protocol should be reliable and stable in set up and operation to avoid disturbances or influence on the test results.

- o The tunneling protocol should not incur any extra cost for those interested in setting up a metric test.

An illustration of a test setup with two tunnels and two flows between two linecards of one implementation is given in Figure 1.

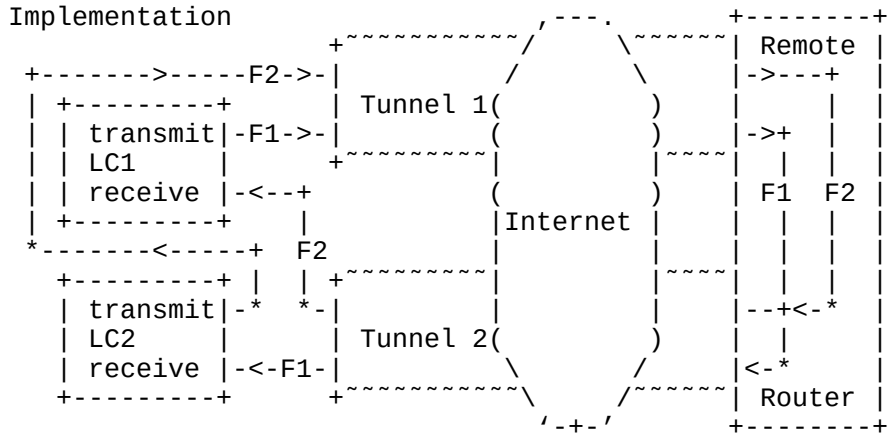


Illustration of a test setup with two tunnels. For simplicity, only two linecards of one implementation and two flows F between them are shown.

Figure 1

Figure 2 shows the network elements required to set up GRE tunnels or as shown by figure 1.

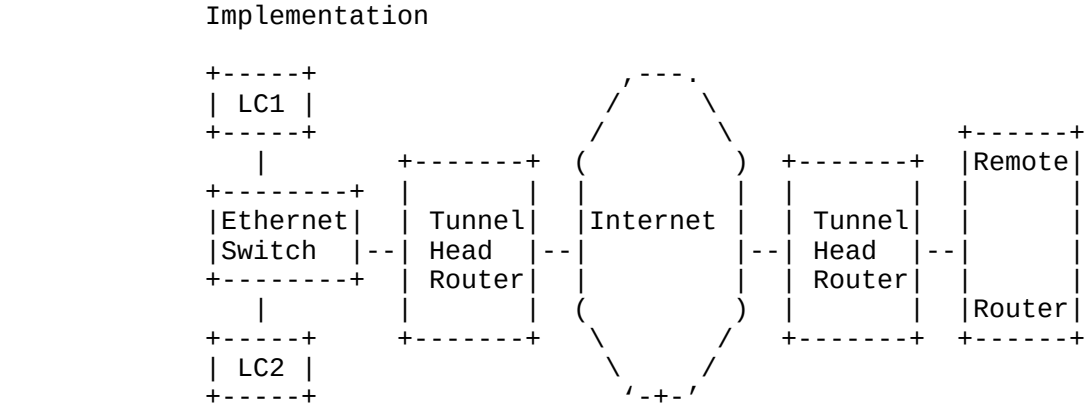


Illustration of a hardware setup to realise the test setup illustrated by figure 1 with GRE tunnels or Pseudowires.

Figure 2

If tunneling is applied, two tunnels MUST carry all test traffic in between the test site and the remote site. For example, if 802.1Q Ethernet Virtual LANs (VLAN) are applied and the measurement streams are carried in different VLANs, the IP tunnel or Pseudo Wires respectively MUST be set up in physical port mode to avoid set up of Pseudo Wires per VLAN (which may see different paths due to ECMP routing), see RFC 4448. The remote router and the Ethernet switch shown in figure 2 must support 802.1Q in this set up.

The IP packet size of the metric implementation SHOULD be chosen small enough to avoid fragmentation due to the added Ethernet and tunnel headers. Otherwise, the impact of tunnel overhead on fragmentation and interface MTU size MUST be understood and taken into account (see [RFC4459]).

An Ethernet port mode IP tunnel carrying several 802.1Q VLANs each containing measurement traffic of a single measurement system was set up as a proof of concept using RFC4719 [RFC4719], Transport of Ethernet Frames over L2TPv3. Ethernet over L2TPv3 seems to fulfill most of the desired tunneling protocol criteria mentioned above.

The following headers may have to be accounted for when calculating total packet length, if VLANs and Ethernet over L2TPv3 tunnels are applied:

- Ethernet 802.1Q: 22 Byte.
- L2TPv3 Header: 4-16 Byte for L2TPv3 data messages over IP; 16-28 Byte for L2TPv3 data messages over UDP.

- IPv4 Header (outer IP header): 20 Byte.

- MPLS Labels may be added by a carrier. Each MPLS Label has a length of 4 Bytes. By the time of writing, between 1 and 4 Labels seems to be a fair guess of what's expectable.

The applicability of one or more of the following tunneling protocols may be investigated by interested parties if Ethernet over L2TPv3 is felt to be not suitable: IP in IP [RFC2003] or Generic Routing Encapsulation (GRE) [RFC2784]. RFC 4928 [RFC4928] proposes measures how to avoid ECMP treatment in MPLS networks.

L2TP is a commodity tunneling protocol [RFC2661]. By the time of writing, L2TPv3 [RFC3931]is the latest version of L2TP. If L2TPv3 is applied, software based implementations of this protocol are not suitable for the test set up, as such implementations may cause incalculable delay shifts.

Ethernet Pseudo Wires may also be set up on MPLS networks [RFC4448]. While there's no technical issue with this solution, MPLS interfaces are mostly found in the network provider domain. Hence not all of the above tunneling criteria are met.

Appendix C provides an experimental tunneling set up for metric implementation testing between two (or more) remote sites.

Each test SHOULD be conducted multiple times. Sequential testing is possible, but may not be a useful metric test option because WAN conditions are likely to change over time. It is RECOMMENDED that tests be carried out by establishing at least 2 different parallel measurement flows. Two linecards per implementation that send and receive measurement flows should be sufficient to create 4 parallel measurement flows (when each card sends and receives 2 flows). Other options are to separate flows by DiffServ marks (without deploying any QoS in the inner or outer tunnel) or using a single CBR flow and evaluating every n-th singleton to belong to a specific measurement flow.

Some additional rules to calculate and compare samples have to be respected to perform a metric test:

- To compare different probes of a common underlying distribution in terms of metrics characterising a communication network requires to respect the temporal nature for which the assumption of common underlying distribution may hold. Any singletons or samples to be compared MUST be captured within the same time interval.

- Whenever statistical events like singletons or rates are used to characterise measured metrics of a time-interval, at least 5 singletons of a relevant metric SHOULD be present to ensure a minimum confidence into the reported value (see Wikipedia on confidence [Rule of thumb]). Note that this criterion also is to be respected e.g. when comparing packet loss metrics. Any packet loss measurement interval to be compared with the results of another implementation SHOULD contain at least five lost packets to have a minimum confidence that the observed loss rate wasn't caused by a small number of random packet drops.

- The minimum number of singletons or samples to be compared by an Anderson-Darling test SHOULD be 100 per tested metric implementation. Note that the Anderson-Darling test detects small differences in distributions fairly well and will fail for high number of compared results (RFC2330 mentions an example with 8192 measurements where an Anderson-Darling test always failed).

- Generally, the Anderson-Darling test is sensitive to differences in the accuracy or bias associated with varying implementations or test conditions. These dissimilarities may result in differing averages of samples to be compared. An example may be different packet sizes, resulting in a constant delay difference between compared samples. Therefore samples to be compared by an Anderson-Darling test MAY be calibrated by the difference of the average values of the samples. Any calibration of this kind MUST be documented in the test result.

3.3. Tests of two or more different implementations against a metric specification

RFC2330 expects "a methodology for a given metric [to] exhibit continuity if, for small variations in conditions, it results in small variations in the resulting measurements. Slightly more precisely, for every positive epsilon, there exists a positive delta, such that if two sets of conditions are within delta of each other, then the resulting measurements will be within epsilon of each other." A small variation in conditions in the context of the metric test proposed here can be seen as different implementations measuring the same metric along the same path.

IPPM metric specifications however allow for implementor options to the largest possible degree. It can not be expected that two implementors pick identical value ranges in options for the implementations. Implementors SHOULD to the highest degree possible pick the same configurations for their systems when comparing their implementations by a metric test.

In some cases, a goodness of fit test may not be possible or show disappointing results. To clarify the difficulties arising from different implementation options, the individual options picked for every compared implementation SHOULD be documented in sufficient detail. Based on this documentation, the underlying metric specification should be improved before it is promoted to a standard.

The same statistical test as applicable to quantify precision of a single metric implementation MUST be used to compare metric result equivalence for different implementations. To document compatibility, the smallest measurement resolution at which the compared implementations passed the ADK sample test MUST be documented.

For different implementations of the same metric, "variations in conditions" are reasonably expected. The ADK test comparing samples of the different implementations MAY result in a lower precision than the test for precision in the same-implementation comparison.

3.4. Clock synchronisation

Clock synchronization effects require special attention. Accuracy of one-way active delay measurements for any metrics implementation depends on clock synchronization between the source and destination of tests. Ideally, one-way active delay measurement (RFC 2679, [RFC2679]) test endpoints either have direct access to independent GPS or CDMA-based time sources or indirect access to nearby NTP primary (stratum 1) time sources, equipped with GPS receivers. Access to these time sources may not be available at all test locations associated with different Internet paths, for a variety of reasons out of scope of this document.

When secondary (stratum 2 and above) time sources are used with NTP running across the same network, whose metrics are subject to comparative implementation tests, network impairments can affect clock synchronization, distort sample one-way values and their interval statistics. It is RECOMMENDED to discard sample one-way delay values for any implementation, when one of the following reliability conditions is met:

- Delay is measured and is finite in one direction, but not the other.
- Absolute value of the difference between the sum of one-way measurements in both directions and round-trip measurement is greater than X% of the latter value.

Examination of the second condition requires RTT measurement for

reference, e.g., based on TWAMP (RFC5357, RFC 5357 [RFC5357]), in conjunction with one-way delay measurement.

Specification of X% to strike a balance between identification of unreliable one-way delay samples and misidentification of reliable samples under a wide range of Internet path RTTs probably requires further study.

An implementation of an RFC that requires synchronized clocks is expected to provide precise measurement results in order to claim that the metric measured is compliant.

IF an implementation publishes a specification of its precision, such as "a precision of 1 ms (+/- 500 us) with a confidence of 95%", then the specification SHOULD be met over a useful measurement duration. For example, if the metric is measured along an Internet path which is stable and not congested, then the precision specification SHOULD be met over durations of an hour or more.

3.5. Recommended Metric Verification Measurement Process

In order to meet their obligations under the IETF Standards Process the IESG must be convinced that each metric specification advanced to Draft Standard or Internet Standard status is clearly written, that there are the a sufficient number of verified equivalent implementations, and that all options have been implemented.

In the context of this document, metrics are designed to measure some characteristic of a data network. An aim of any metric definition should be that it should be specified in a way that can reliably measure the specific characteristic in a repeatable way across multiple independent implementations.

Each metric, statistic or option of those to be validated MUST be compared against a reference measurement or another implementation by at least 5 different basic data sets, each one with sufficient size to reach the specified level of confidence, as specified by this document.

Finally, the metric definitions, embodied in the text of the RFCs, are the objects that require evaluation and possible revision in order to advance to the next step on the standards track.

IF two (or more) implementations do not measure an equivalent metric as specified by this document,

AND sources of measurement error do not adequately explain the lack of agreement,

THEN the details of each implementation should be audited along with the exact definition text, to determine if there is a lack of clarity that has caused the implementations to vary in a way that affects the correspondence of the results.

IF there was a lack of clarity or multiple legitimate interpretations of the definition text,

THEN the text should be modified and the resulting memo proposed for consensus and (possible) advancement along the standards track.

Finally, all the findings MUST be documented in a report that can support advancement on the standards track, similar to those described in [RFC5657]. The list of measurement devices used in testing satisfies the implementation requirement, while the test results provide information on the quality of each specification in the metric RFC (the surrogate for feature interoperability).

The complete process of advancing a metric specification to a standard as defined by this document is illustrated in Figure 3.

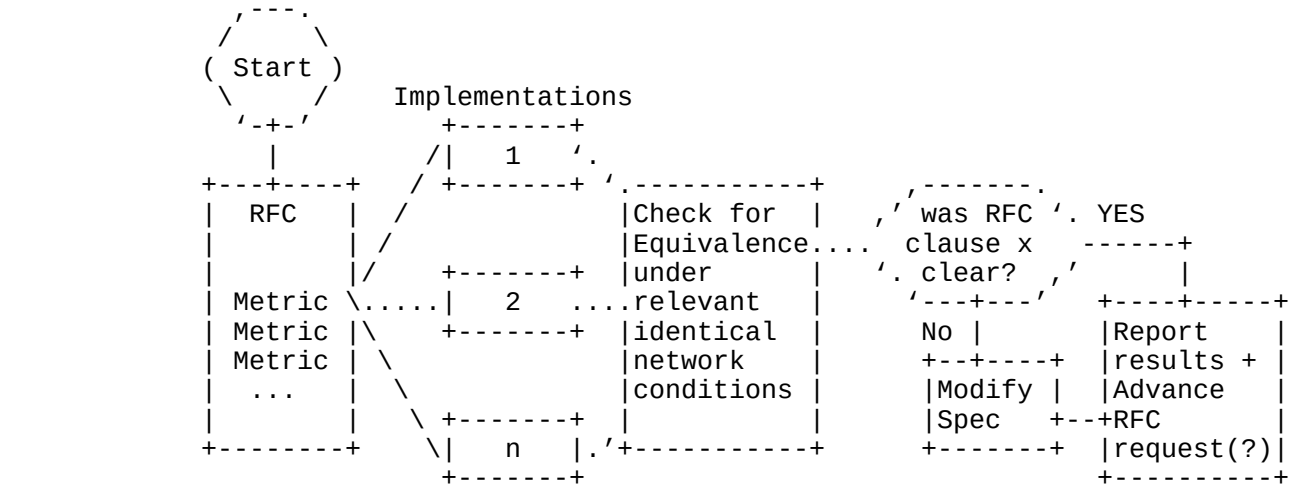


Illustration of the metric standardisation process

Figure 3

Any recommendation for the advancement of a metric specification MUST

be accompanied by an implementation report, as is the case with all requests for the advancement of IETF specifications. The implementation report needs to include the tests performed, the applied test setup, the specific metrics in the RFC and reports of the tests performed with two or more implementations. The test plan needs to specify the precision reached for each measured metric and thus define the meaning of "statistically equivalent" for the specific metrics being tested.

Ideally, the test plan would co-evolve with the development of the metric, since that’s when people have the most context in their thinking regarding the different subtleties that can arise.

In particular, the implementation report MUST as a minimum document:

- The metric compared and the RFC specifying it. This includes statements as required by the section "Tests of an individual implementation against a metric specification" of this document.

- The measurement configuration and setup.

- A complete specification of the measurement stream (mean rate, statistical distribution of packets, packet size or mean packet size and their distribution), DSCP and any other measurement stream properties which could result in deviating results. Deviations in results can be caused also if chosen IP addresses and ports of different implementations can result in different layer 2 or layer 3 paths due to operation of Equal Cost Multi-Path routing in an operational network.

- The duration of each measurement to be used for a metric validation, the number of measurement points collected for each metric during each measurement interval (i.e. the probe size) and the level of confidence derived from this probe size for each measurement interval.

- The result of the statistical tests performed for each metric validation as required by the section "Tests of two or more different implementations against a metric specification" of this document.

- A parameterization of laboratory conditions and applied traffic and network conditions allowing reproduction of these laboratory conditions for readers of the implementation report.

- The documentation helping to improve metric specifications defined by this section.

All of the tests for each set SHOULD be run in a test setup as specified in the section "Test setup resulting in identical live network testing conditions."

If a different test set up is chosen, it is RECOMMENDED to avoid effects falsifying results of validation measurements caused by real data networks (like parallelism in devices and networks). Data networks may forward packets differently in the case of:

- Different packet sizes chosen for different metric implementations. A proposed countermeasure is selecting the same packet size when validating results of two samples or a sample against an original distribution.

- Selection of differing IP addresses and ports used by different metric implementations during metric validation tests. If ECMP is applied on IP or MPLS level, different paths can result (note that it may be impossible to detect an MPLS ECMP path from an IP endpoint). A proposed counter measure is to connect the measurement equipment to be compared by a NAT device, or establishing a single tunnel to transport all measurement traffic The aim is to have the same IP addresses and port for all measurement packets or to avoid ECMP based local routing diversion by using a layer 2 tunnel.

- Different IP options.

- Different DSCP.

- If the N measurements are captured using sequential measurements instead of simultaneous ones, then the following factors come into play: Time varying paths and load conditions.

3.6. Miscellaneous

A minimum amount of singletons per metric is required if results are to be compared. To avoid accidental singletons from impacting a metric comparison, a minimum number of 5 singletons per compared interval was proposed above. Commercial Internet service is not operated to reliably create enough rare events of singletons to characterize bad measurement engineering or bad implementations. In the case that a metric validation requires capturing rare events, an impairment generator may have to be added to the test set up. Inclusion of an impairment generator and the parameterisation of the impairments generated MUST be documented.

A metric characterising a common impairment condition would be one, which by expectation creates a singleton result for each measured

packet. Delay or Delay Variation are examples of this type, and in such cases, the Internet may be used to compare metric implementations.

Rare events are those, where by expectation no or a rather low number of "event is present" singletons are captured during a measurement interval. Packet duplications, packet loss rates above one digit percentages, loss patterns and packet reordering are examples. Note especially that a packet reordering or loss pattern metric implementation comparison may require a more sophisticated test set up than described here. Spatial and temporal effects combine in the case of packet re-ordering and measurements with different packet rates may always lead to different results.

As specified above, 5 singletons are the recommended basis to minimise interference of random events with the statistical test proposed by this document. In the case of ratio measurements (like packet loss), the underlying sum of basic events, against the which the metric's monitored singletons are "rated", determines the resolution of the test. A packet loss statistic with a resolution of 1% requires one packet loss statistic-data point to consist of 500 delay singletons (of which at least 5 were lost). To compare EDFs on packet loss requires one hundred such statistics per flow. That means, all in all at least 50 000 delay singletons are required per single measurement flow. Live network packet loss is assumed to be present during main traffic hours only. Let this interval be 5 hours. The required minimum rate of a single measurement flow in that case is 2.8 packets/sec (assuming a loss of 1% during 5 hours). If this measurement is too demanding under live network conditions, an impairment generator should be used.

3.7. Proposal to determine an "equivalence" threshold for each metric evaluated

This section describes a proposal for maximum error of "equivalence", based on performance comparison of identical implementations. This comparison may be useful for both ADK and non-ADK comparisons.

Each metric tested by two or more implementations (cross-implementation testing).

Each metric is also tested twice simultaneously by the *same* implementation, using different Src/Dst Address pairs and other differences such that the connectivity differences of the cross-implementation tests are also experienced and measured by the same implementation.

Comparative results for the same implementation represent a bound on

cross-implementation equivalence. This should be particularly useful when the metric does *not* produces a continuous distribution of singleton values, such as with a loss metric, or a duplication metric. Appendix A indicates how the ADK will work for 0ne-way delay, and should be likewise applicable to distributions of delay variation.

Proposal: the implementation with the largest difference in homogeneous comparison results is the lower bound on the equivalence threshold, noting that there may be other systematic errors to account for when comparing between implementations.

Thus, when evaluating equivalence in cross-implementation results:

Maximum_Error = Same_Implementation_Error + Systematic_Error

and only the systematic error need be decided beforehand.

In the case of ADK comparison, the largest same-implementation resolution of distribution equivalence can be used as a limit on cross-implementation resolutions (at the same confidence level).

4. Acknowledgements

Gerhard Hasslinger commented a first version of this document, suggested statistical tests and the evaluation of time series information. Henk Uijterwaal and Lars Eggert have encouraged and helped to orgainize this work. Mike Hamilton, Scott Bradner, David Mcdysan and Emile Stephan commented on this draft. Carol Davids reviewed the 01 version of the ID before it was promoted to WG draft.

5. Contributors

Scott Bradner, Vern Paxson and Allison Mankin drafted bradner-metrictest [bradner-metrictest], and major parts of it are included in this document.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

This draft does not raise any specific security issues.

8. References

8.1. Normative References

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, October 1996.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[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.

[RFC2661] Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn, G., and B. Palter, "Layer Two Tunneling Protocol "L2TP"", RFC 2661, August 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.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000.

[RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

[RFC4448] Martini, L., Rosen, E., El-Aawar, N., and G. Heron, "Encapsulation Methods for Transport of Ethernet over MPLS Networks", RFC 4448, April 2006.

[RFC4459] Savola, P., "MTU and Fragmentation Issues with In-the-Network Tunneling", RFC 4459, April 2006.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.

[RFC4719] Aggarwal, R., Townsley, M., and M. Dos Santos, "Transport of Ethernet Frames over Layer 2 Tunneling Protocol Version 3 (L2TPv3)", RFC 4719, November 2006.

[RFC4928] Swallow, G., Bryant, S., and L. Andersson, "Avoiding Equal Cost Multipath Treatment in MPLS Networks", BCP 128, RFC 4928, June 2007.

[RFC5657] Dusseault, L. and R. Sparks, "Guidance on Interoperation and Implementation Reports for Advancement to Draft Standard", BCP 9, RFC 5657, September 2009.

8.2. Informative References

[ADK] Scholz, F. and M. Stephens, "K-sample Anderson-Darling Tests of fit, for continuous and discrete cases", University of Washington, Technical Report No. 81, May 1986.

[GU+Duffield] Gu, Y., Duffield, N., Breslau, L., and S. Sen, "GRE Encapsulated Multicast Probing: A Scalable Technique for Measuring One-Way Loss", SIGMETRICS'07 San Diego, California, USA, June 2007.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.

[Rule of thumb] Hardy, M., "Confidence interval", March 2010.

[bradner-metrictest] Bradner, S., Mankin, A., and V. Paxson, "Advancement of metrics specifications on the IETF Standards Track", draft -bradner-metricstest-03, (work in progress), July 2007.

[morton-advance-metrics] Morton, A., "Problems and Possible Solutions for Advancing Metrics on the Standards Track", draft -morton-ippm-advance-metrics-00, (work in progress), July 2009.

[morton-advance-metrics-01] Morton, A., "Lab Test Results for Advancing Metrics on the Standards Track", draft -morton-ippm-advance-metrics-01, (work in progress), June 2010.

Appendix A. An example on a One-way Delay metric validation

The text of this appendix is not binding. It is an example how parts of a One-way Delay metric test could look like. http://xml.resource.org/public/rfc/bibxml/

A.1. Compliance to Metric specification requirements

One-way Delay, Loss threshold, RFC 2679

This test determines if implementations use the same configured maximum waiting time delay from one measurement to another under different delay conditions, and correctly declare packets arriving in excess of the waiting time threshold as lost. See Section 3.5 of RFC2679, 3rd bullet point and also Section 3.8.2 of RFC2679.

(1) Configure a path with 1 sec one-way constant delay.

(2) Measure one-way delay with 2 or more implementations, using identical waiting time thresholds for loss set at 2 seconds.

(3) Configure the path with 3 sec one-way delay.

(4) Repeat measurements.

(5) Observe that the increase measured in step 4 caused all packets to be declared lost, and that all packets that arrive successfully in step 2 are assigned a valid one-way delay.

One-way Delay, First-bit to Last bit, RFC 2679

This test determines if implementations register the same relative increase in delay from one measurement to another under different delay conditions. This test tends to cancel the sources of error which may be present in an implementation. See Section 3.7.2 of RFC2679, and Section 10.2 of RFC2330.

(1) Configure a path with X ms one-way constant delay, and ideally including a low-speed link.

(2) Measure one-way delay with 2 or more implementations, using identical options and equal size small packets (e.g., 100 octet IP payload).

(3) Maintain the same path with X ms one-way delay.

(4) Measure one-way delay with 2 or more implementations, using identical options and equal size large packets (e.g., 1500 octet IP payload).

(5) Observe that the increase measured in steps 2 and 4 is equivalent to the increase in ms expected due to the larger serialization time for each implementation. Most of the measurement errors in each system should cancel, if they are stationary.

One-way Delay, RFC 2679

This test determines if implementations register the same relative increase in delay from one measurement to another under different delay conditions. This test tends to cancel the sources of error which may be present in an implementation. This test is intended to evaluate measurments in sections 3 and 4 of RFC2679.

(1) Configure a path with X ms one-way constant delay.

(2) Measure one-way delay with 2 or more implementations, using identical options.

(3) Configure the path with X+Y ms one-way delay.

(4) Repeat measurements.

(5) Observe that the increase measured in steps 2 and 4 is ~Y ms for each implementation. Most of the measurement errors in each system should cancel, if they are stationary.

Error Calibration, RFC 2679

This is a simple check to determine if an implementation reports the error calibration as required in Section 4.8 of RFC2679. Note that the context (Type-P) must also be reported.

A.2. Examples related to statistical tests for One-way Delay

A one way delay measurement may pass an ADK test with a timestamp resultion of 1 ms. The same test may fail, if timestamps with a resolution of 100 microseconds are eavluated. The implementation then is then conforming to the metric specification up to a timestamp resolution of 1 ms.

Let’s assume another one way delay measurement comparison between implementation 1, probing with a frequency of 2 probes per second and implementation 2 probing at a rate of 2 probes every 3 minutes. To

ensure reasonable confidence in results, sample metrics are calculated from at least 5 singletons per compared time interval. This means, sample delay values are calculated for each system for identical 6 minute intervals for the whole test duration. Per 6 minute interval, the sample metric is calculated from 720 singletons for implementation 1 and from 6 singletons for implementation 2. Note, that if outliers are not filtered, moving averages are an option for an evaluation too. The minimum move of an averaging interval is three minutes in this example.

The data in table 1 may result from measuring One-Way Delay with implementation 1 (see column Implemnt_1) and implementation 2 (see column implemnt_2). Each data point in the table represents a (rounded) average of the sampled delay values per interval. The resolution of the clock is one micro-second. The difference in the delay values may result eg. from different probe packet sizes.

Implemnt_1	Implemnt_2	Implemnt_2 - Delta_Averages
5000	6549	4997
5008	6555	5003
5012	6564	5012
5015	6565	5013
5019	6568	5016
5022	6570	5018
5024	6573	5021
5026	6575	5023
5027	6577	5025
5029	6580	5028
5030	6585	5033
5032	6586	5034
5034	6587	5035
5036	6588	5036
5038	6589	5037
5039	6591	5039
5041	6592	5040
5043	6599	5047
5046	6606	5054
5054	6612	5060

Table 1

Average values of sample metrics captured during identical time intervals are compared. This excludes random differences caused by differing probing intervals or differing temporal distance of singletons resulting from their Poisson distributed sending times.

In the example, 20 values have been picked (note that at least 100 values are recommended for a single run of a real test). Data must be ordered by ascending rank. The data of Implemnt_1 and Implemnt_2 as shown in the first two columns of table 1 clearly fails an ADK test with 95% confidence.

The results of Implemnt_2 are now reduced by difference of the averages of column 2 (rounded to 6581 us) and column 1 (rounded to 5029 us), which is 1552 us. The result may be found in column 3 of table 1. Comparing column 1 and column 3 of the table by an ADK test shows, that the data contained in these columns passes an ADK tests with 95% confidence.

>>> Comment: Extensive averaging was used in this example, because of the vastly different sampling frequencies. As a result, the distributions compared do not exactly align with a metric in [RFC2679], but illustrate the ADK process adequately.

Appendix B. Anderson-Darling 2 sample C++ code

```
/* Routines for computing the Anderson-Darling 2 sample
 * test statistic.
 *
 * Implemented based on the description in
 * "Anderson-Darling K Sample Test" Heckert, Alan and
 * Filliben, James, editors, Dataplot Reference Manual,
 * Chapter 15 Auxiliary, NIST, 2004.
 * Official Reference by 2010
 * Heckert, N. A. (2001). Dataplot website at the
 * National Institute of Standards and Technology:
 * http://www.itl.nist.gov/div898/software/dataplot.html/
 * June 2001.
 */

#include <iostream>
#include <fstream>
#include <vector>
#include <sstream>

using namespace std;

vector<double> vec1, vec2;
double adk_result;
double adk_criterium = 1.993;

/* vec1 and vec2 to be initialised with sample 1 and
```

```
             * sample 2 values in ascending order.
             */

            /* example for iterating the vectors
             * for(vector<double>::iterator it = vec1->begin();
             * it != vec1->end(); it++
             * {
             * cout << *it << endl;
             * }
             */

            static int k, val_st_z_samp1, val_st_z_samp2,
                       val_eq_z_samp1, val_eq_z_samp2,
                           j, n_total, n_sample1, n_sample2, L,
                       max_number_samples, line, maxnumber_z;
            static int column_1, column_2;
            static double adk, n_value, z, sum_adk_samp1,
                   sum_adk_samp2, z_aux;
            static double H_j, F1j, hj, F2j, denom_1_aux, denom_2_aux;
            static bool next_z_sample2, equal_z_both_samples;
            static int stop_loop1, stop_loop2, stop_loop3,old_eq_line2,
                   old_eq_line1;

            static double adk_criterium = 1.993;

            k = 2;
            n_sample1 = vec1->size() - 1;
            n_sample2 = vec2->size() - 1;

            // -1 because vec[0] is a dummy value

            n_total = n_sample1 + n_sample2;

            /* value equal to the line with a value = zj in sample 1.
             * Here j=1, so the line is 1.
             */

            val_eq_z_samp1 = 1;

            /* value equal to the line with a value = zj in sample 2.
             * Here j=1, so the line is 1.
             */

            val_eq_z_samp2 = 1;

            /* value equal to the last line with a value < zj
             * in sample 1. Here j=1, so the line is 0.
             */
```

```
            val_st_z_samp1 = 0;

            /* value equal to the last line with a value < zj
             * in sample 1. Here j=1, so the line is 0.
             */

            val_st_z_samp2 = 0;

            sum_adk_samp1 = 0;
            sum_adk_samp2 = 0;
            j = 1;

            // as mentioned above, j=1

            equal_z_both_samples = false;
            next_z_sample2 = false;

            //assuming the next z to be of sample 1

            stop_loop1 = n_sample1 + 1;

            // + 1 because vec[0] is a dummy, see n_sample1 declaration

            stop_loop2 = n_sample2 + 1;
            stop_loop3 = n_total + 1;

            /* The required z values are calculated until all values
             * of both samples have been taken into account. See the
             * lines above for the stoploop values. Construct required
             * to avoid a mathematical operation in the While condition
             */

             while (((stop_loop1 > val_eq_z_samp1)
                   || (stop_loop2 > val_eq_z_samp2)) && stop_loop3 > j)
               {
               if(val_eq_z_samp1 < n_sample1+1)
                   {

            /* here, a preliminary zj value is set.
             * See below how to calculate the actual zj.
             */

                    z = (*vec1)[val_eq_z_samp1];

            /* this while sequence calculates the number of values
             * equal to z.
             */
```

```
                while ((val_eq_z_samp1+1 < n_sample1)
                        && z == (*vec1)[val_eq_z_samp1+1] )
                        {
                        val_eq_z_samp1++;
                        }
                }
                else
                {
                val_eq_z_samp1 = 0;
                val_st_z_samp1 = n_sample1;

        // this should be val_eq_z_samp1 - 1 = n_sample1
                }

        if(val_eq_z_samp2 < n_sample2+1)
                {
                z_aux = (*vec2)[val_eq_z_samp2];;

        /* this while sequence calculates the number of values
         * equal to z_aux
         */

                while ((val_eq_z_samp2+1 < n_sample2)
                        && z_aux == (*vec2)[val_eq_z_samp2+1] )
                        {
                        val_eq_z_samp2++;
                        }

        /* the smaller of the two actual data values is picked
         * as the next zj.
         */

           if(z > z_aux)
                        {
                        z = z_aux;
                        next_z_sample2 = true;
                        }
                 else
                        {
                        if (z == z_aux)
                        {
                        equal_z_both_samples = true;
                        }

        /* This is the case, if the last value of column1 is
         * smaller than the remaining values of column2.
         */
                        if (val_eq_z_samp1 == 0)
```

```
                        {
                        z = z_aux;
                        next_z_sample2 = true;
                        }
                    }
                  }
                else
                  {
                  val_eq_z_samp2 = 0;
                  val_st_z_samp2 = n_sample2;

              // this should be val_eq_z_samp2 - 1 = n_sample2

                  }

              /* in the following, sum j = 1 to L is calculated for
               * sample 1 and sample 2.
               */

                 if (equal_z_both_samples)
                   {

              /* hj is the number of values in the combined sample
               * equal to zj
               */
                        hj = val_eq_z_samp1 - val_st_z_samp1
                     + val_eq_z_samp2 - val_st_z_samp2;

              /* H_j is the number of values in the combined sample
               * smaller than zj plus one half the the number of
               * values in the combined sample equal to zj
               * (that's hj/2).
               */

                        H_j = val_st_z_samp1 + val_st_z_samp2
                              + hj / 2;

              /* F1j is the number of values in the 1st sample
               * which are less than zj plus one half the number
               * of values in this sample which are equal to zj.
               */

                        F1j = val_st_z_samp1 + (double)
                    (val_eq_z_samp1 - val_st_z_samp1) / 2;

              /* F2j is the number of values in the 1st sample
               * which are less than zj     plus one half the number
               * of values in this sample which are equal to zj.
```

```
 */
                F2j = val_st_z_samp2 + (double)
        (val_eq_z_samp2 - val_st_z_samp2) / 2;

      /* set the line of values equal to zj to the
       * actual line of the last value picked for zj.
       */
                val_st_z_samp1 = val_eq_z_samp1;

      /* Set the line of values equal to zj to the actual
       * line of the last value picked for zjof each
       * sample. This is required as data smaller than zj
       * is accounted differently than values equal to zj.
       */

           val_st_z_samp2 = val_eq_z_samp2;

      /* next the lines of the next values z, ie. zj+1
       * are addressed.
       */

        val_eq_z_samp1++;

      /* next the lines of the next values z, ie.
       * zj+1 are addressed
       */

                val_eq_z_samp2++;
                }
           else
                {

      /* the smaller z value was contained in sample 2,
       * hence this value is the zj to base the following
       * calculations on.
       */
                if (next_z_sample2)
                {

      /* hj is the number of values in the combined
       * sample equal to zj, in this case these are
       * within sample 2 only.
       */
                hj = val_eq_z_samp2 - val_st_z_samp2;

      /* H_j is the number of values in the combined sample
       * smaller than zj plus one half the the number of
       * values in the combined sample equal to zj
```

```
             * (that's hj/2).
             */

                            H_j = val_st_z_samp1 + val_st_z_samp2
                          + hj / 2;

            /* F1j is the number of values in the 1st sample which
             * are less than zj plus one half the number of values in
             * this sample which are equal to zj.
             * As val_eq_z_samp2 < val_eq_z_samp1, these are the
             * val_st_z_samp1 only.
             */
                        F1j = val_st_z_samp1;

            /* F2j is the number of values in the 1st sample which
             * are less than zj plus one half the number of values in
             * this sample which are equal to zj. The latter are from
             * sample 2 only in this case.
             */

                 F2j = val_st_z_samp2 + (double)
                      (val_eq_z_samp2 - val_st_z_samp2) / 2;

            /* Set the line of values equal to zj to the actual line
             * of the last value picked for zj of sample 2 only in
             * this case.
             */
                            val_st_z_samp2 = val_eq_z_samp2;

            /* next the line of the next value z, ie. zj+1 is
             * addressed. Here, only sample 2 must be addressed.
             */

                 val_eq_z_samp2++;
                                if (val_eq_z_samp1 == 0)
                                {
                                val_eq_z_samp1 = stop_loop1;
                                }
                        }

            /* the smaller z value was contained in sample 2,
             * hence this value is the zj to base the following
             * calculations on.
             */

                        else
                        {
```

```
            /* hj is the number of values in the combined
             * sample equal to zj, in this case these are
             * within sample 1 only.
             */
                        hj = val_eq_z_samp1 - val_st_z_samp1;

            /* H_j is the number of values in the combined
             * sample smaller than zj plus one half the the number
             * of values in the combined sample equal to zj
             * (that's hj/2).
             */

                 H_j = val_st_z_samp1 + val_st_z_samp2
                       + hj / 2;

            /* F1j is the number of values in the 1st sample which
             * are less than zj plus, in this case these are within
             * sample 1 only one half the number of values in this
             * sample which are equal to zj. The latter are from
             * sample 1 only in this case.
             */

                 F1j = val_st_z_samp1 + (double)
                      (val_eq_z_samp1 - val_st_z_samp1) / 2;

            /* F2j is the number of values in the 1st sample which
             * are less than zj plus one half the number of values
             * in this sample which are equal to zj. As
             * val_eq_z_samp1 < val_eq_z_samp2, these are the
             * val_st_z_samp2 only.
             */

                        F2j = val_st_z_samp2;

            /* Set the line of values equal to zj to the actual line
             * of the last value picked for zj of sample 1 only in
             * this case
             */

                 val_st_z_samp1 = val_eq_z_samp1;

            /* next the line of the next value z, ie. zj+1 is
             * addressed. Here, only sample 1 must be addressed.
             */
                        val_eq_z_samp1++;

                        if (val_eq_z_samp2 == 0)
                                {
```

```
                                val_eq_z_samp2 = stop_loop2;
                                }
                        }
                        }

                denom_1_aux = n_total * F1j - n_sample1 * H_j;
                denom_2_aux = n_total * F2j - n_sample2 * H_j;

                sum_adk_samp1 = sum_adk_samp1 + hj
                        * (denom_1_aux * denom_1_aux) /
                                        (H_j * (n_total - H_j)
                        - n_total * hj / 4);
                sum_adk_samp2 = sum_adk_samp2 + hj
                        * (denom_2_aux * denom_2_aux) /
                                        (H_j * (n_total - H_j)
                        - n_total * hj / 4);

                next_z_sample2 = false;
                equal_z_both_samples = false;

        /* index to count the z. It is only required to prevent
         * the while slope to execute endless
         */
                j++;
                }

        // calculating the adk value is the final step.

        adk_result = (double) (n_total - 1) / (n_total
                * n_total * (k - 1))
                * (sum_adk_samp1 / n_sample1
                + sum_adk_samp2 / n_sample2);


        /* if(adk_result <= adk_criterium)
         * adk_2_sample test is passed
         */
```

Figure 4

Appendix C. A tunneling set up for remote metric implementation testing

Parties interested in testing metric compliance is most convenient if all involved parties can stay in their local test laboratories. Figure 4 shows a test configuration which may enable remote metric compliance testing.

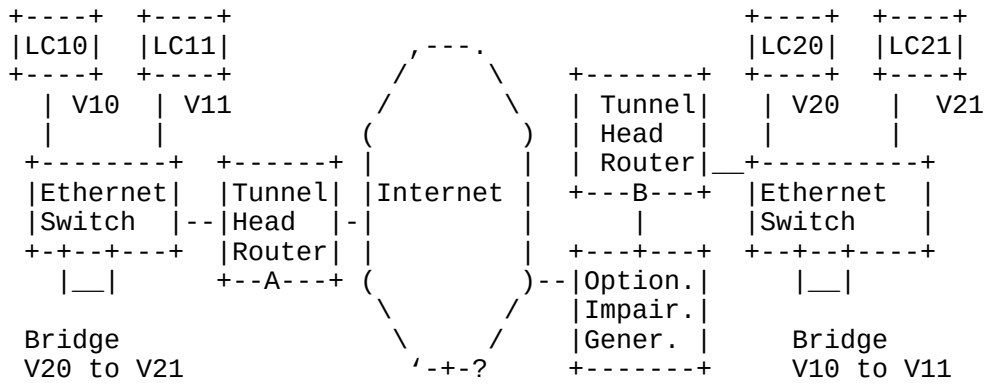


Figure 5

LC10 identify measurement clients /line cards. V10 and the others denote VLANs. All VLANs are using the same tunnel from A to B and in the reverse direction. The remote site VLANs are U-bridged at the local site Ethernet switch. The measurement packets of site 1 travel tunnel A->B first, are U-bridged at site 2 and travel tunnel B->A second. Measurement packets of site 2 travel tunnel B->A first, are U-bridged at site 1 and travel tunnel A->B second. So all measurement packets pass the same tunnel segments, but in different segment order. An experiment to prove or reject the above test set up shown in figure 4 has been agreed but not yet scheduled between Deutsche Telekom and RIPE.

Figure 4 includes an optional impairment generator. If this impairment generator is inserted in the IP path between the tunnel head end routers, it equally impacts all measurement packets and flows. Thus trouble with ensuring identical test set up by configuring two separated impairment generators identically is avoided (which was another proposal allowing remote metric compliance testing).

Appendix D. Glossary

ADK	Anderson-Darling K-Sample test, a test used to check whether two samples have the same statistical distribution.
ECMP	Equal Cost Multipath, a load balancing mechanism evaluating MPLS labels stacks, IP addresses and ports.
EDF	The "Empirical Distribution Function" of a set of scalar measurements is a function F(x) which for any x gives the fractional proportion of the total measurements that were smaller than or equal as x.
Metric	A measured quantity related to the performance and reliability of the Internet, expressed by a value. This could be a singleton (single value), a sample of single values or a statistic based on a sample of singletons.
OWAMP	One-way Active Measurement Protocol, a protocol for communication between IPPM measurement systems specified by IPPM.
OWD	One-Way Delay, a performance metric specified by IPPM.
Sample metric	A sample metric is derived from a given singleton metric by evaluating a number of distinct instances together.
Singleton metric	A singleton metric is, in a sense, one atomic measurement of this metric.
Statistical metric	A ’statistical’ metric is derived from a given sample metric by computing some statistic of the values defined by the singleton metric on the sample.
TWAMP	Two-way Active Measurement Protocol, a protocol for communication between IPPM measurement systems specified by IPPM.

Table 2

Authors' Addresses

Ruediger Geib (editor)
Deutsche Telekom
Heinrich Hertz Str. 3-7
Darmstadt, 64295
Germany

Phone: +49 6151 628 2747
Email: Ruediger.Geib@telekom.de

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/

Reza Fardid
Cariden Technologies
888 Villa Street, Suite 500
Mountain View, CA 94041
USA

Phone:
Email: rfardid@cariden.com

Alexander Steinmitz
HS Fulda
Marquardstr. 35
Fulda, 36039
Germany

Phone:
Email: steinionline@gmx.de

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2011

A. Morton
AT&T Labs

February 14, 2011

Round-trip Loss Metrics
draft-ietf-ippm-rt-loss-00

Abstract

Many user applications (and the transport protocols that make them possible) require two-way communications. To assess this capability, and to achieve test system simplicity, round-trip loss measurements are frequently conducted in practice. The Two-Way Active Measurement Protocol specified in RFC 5357 establishes a round-trip loss measurement capability for the Internet. However, there is currently no metric specified according to the RFC 2330 framework.

This memo adds round-trip loss to the set of IP Performance Metrics (IPPM).

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 August 18, 2011.

Copyright Notice

Table of Contents

1. Introduction

This memo defines a metric for round-trip loss on Internet paths. It builds on the notions and conventions introduced in the IP Performance Metrics (IPPM) framework [RFC2330]. Also, the specifications of the One-way Loss metric [RFC2680] and the Round-trip Delay metric [RFC2681] are frequently referenced and modified to match the round-trip circumstances addressed here. However, this memo assumes that the reader is familiar with the references, and does not repeat material as was done in [RFC2681].

This memo uses the terms "two-way" and "round-trip" synonymously.

1.1. Motivation

Many user applications and the transport protocols that make them possible require two-way communications. For example, the TCP SYN->, <-SYN-ACK, ACK-> three-way handshake attempted billions of times each day cannot be completed without two-way connectivity in a near-simultaneous time interval. Thus, measurements of Internet round-trip loss performance provide a basis to infer application performance more easily.

Measurement system designers have also recognized advantages of system simplicity when one host simply echoes or reflects test packets to the sender. Round-trip loss measurements are frequently conducted and reported in practice. The Two-Way Active Measurement Protocol specified in [RFC5357] establishes a round-trip loss measurement capability for the Internet. However, there is currently no round-trip loss metric specified according to the [RFC2330] framework.

[RFC2681] indicates that round-trip measurements may sometimes encounter "asymmetric" paths. When loss is observed using a round-trip measurement, there is often a desire to ascertain which of the two directional paths "lost" the packet. Under some circumstances, it is possible to make this inference. The round-trip measurement method raises a few complications when interpreting the embedded one-way results, and the user should be aware of them.

[RFC2681] also points out that loss measurement conducted sequentially in both directions of a path and reported as a round-trip result may be exactly the desired metric. On the other hand, it may be difficult to derive the state of round-trip loss from one-way measurements conducted in each direction unless a method to match the appropriate one-way measurements has pre-arranged.

Finally, many measurement systems report statistics on a conditional

delay distribution, where the condition is packet arrival at the destination. This condition is encouraged in [RFC3393], [RFC5481], and [draft-ietf-ippm-reporting-metrics]. As a result, lost packets need to be reported separately, according to a standardized metric. This memo defines such a metric.

See Section 1.1 of[RFC2680] for additional motivation of the packet loss metric.

2. Scope

This memo defines a round-trip loss metric using the conventions of the IPPM framework [RFC2330].

The memo defines a singleton metric, a sample metric, and a statistic, as per [RFC2330].

The memo also investigates the topic of one-way loss inference from a two-way measurement, and lists some key considerations.

3. Common Specifications for Round-trip Metrics

To reduce the redundant information presented in the detailed metrics sections that follow, this section presents the specifications that are common to two or more metrics. The section is organized using the same subsections as the individual metrics, to simplify comparisons.

3.1. Name: Type-P-*

All metrics use the Type-P convention as described in [RFC2330]. The rest of the name is unique to each metric.

3.2. Metric Parameters

- Src, the IP address of a host
- Dst, the IP address of a host
- T, a time (start of test interval)
- Tf, a time (end of test interval)
- lambda, a rate in reciprocal seconds (for Poisson Streams)

- incT, the nominal duration of inter-packet interval, first bit to first bit (for Periodic Streams)

- T0, a time that MUST be selected at random from the interval [T, T+dT] to start generating packets and taking measurements (for Periodic Streams)

- TstampSrc, the wire time of the packet as measured at MP(Src) as it leaves for Dst.

- TstampDst, the wire time of the packet as measured at MP(Dst), assigned to packets that arrive within a "reasonable" time.

- Tmax, a maximum waiting time for packets to arrive, set sufficiently long to disambiguate packets with long delays from packets that are discarded (lost).

- M, the total number of packets sent between T0 and Tf

- N, the total number of packets received at Dst (sent between T0 and Tf)

- Type-P, as defined in [RFC2330], which includes any field that may affect a packet's treatment as it traverses the network

3.3. Metric Definition

This section is specific to each metric.

3.4. Metric Units

The metric units are logical (1 or 0) when describing a single packet's loss performance, where a 0 indicates successful packet transmission and a 1 indicates packet loss.

Units of time are as specified in [RFC2330].

Other units used are defined in the associated section.

4. A Singleton Round-trip Loss Metric

4.1. Name: Type-P-Round-trip-Loss

4.2. Metric Parameters

See section 3.2.

4.3. Definition and Metric Units

Type-P-Round-trip-Loss SHALL be represented by the binary logical values (or their equivalents) when the following conditions are met:

Type-P-Round-trip-Loss = 0:

- Src sent the first bit of a Type-P packet to Dst at wire-time TstampSrc,
- that Dst received that packet,
- the Dst immediately sent a Type-P packet back to the Src, and
- that Src received the last bit of the reflected packet at wire-time TstampSrc + Tmax.

Type-P-Round-trip-Loss = 1:

- Src sent the first bit of a Type-P packet to Dst at wire-time TstampSrc,
- that Src did not receive the last bit of the reflected packet before the waiting time lapsed at TstampSrc + Tmax
- (possibly because that Dst did not receive that packet,
- the Dst did not immediately sent a Type-P packet back to the Src, or
- the Src did not receive a reflected Type-P packet sent from the Dst).

Following the precedent of[RFC2681], we make the simplifying assertion:

Type-P-Round-trip-Loss(Src->Dst) = Type-P-Round-trip-Loss(Dst->Src)

(and agree with the rationale presented, that the ambiguity introduced is a small price to pay for measurement efficiency).

Therefore, each singleton can be represented by pairs of elements as follows:

- o TstampSrc, the wire time of the packet at the Src (beginning the round-trip journey).

- o L, either zero or one (or some logical equivalent), where L=1 indicates loss and L=0 indicates successful round-trip arrival prior to TstampSrc + Tmax.

4.4. Discussion and other details

See [RFC2680] and [RFC2681] for extensive discussion, methods of measurement, errors and uncertainties, and other fundamental considerations that need not be repeated here.

5. A Sample Round-trip Loss Metric

Given the singleton metric Type-P-Round-trip-Loss, we now define one particular sample of such singletons. The idea of the sample is to select a particular binding of the parameters Src, Dst, and Type-P, then define a sample of values of parameter TstampSrc. This can be done in several ways, including:

1. Poisson: a pseudo-random Poisson process of rate lambda, whose values fall between T and Tf. The time interval between successive values of TstampSrc will then average 1/lambda, as per [RFC2330].

2. Periodic: a periodic stream process with pseudo-random start time T0 between T and dT, and nominal inter-packet interval incT, as per [RFC3432].

In the metric name, the variable <Stream> SHALL be replaced with the process used to define the sample, using one of the above processes (or other process, the details of which MUST be specified if used).

5.1. Name: Type-P-Round-trip-Loss-<Sample>-Stream

5.2. Metric Parameters

See section 3.2.

5.3. Definition and Metric Units

Given one of the methods for defining the test interval, the sample of times (TstampSrc) and other metric parameters, we obtain a sequence of Type-P-Round-trip-Loss singletons as defined in section 4.3.

Type-P-Round-trip-Loss-<Sample>-Stream SHALL be a sequence of pairs with elements as follows:

- TstampSrc, as above
- L, either zero or one (or some logical equivalent), where L=1 indicates loss and L=0 indicates successful round-trip arrival prior to TstampSrc + Tmax.

where <Sample> SHALL be replaced with "Poisson", "Periodic", or an appropriate term to designate another sample method meeting the criteria of [RFC2330].

5.4. Discussion and other details

See [RFC2680] and [RFC2681] for extensive discussion, methods of measurement, errors and uncertainties, and other fundamental considerations that need not be repeated here. However, when these references were approved, the packet reordering metrics in [RFC4737] had not yet been defined, nor had reordering been addressed in IPPM methodologies.

[RFC4737] defines packets that arrive "late" with respect to their sending order as reordered. For example, when packets arrive with sequence numbers 4, 7, 5, 6, then packets 5 and 6 are reordered, and they are obviously not lost because they have arrived within some reasonable waiting time threshold. The presence of reordering on a round-trip path has several likely affects on the measurement.

1. Methods of measurement should continue to wait the specified time for packets, and avoid prematurely declaring round-trip loss when a sequence gap or error is observed.

2. The time distribution of the singletons in the sample has been significantly changed.

3. Either the original packet stream or the reflected packet stream experienced path instability, and the original conditions may no longer be present.

Measurement implementations SHOULD address the possibility for packet reordering and avoid related errors in their processes.

6. Round-trip Loss Statistic

This section gives the primary and overall statistic for loss performance. Additional statistics and metrics originally prepared

for One-way loss MAY also be applicable.

6.1. Type-P-Round-trip-Loss-<Sample>-Ratio

Given a Type-P-Round-trip-Loss-<Sample>-Stream, the average of all the logical values, L, in the Stream is the Type-P-Round-trip-Loss-<Sample>-Ratio. This ratio is in units of lost packets per round-trip transmissions attempted.

In addition, the Type-P-Round-trip-Loss-<Sample>-Ratio is undefined if the sample is empty.

7. Round-trip Testing and One-way Reporting

This section raises considerations for results collected using a round-trip measurement architecture, such as in TWAMP [RFC5357].

The sampling process for the return path (Dst->Src) is a conditional process that depends on successful packet arrival at the Dst and correct operation at the Dst to generate the reflected packet. Therefore, the sampling process for the return path will be significantly affected when appreciable loss occurs on the Src->Dst path, making an attempt to assess the return path performance invalid (for loss or possibly any metric).

Further, the sampling times for the return path (Dst->Src) are a random process that depends on the original sample times (TstampSrc), the one-way-delay for successful packet arrival at the Dst, and time taken at the Dst to generate the reflected packet. Therefore, the sampling process for the return path will be significantly affected when appreciable delay variation occurs on the Src->Dst path, making an attempt to assess the return path performance invalid (for loss or possibly any metric).

As discussed above, packet reordering is always a possibility. In addition to the severe delay variation that usually accompanies it, reordering on the Src->Dst path will cause a mis-alignment of sequence numbers applied at the reflector when compared to the sender numbers. Measurement implementations SHOULD address this possible outcome.

8. Security Considerations

8.1. Denial of Service Attacks

This metric requires a stream of packets sent from one host (source) to another host (destination) through intervening networks, and back. This method could be abused for denial of service attacks directed at the destination and/or the intervening network(s).

Administrators of source, destination, and the intervening network(s) should establish bilateral or multi-lateral agreements regarding the timing, size, and frequency of collection of sample metrics. Use of this method in excess of the terms agreed between the participants may be cause for immediate rejection or discard of packets or other escalation procedures defined between the affected parties.

8.2. User Data Confidentiality

Active use of this method generates packets for a sample, rather than taking samples based on user data, and does not threaten user data confidentiality. Passive measurement must restrict attention to the headers of interest. Since user payloads may be temporarily stored for length analysis, suitable precautions MUST be taken to keep this information safe and confidential. In most cases, a hashing function will produce a value suitable for payload comparisons.

8.3. Interference with the metrics

It may be possible to identify that a certain packet or stream of packets is part of a sample. With that knowledge at the destination and/or the intervening networks, it is possible to change the processing of the packets (e.g. increasing or decreasing delay) that may distort the measured performance. It may also be possible to generate additional packets that appear to be part of the sample metric. These additional packets are likely to perturb the results of the sample measurement.

To discourage the kind of interference mentioned above, packet interference checks, such as cryptographic hash, may be used.

9. IANA Considerations

Metrics defined in IETF are typically registered in the IANA IPPM METRICS REGISTRY as described in initial version of the registry [RFC4148]. However, areas for improvement of this registry have been identified, and the registry structure has to be revisited when there is consensus to do so.

Therefore, the metrics in this draft may be considered for

registration in the future, and no IANA Action is requested at this time.

10. Acknowledgements

11. References

11.1. Normative References

[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.

[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.

[RFC4148] Stephan, E., "IP Performance Metrics (IPPM) Metrics Registry", BCP 108, RFC 4148, August 2005.

[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.

[RFC5835] Morton, A. and S. Van den Berghe, "Framework for Metric Composition", RFC 5835, April 2010.

11.2. Informative References

[RFC5474] Duffield, N., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, March 2009.

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

[Stats] McGraw-Hill NY NY, "Introduction to the Theory of Statistics, 3rd Edition,", 1974.

Author's Address

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

Phone: +1 732 420 1571
Fax: +1 732 368 1192
Email: acmorton@att.com
URI: http://home.comcast.net/~acmacm/

Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 7, 2011

L. Ciavattone
AT&T Labs
R. Geib
Deutsche Telekom
A. Morton
AT&T Labs
M. Wieser
University of Applied Sciences
Darmstadt
March 6, 2011

Test Plan and Results for Advancing RFC 2679 on the Standards Track
draft-morton-ippm-testplan-rfc2679-00

Abstract

This memo proposes to advance a performance metric RFC along the standards track, specifically RFC 2679 on One-way Delay Metrics. Observing that the metric definitions themselves should be the primary focus rather than the implementations of metrics, this memo describes the test procedures to evaluate specific metric requirement clauses to determine if the requirement has been interpreted and implemented as intended. Two completely independent implementations have been tested against the key specifications of RFC 2679.

Requirements Language

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

Status of this Memo

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

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

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

This Internet-Draft will expire on September 7, 2011.

Table of Contents

1. Introduction

The IETF (IP Performance Metrics working group, IPPM) has considered how to advance their metrics along the standards track since 2001, with the initial publication of Bradner/Paxson/Mankin's memo [ref to work in progress, draft-bradner-metricstest-]. The original proposal was to compare the results of implementations of the metrics, because the usual procedures for advancing protocols did not appear to apply. It was found to be difficult to achieve consensus on exactly how to compare implementations, since there were many legitimate sources of variation that would emerge in the results despite the best attempts to keep the network paths equal, and because considerable variation was allowed in the parameters (and therefore implementation) of each metric. Flexibility in metric definitions, essential for customization and broad appeal, made the comparison task quite difficult.

A renewed work effort sought to investigate ways in which the measurement variability could be reduced and thereby simplify the problem of comparison for equivalence.

There is *preliminary* consensus [I-D.ietf-ippm-metrictest] that the metric definitions should be the primary focus of evaluation rather than the implementations of metrics, and equivalent results are deemed to be evidence that the metric specifications are clear and unambiguous. This is the metric specification equivalent of protocol interoperability. The advancement process either produces confidence that the metric definitions and supporting material are clearly worded and unambiguous, OR, identifies ways in which the metric definitions should be revised to achieve clarity.

The process should also permit identification of options that were not implemented, so that they can be removed from the advancing specification (this is an aspect more typical of protocol advancement along the standards track).

This memo's purpose is to implement the current approach for [RFC2679]. It was prepared to help progress discussions on the topic of metric advancement, both through e-mail and at the upcoming IPPM meeting at IETF.

In particular, consensus is sought on the extent of tolerable errors when assessing equivalence in the results. In discussions, the IPPM working group agreed that test plan and procedures should include the threshold for determining equivalence, and this information should be available in advance of cross-implementation comparisons. This memo includes procedures for same-implementation comparisons to help set the equivalence threshold.

Another aspect of the metric RFC advancement process is the requirement to document the work and results. The procedures of [RFC2026] are expanded in[RFC5657], including sample implementation and interoperability reports. This memo follows the template in [I-D.morton-ippm-advance-metrics] for the report that accompanies the protocol action request submitted to the Area Director, including description of the test set-up, procedures, results for each implementation and conclusions.

1.1. RFC 2679 Coverage

This plan, in it's first draft version, does not cover all critical requirements and sections of [RFC2679]. Material will be added as it is "discovered" (not all requirements use requirements language).

2. A Definition-centric metric advancement process

The process described in Section 3.5 of [I-D.ietf-ippm-metrictest] takes as a first principle that the metric definitions, embodied in the text of the RFCs, are the objects that require evaluation and possible revision in order to advance to the next step on the standards track.

IF two implementations do not measure an equivalent singleton or sample, or produce the an equivalent statistic,

AND sources of measurement error do not adequately explain the lack of agreement,

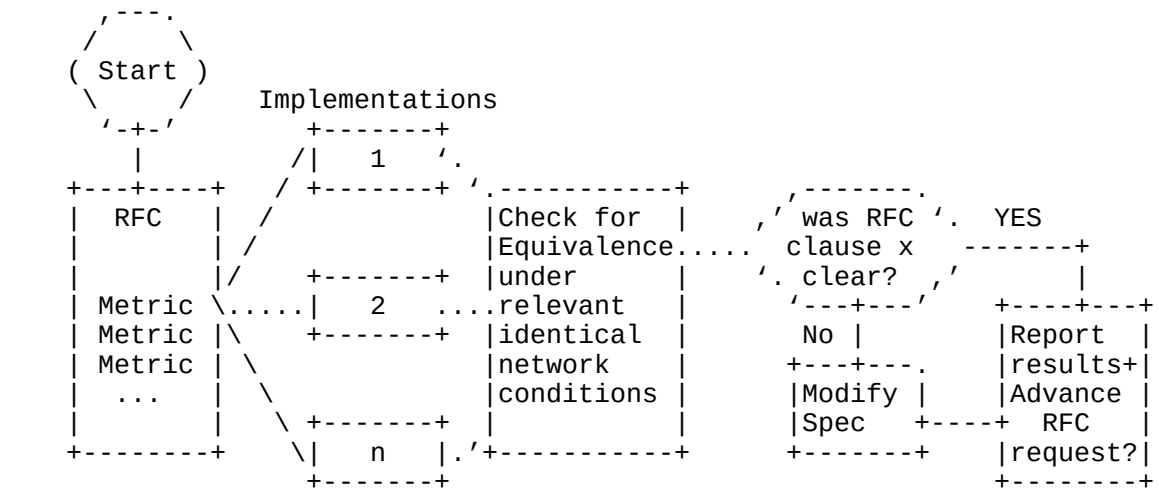
THEN the details of each implementation should be audited along with the exact definition text, to determine if there is a lack of clarity that has caused the implementations to vary in a way that affects the correspondence of the results.

IF there was a lack of clarity or multiple legitimate interpretations of the definition text,

THEN the text should be modified and the resulting memo proposed for consensus and advancement along the standards track.

Finally, all the findings MUST be documented in a report that can support advancement on the standards track, similar to those described in [RFC5657]. The list of measurement devices used in testing satisfies the implementation requirement, while the test results provide information on the quality of each specification in the metric RFC (the surrogate for feature interoperability).

The figure below illustrates this process:



3. Test configuration

>>>> This section needs to be updated <<<<

One metric implementation used was NetProbe version 5.8.5, (an earlier version is used in the WIPM system and deployed world-wide). NetProbe uses UDP packets of variable size, and can produce test streams with periodic or Poisson sample distributions.

>>> Add DT's Perfas Description

Figure 2 shows a view of the test path as each Implementation's test flows pass through the Internet and the L2TPv3 tunnel IDs (1 and 2), based on Figure 1 of [I-D.ietf-ippm-metrictest].

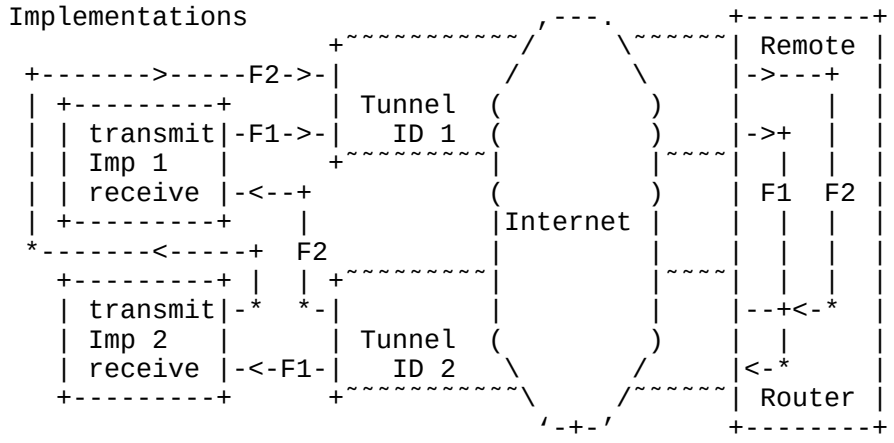


Illustration of a test setup with a bi-directional tunnel. For simplicity, only two measurement implementations and two flows (F#) between them are shown.

Figure 1

The testing employs the Layer 2 Tunnel Protocol, version 3 (L2TPv3) [RFC3931] tunnel between test sites on the Internet. The tunnel IP and L2TPv3 headers are intended to conceal the test equipment addresses and ports from hash functions that would tend to spread different test streams across parallel network resources, with likely variation in performance as a result.

At each end of the tunnel, VLANs encapsulated in the tunnel are looped-back so that test traffic is returned to each test site. Thus, test streams traverse the L2TP tunnel twice, but appear to be one-way tests from the test equipment point of view.

The network emulator is a host running Fedora Core Linux [http://fedoraproject.org/] with IP forwarding enabled and the NIST Net emulator 2.0.12b [http://snad.ncsl.nist.gov/nistnet/] loaded and operating.

The links between NetProbe hosts and the NIST Net emulator host were 100baseTx-FD (100Mbps full duplex) as reported by "mii-tool", except as noted below.

>>>> We need to decide on common packet rates, Poisson/Periodic, packet sizes, etc.

For these tests, a stream of at least 30 packets were sent from Source to Destination in each implementation. Periodic streams (as

per [RFC3432]) with 1 second spacing were used, except as noted.

Thus, the metric name for the testing configured here, with respect to the IP header exposed to Internet processing, is:

Type-IP-protocol-115-One-way-Delay-<StreamType>-Stream

With (Section 4.2. [RFC2679]) Metric Parameters: + Src, the IP address of a host + Dst, the IP address of a host + T0, a time + Tf, a time + lambda, a rate in reciprocal seconds

+ Thresh, a maximum waiting time in seconds (see Section 3.82 of [RFC2679]) And (Section 4.3. [RFC2679])Metric Units: A sequence of pairs; the elements of each pair are: + T, a time, and + dT, either a real number or an undefined number of seconds. The values of T in the sequence are monotonic increasing. Note that T would be a valid parameter to Type-P-One-way-Delay, and that dT would be a valid value of Type-P-One-way-Delay.

Also, Section 3.8.4 of [RFC2679] recommends that the path SHOULD be reported. In this test set-up, most of the path details will be concealed from the implementations by the L2TPv3 tunnels, thus a more informative path trace route can be conducted by the routers at each location.

When NetProbe is used in production, a trace route is conducted in parallel at the outset of measurements.

In Perfas, ???

4. Error Calibration, RFC 2679

An implementation is required to report on its error calibration in Section 3.8 of [RFC2679] (also required in Section 4.8 for sample metrics). Sections 3.6, 3.7, and 3.8 of [RFC2679] give the detailed formulation of the errors and uncertainties for calibration. In summary, Section 3.7.1 of [RFC2679] describes the total time-varying uncertainty as:

Esynch(t)+ Rsource + Rdest

where:

Esynch(t) denotes an upper bound on the magnitude of clock synchronization uncertainty.

Rsource and Rdest denote the resolution of the source clock and the

destination clock, respectively.

Further, Section 3.7.2 of [RFC2679] describes the total wire-time uncertainty as

Hsource + Hdest

referring to the upper bounds on host-time to wire-time for source and destination, respectively.

Section 3.7.3 of [RFC2679] describes a test with small packets over an isolated minimal network where the results can be used to estimate systematic and random components of the sum of the above errors or uncertainties. In a test with hundreds of singletons, the median is the systematic error and when the median is subtracted from all singletons, the remaining variability is the random error.

>>>>> The RFC text indicates that the clock-related errors are not included in this analysis, but a sufficiently long test (under full test load) should include all forms of error, IAO (in Al's opinion).

The test context, or Type-P of the test packets, must also be reported, as required in Section 3.8 of [RFC2679] and all metrics defined there. Type-P is defined in Section 13 of [RFC2330] (as are many terms used below).

4.1. NetProbe Error and Type-P

Type-P for this test was IP-UDP with Best Effort DCSP. These headers were encapsulated according to the L2TPv3 specifications [RFC3931], and thus may not influence the treatment received as the packets traversed the Internet.

In general, NetProbe error is dependent on the specific version and installation details.

NetProbe operates using host time above the UDP layer, which is different from the wire-time preferred in [RFC2330], but can be identified as a source of error according to Section 3.7.2 of [RFC2679].

Accuracy of NetProbe measurements is usually limited by NTP synchronization performance (which is typically taken as ~+/-1ms error or greater), although the installation used in this testing often exhibits errors much less than typical for NTP. The primary stratum 1 NTP server is closely located on a sparsely utilized network management LAN, thus it avoids many concerns raised in Section 10 of[RFC2330] (in fact, smooth adjustment, long-term drift

analysis and compensation, and infrequent adjustment all lead to stability during measurement intervals, the main concern).

The resolution of the reported results is 1us (us = microsecond) in the version of NetProbe tested here, which contributes to at least +/-1us error.

NetProbe implements a time-keeping sanity check on sending and receiving time-stamping processes. When the significant process interruption takes place, individual test packets are flagged as possibly containing unusual time errors, and are excluded from the sample used for all "time" metrics.

We performed a NetProbe calibration of the type described in Section 3.7.3 of [RFC2679], using 64 Byte packets over a cross-connect cable. The results estimate systematic and random components of the sum of the Hsource + Hdest errors or uncertainties. In a test with 300 singletons conducted over 30 seconds (periodic sample with 100ms spacing), the median is the systematic error and the remaining variability is the random error. One set of results is tabulated below:

(Results from the "R" software environment for statistical computing and graphics - http://www.r-project.org/)

```
> summary(XD4CAL)
      CAL1            CAL2            CAL3
 Min.   : 89.0   Min.   : 68.00   Min.   : 54.00
 1st Qu.: 99.0   1st Qu.: 77.00   1st Qu.: 63.00
 Median :110.0   Median : 79.00   Median : 65.00
 Mean   :116.8   Mean   : 83.74   Mean   : 69.65
 3rd Qu.:127.0   3rd Qu.: 88.00   3rd Qu.: 74.00
 Max.   :205.0   Max.   :177.00   Max.   :163.00
> boxplot(XD4CAL$CAL1,XD4CAL$CAL2,XD4CAL$CAL3)
```

NetProbe Calibration with Cross-Connect Cable, one-way delay values in microseconds (us)

The median or systematic error can be as high as 110 us, and the range of the random error is also on the order of 110 us for all streams.

Also, anticipating the Anderson-Darling K-sample (ADK) comparisons to follow, we corrected the CAL2 values for the difference between means between CAL2 and CAL3 (as specified in [I-D.ietf-ippm-metrictest]), and found strong support for the (Null Hypothesis that) the samples are from the same distribution (resolution of 1 us and alpha equal 0.05 and 0.01)

```
> XD4CVCAL2 <- XD4CAL$CAL2 - (mean(XD4CAL$CAL2)-mean(XD4CAL$CAL3))
> boxplot(XD4CVCAL2,XD4CAL$CAL3)
> XD4CV2_ADK <- adk.test(XD4CVCAL2, XD4CAL$CAL3)
> XD4CV2_ADK
Anderson-Darling k-sample test.

Number of samples:  2
Sample sizes: 300 300
Total number of values: 600
Number of unique values: 97

Mean of Anderson Darling Criterion: 1
Standard deviation of Anderson Darling Criterion: 0.75896

T = (Anderson Darling Criterion - mean)/sigma

Null Hypothesis: All samples come from a common population.

                      t.obs P-value extrapolation
not adj. for ties  0.71734 0.17042             0
adj. for ties     -0.39553 0.44589             1
>
```

4.2. Perfas Error and Type-P

5. Pre-determined Limits on Equivalence

>>>> This section contains many proposals <<<<<

In this section, we provide the numerical limits on comparisons between implementations, in order to declare that the results are equivalent and therefore, the tested specification is clear.

A key point is that the allowable errors, corrections, and confidence levels only need to be sufficient to detect mis-interpretation of the tested specification resulting in diverging implementations.

Also, the allowable error must be sufficient to compensate for measured path differences. It was simply not possible to measure fully identical paths in the VLAN-loopback test configuration used, and this practical compromise must be taken into account.

For Anderson-Darling K-sample (ADK) comparisons, the required confidence factor for the cross-implementation comparisons SHALL be the smallest of:

- o 0.95 confidence factor at 1ms resolution, or
- o the smallest confidence factor (in combination with resolution) of the two same-implementation comparisons for the same test conditions.

A constant time accuracy error of as much as +/-0.5ms MAY be removed from one implementation's distributions (all singletons) before the ADK comparison is conducted.

A constant propagation delay error (due to use of different sub-nets between the switch and measurement devices at each location) of as much as +2ms MAY be removed from one implementation's distributions (all singletons) before the ADK comparison is conducted.

For comparisons involving the mean of a sample or other central statistics, the limits on both the time accuracy error and the propagation delay error constants given above also apply.

6. Tests to evaluate RFC 2679 Specifications

This section describes some results from real-world (cross-Internet) tests with measurement devices implementing IPPM metrics and a network emulator to create relevant conditions, to determine whether the metric definitions were interpreted consistently by implementors.

The procedures are slightly modified from the original procedures contained in Appendix A.1 of [I-D.ietf-ippm-metrictest]. The modifications include the use of the mean statistic for comparisons.

Note that there are only five instances of the requirement term "MUST" in [RFC2679] outside of the boilerplate and [RFC2119] reference.

6.1. One-way Delay, ADK Sample Comparison - Same Implementation

This test determines if implementations produce results that appear to come from the same delay distribution, as an overall evaluation of Section 4 of [RFC2679], "A Definition for Samples of One-way Delay". Same-implementation comparison results help to set the threshold of equivalence that will be applied to cross-implementation comparisons.

This test is intended to evaluate measurements in sections 3 and 4 of [RFC2679].

By testing the extent to which the distributions of one-way delay singletons from two implementations of [RFC2679] appear to be from

the same distribution, we economize on comparisons, because comparing a set of individual summary statistics (as defined in Section 5 of [RFC2679]) would require another set of individual evaluations of equivalence. Instead, we can simply check which statistics were implemented, and report on those facts.

1. Configure an L2TPv3 path between test sites, and each pair of measurement devices to operate tests in their designated pair of VLANs.

2. Measure a sample of one-way delay singletons with 2 or more implementations, using identical options.

3. Measure a sample of one-way delay singletons with *five* additional instances of the *same* implementations, using identical options, noting that connectivity differences SHOULD be the same as for the cross implementation testing.

4. Apply the ADK comparison procedures (see Appendix C of [I-D.ietf-ippm-metrictest]) and determine the resolution and confidence factor for distribution equivalence of each same-implementation comparison and each cross-implementation comparison.

5. Take the coarsest resolution and confidence factor for distribution equivalence from the same-implementation pairs, or the limit defined in Section 5 above, as a limit on the equivalence threshold for these experimental conditions.

6. Apply constant correction factors to all singletons of the sample distributions, as described and limited in Section 5 above.

7. Compare the cross-implementation ADK performance with the equivalence threshold determined in step 5 to determine if equivalence can be declared.

6.1.1. NetProbe Same-implementation results

To be provided,

NetProbe ADK Results for same-implementation

6.1.2. Perfas Same-implementation results

To be provided,

Perfas ADK Results for same-implementation

6.1.3. One-way Delay, Cross-Implementation ADK Comparison

6.1.4. Conclusions on the ADK Results for One-way Delay

>>> Comment: this section is a placeholder

6.2. One-way Delay, Loss threshold, RFC 2679

This test determines if implementations use the same configured maximum waiting time delay from one measurement to another under different delay conditions, and correctly declare packets arriving in excess of the waiting time threshold as lost.

See Section 3.5 of [RFC2679], 3rd bullet point and also Section 3.8.2 of [RFC2679].

1. configure an L2TPv3 path between test sites, and each pair of measurement devices to operate tests in their designated pair of VLANs.

2. configure the network emulator to add 0.5 sec one-way constant delay to each direction of transmission (or 1 second one-way).

3. measure (average) one-way delay with 2 or more implementations, using identical waiting time thresholds (Thresh) for loss set at 2 seconds

4. configure the network emulator to add 1 sec one-way constant delay to each direction of transmission equivalent to 2 seconds of additional one-way delay, or change the path delay while test is in progress, when there are sufficient packets at the first delay setting)

5. repeat/continue measurements

6. observe that the increase measured in step 5 caused all packets with 2 sec additional delay to be declared lost, and that all packets that arrive successfully in step 3 are assigned a valid one-way delay.

6.2.1. NetProbe results for Loss Threshold

In NetProbe, the Loss Threshold is implemented uniformly over all packets as a post-processing routine. With the Loss Threshold set at 2 seconds, all packets with one-way delay >2 seconds are marked "Lost" and included in the Lost Packet list with their transmission

time (as required in Section 3.3 of [RFC2680]). 22 of 38 packets were declared lost.

6.2.2. Perfas Results for Loss Threshold

>>> Comment: this section is a placeholder

6.2.3. Conclusions on Lab Results for Loss Threshold

>>> Comment: this section is a placeholder

6.3. One-way Delay, First-bit to Last bit, RFC 2679

This test determines if implementations register the same relative increase in delay from one measurement to another under different delay conditions. This test tends to cancel the sources of error which may be present in an implementation.

See Section 3.7.2 of [RFC2679], and Section 10.2 of [RFC2330].

1. configure an L2TPv3 path between test sites, and each pair of measurement devices to operate tests in their designated pair of VLANs, and ideally including a low-speed link

2. measure (average) one-way delay with 2 or more implementations, using identical options and equal size small packets (e.g., 100 octet IP payload)

3. maintain the same path with X ms one-way delay

4. measure (average) one-way delay with 2 or more implementations, using identical options and equal size large packets (e.g., 1500 octet IP payload)

5. observe that the increase measured in steps 2 and 4 is equivalent to the increase in ms expected due to the larger serialization time for each implementation. Most of the measurement errors in each system should cancel, if they are stationary.

6.3.1. NetProbe Lab results for Serialization

For this test only, the link between the NetProbe Source host and the NIST Net emulator host was changed to 10baseT-FD (10Mbps full duplex) as configured by "mii-tool".

When the UDP payload size was increased from 32 octets to 1400 octets, the NIST Net emulator exhibited a bi-modal delay distribution. Investigation confirmed that the NetProbe

implementations tested did not exhibit bi-modal delay on an alternate (network management) path.

1400 byte payload Delay for each mode microseconds	32 byte payload (one mode) microseconds	Delay Diff microseconds	Expected Diff microseconds
1001621	1000356	1265	1094.4
1002735	1000356	2379	1094.4

Average Delay over 60 packets for different payload sizes with Delay computations and comparison with expected delay difference for serialization.

6.4. One-way Delay, Difference Sample Metric (Lab)

This test determines if implementations register the same relative increase in delay from one measurement to another under different delay conditions. This test tends to cancel the sources of error which may be present in an implementation.

This test is intended to evaluate measurements in sections 3 and 4 of [RFC2679].

1. configure an L2TPv3 path between test sites, and each pair of measurement devices to operate tests in their designated pair of VLANs.

2. measure (average) one-way delay with 2 or more implementations, using identical options

3. configure the path with X+Y ms one-way delay

4. repeat measurements

5. observe that the (average) increase measured in steps 2 and 4 is ~Y ms for each implementation. Most of the measurement errors in each system should cancel, if they are stationary.

6.4.1. NetProbe Lab results for Differential Delay

In this test, X=1000ms and Y=2000ms.

Average pre-increase delay, microseconds	1000276.6
Average post 2s additional, microseconds	3000282.6
Difference (should be ~= Y = 2s)	2000006

Average delays before/after 2 second increase

The NetProbe implementation exhibited a 2 second increase with a 6 microsecond error (assuming that the NIST Net emulated delay difference is exact).

6.5. Implementation of Statistics for One-way Delay

The ADK tests the extent to which the sample distributions of one-way delay singletons from two implementations of [RFC2679] appear to be from the same overall distribution. By testing this way, we economize on the number of comparisons, because comparing a set of individual summary statistics (as defined in Section 5 of [RFC2679]) would require another set of individual evaluations of equivalence. Instead, we can simply check which statistics were implemented, and report on those facts, noting that Section 5 of [RFC2679] does not specify the calculations exactly, and gives only some illustrative examples.

	NetProbe	Perfas
5.1. Type-P-One-way-Delay-Percentile	yes	
5.2. Type-P-One-way-Delay-Median	yes	
5.3. Type-P-One-way-Delay-Minimum	yes	
5.4. Type-P-One-way-Delay-Inverse-Percentile	no	

Implementation of Section 5 Statistics

5.1. Type-P-One-way-Delay-Percentile 5.2. Type-P-One-way-Delay-Median 5.3. Type-P-One-way-Delay-Minimum 5.4. Type-P-One-way-Delay-Inverse-Percentile

7. Security Considerations

The security considerations that apply to any active measurement of live networks are relevant here as well. See [RFC4656] and [RFC5357].

8. IANA Considerations

This memo makes no requests of IANA, and hopes that IANA will be as accepting of our new computer overlords as the authors intend to be.

9. Acknowledgements

The authors thank Lars Eggert for his continued encouragement to advance the IPPM metrics during his tenure as AD Advisor.

10. References

10.1. Normative References

[I-D.ietf-ippm-metrictest] Geib, R., Morton, A., Fardid, R., and A. Steinmitz, "IPPM standard advancement testing", draft-ietf-ippm-metrictest-01 (work in progress), October 2010.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[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.

[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.

[RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, November 2002.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.

[RFC4814] Newman, D. and T. Player, "Hash and Stuffing: Overlooked Factors in Network Device Benchmarking", RFC 4814,

March 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.

[RFC5657] Dusseault, L. and R. Sparks, "Guidance on Interoperation and Implementation Reports for Advancement to Draft Standard", BCP 9, RFC 5657, September 2009.

10.2. Informative References

[I-D.morton-ippm-advance-metrics] Morton, A., "Lab Test Results for Advancing Metrics on the Standards Track", draft-morton-ippm-advance-metrics-02 (work in progress), October 2010.

[RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

Authors' Addresses

Len Ciavattone
AT&T Labs
200 Laurel Avenue South
Middletown, NJ 07748
USA

Phone: +1 732 420 1239
Fax:
Email: lencia@att.com
URI:

Ruediger Geib
Deutsche Telekom
Heinrich Hertz Str. 3-7
Darmstadt, 64295
Germany

Phone: +49 6151 628 2747
Email: Ruediger.Geib@telekom.de

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/

Matthias Wieser
University of Applied Sciences Darmstadt
Birkenweg 8 Department EIT
Darmstadt, 64295
Germany

Phone:
Email: matthias.wieser@stud.h-da.de