

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 19, 2013

M. Bagnulo
UC3M
T. Burbridge
BT
S. Crawford
SamKnows
P. Eardley
BT
A. Morton
AT&T Labs
January 15, 2013

A registry for commonly used metrics
draft-bagnulo-ippm-new-registry-00

Abstract

This document creates a registry for commonly used metrics, defines the rules for assignments in the new registry and performs initial allocations. This document proposes one particular registry structure with a single registry with multiple sub-registries. A companion document draft-bagnulo-ippm-new-registry-independent explores an alternative structure with independent registries for each of the fields involved. .

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 19, 2013.

Copyright Notice

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

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

Table of Contents

1. Introduction	5
2. The commonly used metrics registry	6
2.1. The metrics registry	6
2.2. The Scheduling registry	7
2.3. The Environment registry	7
2.4. The Output type registry	7
3. Initial assignment for the Scheduling registry	8
3.1. Common parameter definitions	8
3.2. Poisson scheduling	8
3.3. Periodic scheduling	9
3.4. Singleton scheduling	9
4. Initial assignments for the Output Type registry	9
4.1. Raw	9
4.2. Xth percentile interval	10
4.3. Xth percentile mean	10
5. Initial assignments for the Environment registry	10
5.1. Undefined	10
5.2. No cross traffic	11
6. Initial assignments for the Metric registry	12
6.1. Comment	12
6.2. UDP related metrics	12
6.2.1. No cross traffic, raw, Poisson, UDP latency metric . .	13
6.2.2. No cross traffic, 99th percentile mean, Poisson, UDP latency metric	13
6.2.3. No cross traffic, 99th percentile interval, Poisson, UDP latency metric	14
6.2.4. No cross traffic, Poisson UDP packet-loss ratio metric	15
6.3. ICMP related metrics	15
6.3.1. No cross traffic, Periodic, ICMP packet-loss ratio metric	16
6.4. DNS related metrics	16
6.4.1. DNS latency metric	17
6.5. VoIP related metrics	18
6.5.1. No cross traffic, raw, Periodic, UDP latency metric .	18
6.5.2. No cross traffic, raw, Periodic, UDP loss metric . .	19
6.5.3. No cross traffic, raw, Periodic, UDP, PDV metric . .	20
6.5.4. No cross traffic, Periodic, UDP PDV:99.9 metric . .	21
6.5.5. No cross traffic, Periodic UDP packet-loss ratio metric	22
7. Security considerations	23
8. IANA Considerations	23
9. Acknowledgments	23
10. References	23
10.1. Normative References	23
10.2. Informative References	24

Authors' Addresses 24

1. Introduction

This document creates a registry for commonly used metrics. In order to do that, it creates a number of namespaces whose values will be recorded by the registry and will uniquely and precisely identify metrics.

The motivation for having such registry is to allow a controller to request a measurement agent to execute a measurement using a specific metric. Such request can be performed using any control protocol that refers to the value assigned to the specific metric in the registry. Similarly, the measurement agent can report the results of the measurement and by referring to the metric value it can unequivocally identify the metric that the results correspond to.

There was a previous attempt to define a metric registry RFC 4148 [RFC4148]. However, it was obsoleted by RFC 6248 [RFC6248] because it was "found to be insufficiently detailed to uniquely identify IPPM metrics... [there was too much] variability possible when characterizing a metric exactly" which led to the RFC4148 registry having "very few users, if any".

Our approach learns from this, by tightly defining each entry in the registry with only a few parameters open for each. The idea is that the entries in the registry represent different measurement tests, whilst the parameters set things like source and destination addresses that don't change the fundamental nature of the test. The downside of this approach is that it could result in an explosion in the number of entries in the registry. We believe that less is more in this context - it is better to have a reduced set of useful metrics rather than a large set of metrics with questionable usefulness. Therefore this document defines that the registry only includes commonly used metrics that are well defined; hence we require both specification required AND expert review policies for the assignment of values in the registry.

There are a couple of side benefits of having such registry. First the registry could serve as an inventory of useful and used metrics, that are normally supported by different implementations of measurement agents. Second, the results of the metrics would be comparable even if they are performed by different implementations and in different networks, as the metric is properly defined.

The registry forms part of a Measurement Plan {do you prefer the term 'Characterization Plan', 'control framework' or 'test schedule'?}. It describes various factors that need to be set by the party controlling the measurements, for example: specific values for the parameters associated with the selected registry entry (for instance,

source and destination addresses); and how often the measurement is made. The Measurement Plan might look something like: "Dear measurement agent: Please start test DNS(example.com) and RTT(server.com,150) every day at 2000 GMT. Run the DNS test 5 times and the RTT test 50 times. Do that when the network is idle. Generate both raw results and 99th percentile mean. Send measurement results to collector.com in IPFIX format". The Measurement Plan depends on the requirements of the controlling party. For instance the broadband consumer might want a one-off measurement made immediately to one specific server; a regulator might want the same measurement made once a day until further notice to the 'top 10' servers; whilst an operator might want a varying series of tests (some of which will be beyond those defined in the registry) as determined from time to time by their operational support system. While the registries defined in this document help to define the Measurement Plan its full specification falls outside the scope of this document.

2. The commonly used metrics registry

In this section we define the registry for commonly used metrics. It is composed by the following sub-registries:

- o Scheduling registry
- o Environment registry
- o Output-type registry
- o Metric registry

The rationale for the registry structure is to allow flexibility but yet precise definition of metrics. The metric registry is the fundamental registry and the other are auxiliary registries that define values for different fields in the metric registry.

2.1. The metrics registry

Each entry for the metrics registry contain the following information:

- o Identifier: A text string that uniquely identifies the metric
- o Name: The name of the metric
- o Environment: A value from the Environment registry
- o Output-type: A value from the Output-type registry
- o Scheduling: A value form the Scheduling registry
- o Reference: The specification where the metric is defined

The policy for the assignments in the metric registry is both specification required AND expert review. This means that in order to create an entry for the metric value a specification defining the metric is required and when that happens, the request for allocation

will be reviewed by an expert.

The specification must define the input parameters for the metric as well as the output of the metric. The metric must be well defined, in the sense that two independent implementations must produce uniform and comparable results.

The expert review must make sure that the proposed metric is operationally useful. This means that the metric has proven to be useful in operational/real scenarios.

2.2. The Scheduling registry

Each entry for the scheduling registry contain the following information:

- o Value: The name of the scheduling
- o Reference: the specification where the scheduling is defined

The scheduling defines the scheduling strategy for the metric. Simplest is Singleton scheduling, where an atomic measurement is made. Other strategies make a series of atomic measurements in a "sample" or "stream", with the schedule defining the timing between each distinct measurement. Each atomic measurement could consist of sending a single packet (such as a DNS request) or sending several packets (for example a webpage). A scheduling strategy requires input parameter(s). Assignment in this registry follows the specification required policy.

2.3. The Environment registry

Each entry for the environment registry contain the following information:

- o Value: The name of the environment
- o Reference: the specification where the environment is defined

The environment defines the conditions where the metric is expected to be used. It does not define the metric itself, but the context where the metric is executed. Assignment in this registry follows the specification required policy.

2.4. The Output type registry

Each entry for the output type registry contain the following information:

- o Value: The name of the output type
- o Reference: the specification where the output type is defined

The output type define the type of output that the metric produces.

It can be the raw results or it can be some form of statistic. Assignment in this registry follows the specification required policy. The specification of the output type must define the format of the output. Note that if two types of statistic are required from the same test (for example, both "Xth percentile mean" and "Raw") then a new output type must be defined ("Xth percentile mean AND Raw").

3. Initial assignment for the Scheduling registry

3.1. Common parameter definitions

Although each IPPM RFC defines individual parameters and uses them consistently, the parameter names are not completely consistent across the RFC set. For example, the variable "dT" is used in several different ways. This memo uses one set of parameter names, and the reader is cautioned to map the names according to their definitions.

We define some parameters that are used by several types of scheduling:

- o T0: time to begin a test
- o Tf: time to end a test

T0 and Tf are both in seconds and use the date (yyyy-mm-dd) and NTP 64 bit timestamp. T0 includes any control handshaking before the test stream or singleton. Tf is the time the last test data is sent.

As a result, we have:

- o Time when test devices may close the test socket: Tf + Waiting Time (the time to wait before declaring a packet lost is fixed for each metric)
- o Total duration of the test: Tf - T0 + Waiting Time

3.2. Poisson scheduling

The values for this entry are as follows:

- o Value: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

The Poisson scheduling is defined in section 11.1.1 of RFC 2330 [RFC2330] and needs input parameters:

- o T0 and Tf: defined above

- o `lambda`: the parameter defining the Poisson distribution. `Lambda` is the mean number of distinct measurements per second in the sample.

3.3. Periodic scheduling

The values for this entry are as follows:

- o Value: `Periodic`
- o Reference: `draft-bagnulo-ippm-new-registry`

The Periodic sampling is defined in RFC 3432 [RFC3432]. The additional input parameters for the metric required by Periodic scheduling are:

- o `T0` and `Tf`: defined above
 - * Note that with Periodic sampling, `T0` MUST NOT be strictly periodic with other tests of the same type. RFC 3432 [RFC3432] requires randomized start times and describes one way to accomplish this. Also, the duration of the test MUST be limited.
- o `incT`: the time in seconds between one distinct event and the next, where events typically result in repeating singleton measurements of various types (illustrated below).
 - * for a periodic stream this is the time between packets in the sample, first bit to first bit
 - * for measurements on a process this is the time between the first packets of the process, for example first bit to first bit of the SYN in a TCP 3-way handshake

3.4. Singleton scheduling

The values for this entry are as follows:

- o Value: `singleton`
- o Reference: `draft-bagnulo-ippm-new-registry`

The singleton scheduling covers the case when an atomic metric is performed as per RFC 2330 [RFC2330]. The additional input parameter for the metric required by Singleton scheduling is:

- o `T0`: defined above

4. Initial assignments for the Output Type registry

4.1. Raw

The values for this entry are as follows:

- o Value: Raw
- o Reference: draft-bagnulo-ippm-new-registry

The results of the metric are delivered in the exact way they are produced by the measurements without any further processing or filtering.

4.2. Xth percentile interval

The values for this entry are as follows:

- o Value: Xth percentile interval
- o Reference: draft-bagnulo-ippm-new-registry

The additional input parameter for the metric is:

- o X: the percentile (e.g, if the X input parameter is 99, then the output will be the 99th percentile interval.)

The output when using this Output type will be a couple of values, expressed in the same units as the raw output, that is the Xth percentile interval, as defined in section 1.3 of RFC 2330 [RFC2330].

4.3. Xth percentile mean

The values for this entry are as follows:

- o Value: Xth percentile mean
- o Reference: draft-bagnulo-ippm-new-registry

The additional input parameter for the metric is

- o X: the percentile (e.g, if the X input parameter is 99, then the output will be the 99th percentile mean.)

The output when using this Output type will be a single value, expressed in the same units as the raw output, that is the mean of the sample only considering the values contained in the Xth percentile interval, as defined in RFC 2330 [RFC2330].

5. Initial assignments for the Environment registry

5.1. Undefined

The values for this entry are as follows:

- o Value: Undefined
- o Reference: draft-bagnulo-ippm-new-registry

The undefined environment is the case where no additional environment settings are defined to perform the metric.

5.2. No cross traffic

The values for this entry are as follows:

- o Value: No cross-traffic
- o Reference: draft-bagnulo-ippm-new-registry

It is often important that there is no other traffic than the one generated by the measurement itself while doing the measurement. The reasons for this are two-folded, first, it is sometimes important that the traffic created by the measurement doesn't impact the experience of the users of the measured resource. Second it is sometimes important that no other traffic interferes with the measurement. This can be ensured by checking that the level of user traffic is either zero or low enough to be confident that it won't impact or be impacted by the measurement.

The "No cross traffic" condition is satisfied when, during the 5 seconds preceding measurement of the metric:

- o the level of traffic flowing through the interface that will be used to send measurement packets in either direction is less than a threshold value of 1% of the line rate of the aforementioned interface.

The "cross traffic" measurement is made at the interface, associated with the measurement agent, that user traffic flows across. For example, if the probe is attached to the home gateway, then the interface is the service demarcation point where the subscriber connects their private equipment or network to the subscribed service.

Note that the No-cross traffic condition is defined only for the link directly attached to the measurement agent initiating the measurement. There is nothing mentioned about cross traffic on other parts of the path used by measurement packets. In the case the bottleneck of the path is other link than the one directly attached to the device running the measurement agent, it may affect and be affected by the measurement even if the No cross traffic as defined here holds.

DISCUSSION

- o clarify whether traffic for each direction is less than threshold, or the sum
- o current SamKnows probes measure cross-traffic before the measurement of the metric. Another approach would be to measure cross-traffic during the time the metric is measured. Or a hybrid approach. These would either be separate environment entries, or parameterise the existing one.

- o current SamKnows probes define a fixed threshold. it could be a parameter
- o could ignore broadcast traffic (think SamKnows includes)
- o It would be possible to define this a bit more precisely as follows:
 - * The "No cross-traffic" condition is defined for active measurements. The measurement agent runs in a device that has one or more interfaces. In active measurements, the measurement agent sends one or more packets. Lets call if0 the interface through with the packets resulting from the measurement are sent through. The no cross traffic condition is fulfilled when during the 5 seconds prior sending each of the packets of the measurement:
 - + The traffic incoming through if0 that does not belong to the measurement is lower than 1% of the line rate of if0
 - + The traffic coming through the rest of the interfaces towards if0 is less than 1% of the line rate of if0.

6. Initial assignments for the Metric registry

6.1. Comment

Need to work through that we only define T0 and Tf (and not T, dT).

6.2. UDP related metrics

RFC 2681 [RFC2681] defines a Round-trip delay metric and RFC 6673 [RFC6673] defines a Round-trip packet loss metric. We build on these two metrics by specifying several of the open parameters to precisely define several metrics for measuring UDP latency and packet loss. All the UDP related metrics defined in this section use the following:

P-Type:

- o IPv4 header values:
 - * DSCP: set to 0
 - * TTL set to 255
 - * Protocol: Set to 17 (UDP)
- o UDP header values:
 - * Checksum: the checksum must be calculated
- o Payload
 - * Sequence number: 8-byte integer
 - * Timestamp: 8 byte integer. Expressed as 64-bit NTP timestamp as per section 6 of RFC 5905 [RFC5905]
 - * No padding

Timeout: 3 seconds

6.2.1. No cross traffic, raw, Poisson, UDP latency metric

We define the No cross traffic, raw, Poisson, UDP latency metric as follows:

- o Identifier: TBD1
- o Name: No cross-traffic, raw, Poisson, UDP latency
- o Environment: No cross-traffic, access measurement
- o Output type: raw
- o Scheduling: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-Round-trip-Delay-Poisson-Stream in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Initial time T
- o Duration dT
- o Rate lambda

The output of this metric is a list of elements. Each element corresponds to one packet sent. Each element contains the timestamp of the sent packet and the time when the echo was received.

6.2.2. No cross traffic, 99th percentile mean, Poisson, UDP latency metric

The methodology for this metric is defined as Type-P-Round-trip-Delay-Poisson-Stream in RFC 2681 [RFC2681] using the P-Type and Timeout defined above. However, the output of the metric is not the raw output, but the 99th percentile mean statistic.

The input parameters for this metric are:

- o Identifier: TBD2
- o Name: No cross-traffic, 99th percentile mean, Poisson, UDP latency
- o Environment: No cross-traffic, access measurement
- o Output type: Xth percentile mean
- o Scheduling: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Initial time T
- o Duration dT
- o Rate lambda
- o Xth percentile: 99

The output of this metric is a single value that corresponds to the 99th percentile mean of the results.

6.2.3. No cross traffic, 99th percentile interval, Poisson, UDP latency metric

The methodology for this metric is defined as Type-P-Round-trip-Delay-Poisson-Stream in RFC 2681 [RFC2681] using the P-Type and Timeout defined above. However, the output of the metric is not the raw output, but the 99th percentile interval statistic.

The input parameters for this metric are:

- o Identifier: TBD3
- o Name: No cross-traffic, 99th percentile interval, Poisson, UDP latency
- o Environment: No cross-traffic, access measurement
- o Output type: Xth percentile interval
- o Scheduling: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Initial time T
- o Duration dT
- o Rate lambda
- o Xth percentile: 99

The output of this metric is a single value that corresponds to the 99th percentile interval of the results.

6.2.4. No cross traffic, Poisson UDP packet-loss ratio metric

We define the No cross traffic, Poisson, UDP packet-loss ratio metric as follows:

- o Identifier: TBD4
- o Name: No cross-traffic, Poisson, UDP packet-loss ratio
- o Environment: No cross-traffic, access measurement
- o Output type: Xth percentile mean
- o Scheduling: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-Round-trip-Loss-Poisson-Ratio in RFC 6673 [RFC6673] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Initial time T
- o Duration dT
- o Rate lambda
- o X percentile: 100

The output of this metric is a single value that corresponds to the ratio of loss packets divided by the total number of packets sent.

6.3. ICMP related metrics

RFC 6673 [RFC6673] defines a Round-trip packet loss metric. We build on that metrics by specifying several of the open parameters to precisely define a metric for measuring ICMP packet loss. The ICMP related metric defined in this document use the following:

P-Type:

- o IPv4 header values:
 - * DSCP: set to 0
 - * TTL set to 255
 - * Protocol: Set to 1 (ICMP)
- o ICMP header values:
 - * Type: 8 (Echo request)
 - * Code: 0

Observation: reply packets will contain an ICMP type of 0 Echo reply.

Timeout: 3 seconds

6.3.1. No cross traffic, Periodic, ICMP packet-loss ratio metric

We define the No cross traffic, Periodic, ICMP packet-loss ratio metric as follows:

- o Identifier: TBD6
- o Name: No cross-traffic, Periodic, ICMP packet-loss ratio
- o Environment: No cross-traffic, access measurement
- o Output type: Xth percentile mean
- o Scheduling: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-Round-trip-Loss-Periodic-Ratio in RFC 6673 [RFC6673] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Initial time T
- o End Time Tf
- o dt (the interval allowed for sample start times)
- o Inter-packet time: incT
- o X percentile: 100

The output of this metric is a single value that corresponds to the ratio of loss packets divided by the total number of packets sent.

6.4. DNS related metrics

RFC 2681 [RFC2681] defines a Round-trip delay metric. We build on that metric by specifying several of the open parameters to precisely define a metric for measuring DNS latency. The metric uses the following parameters:

P-Type:

- o IPv4 header values:
 - * DSCP: set to 0
 - * TTL set to 255
 - * Protocol: Set to 17 (UDP)
- o UDP header values:
 - * Source port: 53
 - * Destination port: 53
 - * Checksum: the checksum must be calculated
- o Payload: The payload contains a DNS message as defined in RFC 1035 [RFC1035] with the following values:
 - * The DNS header section contains:
 - + QR: set to 0 (Query)

- + OPCODE: set to 0 (standard query)
- + AA: not set
- + TC: not set
- + RD: set to one (recursion desired)
- + RA: not set
- + RCODE: not set
- + QDCOUNT: set to one (only one entry)
- + ANCOUNT: not set
- + NSCOUNT: not set
- + ARCOUNT: not set
- * The Question section contains:
 - + QNAME: the FQDN provided as input for the test
 - + QTYPE: the query type provided as input for the test
 - + QCLASS: set to IN
- * The other sections do not contain any Resource Records.

Observation: reply packets will contain a DNS response and may contain RRs.

Timeout: 3 seconds

6.4.1. DNS latency metric

We define the DNS latency metric as follows:

- o Identifier: TBD7
- o Name: DNS latency
- o Environment: Undefined
- o Output type: raw
- o Scheduling: Singleton
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-Round-trip-Delay in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address (the address of the DNS server to be tested)
- o QTYPE: A RR
- o FQDN: a valid FQDN that will be queried for.
- o Time T

The output of this metric is the timestamp when the packet was sent and the delay that it took to receive a response. Please note that any DNS response is valid, including no records in the answer. (Should we be more explicit about what is the output when there is no reply packet received?)

6.5. VoIP related metrics

[RFC2679] defines a one-way delay metric and [RFC2680] defines a one-way packet loss metric. IPPM has derived a general packet delay variation metric in [RFC3393], which we apply as recommended in section 4.2 of [RFC5481]. We build on these specifications by specifying several of the open parameters to precisely define several metrics for measuring Voice over IP (VoIP) delay, delay variation, and packet loss. All the VoIP related metrics defined in this section use the following:

Type-P:

- o IPv4 header values:
 - * DSCP: set to 0 (I think we move this to the sub-sections)
 - * TTL set to 255
 - * Protocol: Set to 17 (UDP)
- o UDP header values:
 - * Checksum: the checksum must be calculated
- o Payload - sufficient octets to emulate a VoIP audio payload, including the an RTP header if desired, the actual test protocol will populate the payload with a measurement header containing fields such as:
 - * Sequence number:
 - * Timestamp:
 - * Random bit pattern:

Waiting Time to declare a packet lost: 5 seconds

Periodic Stream Description:

- o Nominal inter-packet interval incT=20ms (first bit to first bit)

6.5.1. No cross traffic, raw, Periodic, UDP latency metric

We define the No cross traffic, raw, Periodic, UDP latency metric as follows:

- o Identifier: TBD641
- o Name: No cross-traffic, raw, Periodic, UDP latency
- o Environment: No cross-traffic, access measurement
- o Output type: raw
- o Scheduling: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-One-way-Delay-Periodic-Stream in section 4 of [RFC3432], including parameters from section 3 of [RFC3432] and using the Type-P and Waiting Time defined above in section 6.4.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Beginning of testing interval, T
- o Initial time T0 (including a random offset from the beginning of T)
- o Ending time Tf

Variable aspects of Type-P are:

- o DSCP value
- o UDP Payload length

The output of this metric is a list of elements. Each element corresponds to one packet sent. Each element contains the timestamp of the sent packet and the time when the packet was received at the destination (from which the one-way delay can be calculated). The methodology's sequence number MAY be included. For packets which do not arrive prior to the Waiting Time, the received timestamp for that packet SHOULD be indicated as "not available", or post-processing may be applied to enforce the constant Waiting Time to exclude long-delayed packets and lost packets from further analysis.

6.5.2. No cross traffic, raw, Periodic, UDP loss metric

We define the No cross traffic, raw, Periodic, UDP loss metric as follows:

- o Identifier: TBD642
- o Name: No cross-traffic, raw, Periodic, UDP latency
- o Environment: No cross-traffic, access measurement
- o Output type: raw
- o Scheduling: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is identical to Type-P-One-way-Delay-Periodic-Stream in section 4 of [RFC3432], including parameters from section 3 of [RFC3432] and using the Type-P and Waiting Time defined above in section 6.4.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Beginning of testing interval, T
- o Initial time T0 (including a random offset from the beginning of T)

- o Ending time T_f

Variable aspects of Type-P are:

- o DSCP value
- o UDP Payload length

The output of this metric is a list of elements. Each element corresponds to one packet sent. Each element contains the timestamp of the sent packet and the time when the packet was received at the destination (from which the one-way delay can be calculated). The methodology's sequence number MAY be included. For packets which do not arrive prior to the Waiting Time, the received timestamp for that packet SHOULD be indicated as "not available", or post-processing may be applied to enforce the constant Waiting Time to exclude long-delayed packets and lost packets from further analysis.

Note that the same raw output format MAY serve both loss and delay metrics.

6.5.3. No cross traffic, raw, Periodic, UDP, PDV metric

We define the No cross traffic, Periodic, UDP Packet Delay Variation metric as follows:

- o Identifier: TBD643
- o Name: No cross-traffic, Periodic, UDP PDV
- o Environment: No cross-traffic, access measurement
- o Output type: raw
- o Scheduling: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for the delay singletons from which this metric is derived take the first steps defined as Type-P-One-way-Delay-Periodic-Stream in section 4 of [RFC3432], including parameters from section 3 of [RFC3432] and using the Type-P and Waiting Time defined above in section 6.4. This collects the one-way delay singletons.

The next step in the methodology follows from sections 2 and 3 of [RFC3393] (which describes how to use a selection function to determine pairs of packets to derive PDV) and section 4.2 of [RFC5481], where the packet with the minimum delay is specified as a fixed member of the pair.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port

- o Destination UDP port
- o Beginning of testing interval, T
- o Initial time T0 (including a random offset from the beginning of T)
- o Ending time Tf

Variable aspects of Type-P are:

- o DSCP value
- o UDP Payload length

The output of this metric is a list of triples (3 elements). Each element corresponds to one packet in the sample. Each element contains the one way delay of the first packet in the pair, the one way delay of the second packet in the pair (having minimum delay), and the variation in transmission time calculated for packet with sequence number i as $PDV(i) = D(i) - D(\min)$. The methodology's sequence number MAY be included. For packets which do not arrive prior to the Waiting Time, the delay for that packet and its PDV SHOULD be indicated as "not available" (following section 4.1 of [RFC3393]).

6.5.4. No cross traffic, Periodic, UDP PDV:99.9 metric

We define the No cross traffic, Periodic, UDP Packet Delay Variation (99.9 percentile) metric as follows:

- o Identifier: TBD644
- o Name: No cross-traffic, Periodic, UDP PDV:99.9
- o Environment: No cross-traffic, access measurement
- o Output type: 99.9 percentile
- o Scheduling: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for the delay singletons from which this metric is derived take the first steps defined as Type-P-One-way-Delay-Periodic-Stream in section 4 of [RFC3432], including parameters from section 3 of [RFC3432] and using the Type-P and Waiting Time defined above in section 6.4. This collects the one-way delay singletons.

The next step in the methodology follows from sections 2 and 3 of [RFC3393] (which describes how to use a selection function to determine pairs of packets to derive PDV) and section 4.2 of [RFC5481], where the packet with the minimum delay is specified as a fixed member of the pair.

The input parameters for this metric are:

- o Source IP Address

- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Beginning of testing interval, T
- o Initial time T0 (including a random offset from the beginning of T)
- o Ending time Tf

Variable aspects of Type-P are:

- o DSCP value
- o UDP Payload length

The output of this metric is a single value corresponding to the 99.9th percentile of PDV. For packets which do not arrive prior to the Waiting Time, the delay for that packet and its PDV SHOULD be indicated as "not available" (following section 4.1 of [RFC3393]). If the 99.9th percentile of singletons corresponds to packet whose delay and PDV are "not available", then the output of this metric is "not available".

6.5.5. No cross traffic, Periodic UDP packet-loss ratio metric

We define the No cross traffic, Periodic, UDP packet-loss ratio metric as follows:

- o Identifier: TBD645
- o Name: No cross-traffic, Periodic, UDP packet-loss ratio
- o Environment: No cross-traffic, access measurement
- o Output type: X percentile mean
- o Scheduling: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-One-way-Loss-Average in Section 4.1 of [RFC2680] using the Type-P and Waiting Time defined in section 6.4 above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Beginning of testing interval, T
- o Initial time T0 (including a random offset from the beginning of T)
- o Ending time Tf
- o X percentile mean: 100

Variable aspects of Type-P are:

- o DSCP value
- o UDP Payload length

The output of this metric is one value that corresponds to the ratio of lost packets divided by the total number of packets sent. This can be calculated from the singleton elements of section 6.4.2 above, assigning the logical value "0" to packets with a valid one-way delay and the value "1" to all packets whose one-way delay is recorded as "not available". As section 4.1 of [RFC2680] indicates, the average of all the logical values is the ratio of lost to total packets.

7. Security considerations

TBD

8. IANA Considerations

TBD

9. Acknowledgments

We would like to thank Henning Schulzrinne for many constructive comments and input on early versions of this document.

10. References

10.1. Normative References

- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, November 2002.
- [RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.
- [RFC6673] Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, August 2012.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

- [RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.
- [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.
- [RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, November 2002.
- [RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, March 2009.

10.2. Informative References

- [RFC4148] Stephan, E., "IP Performance Metrics (IPPM) Metrics Registry", BCP 108, RFC 4148, August 2005.
- [RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, April 2011.

Authors' Addresses

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6249500
Email: marcelo@it.uc3m.es
URI: <http://www.it.uc3m.es>

Trevor Burbridge
British Telecom
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: trevor.burbridge@bt.com

Sam Crawford
SamKnows

Email: sam@samknows.com

Philip Eardley
British Telecom
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: philip.eardley@bt.com

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

Email: acmorton@att.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 19, 2013

M. Bagnulo
UC3M
T. Burbridge
BT
S. Crawford
SamKnows
P. Eardley
BT
A. Morton
AT&T Labs
January 15, 2013

A registry for commonly used metrics. Independent registries
draft-bagnulo-ippm-new-registry-independent-00

Abstract

This document creates a registry for commonly used metrics, defines the rules for assignments in the new registry and performs initial allocations. This document proposes one particular registry structure with independent registries for each of the fields involved. A companion document draft-bagnulo-ippm-new-registry explores an alternative structure with a single registry with multiple sub-registries.

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 19, 2013.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
2. The commonly used metrics registry	4
2.1. The metrics registry	4
2.2. The Scheduling registry	5
2.3. The Environment registry	5
2.4. The Output type registry	5
3. Initial assignment for the Scheduling registry	6
3.1. Common parameter definitions	6
3.2. Poisson scheduling	6
3.3. Periodic scheduling	7
3.4. Singleton scheduling	7
4. Initial assignments for the Output Type registry	7
4.1. Raw	7
4.2. Xth percentile interval	8
4.3. Xth percentile mean	8
5. Initial assignments for the Environment registry	8
5.1. Undefined	8
5.2. No cross traffic	9
6. Initial assignments for the Metric registry	10
6.1. Comment	10
6.2. UDP related metrics	10
6.2.1. UDP latency metric	11
6.2.2. UDP packet-loss metric	11
7. ICMP related metrics	12
7.1. ICMP packet-loss metric	12
8. DNS related metrics	12
8.1. DNS latency metric	13
9. Some examples of measurement plans	14
10. Security considerations	15
11. IANA Considerations	15
12. Acknowledgments	15
13. References	15
13.1. Normative References	15
13.2. Informative References	16
Authors' Addresses	16

1. Introduction

This document creates a registry for commonly used metrics. In order to do that, it creates a number of namespaces whose values will be recorded by the registry and will uniquely and precisely identify metrics.

The motivation for having such registry is to allow a controller to request a measurement agent to execute a measurement using a specific metric. Such request can be performed using any control protocol that refers to the value assigned to the specific metric in the registry. Similarly, the measurement agent can report the results of the measurement and by referring to the metric value it can unequivocally identify the metric that the results correspond to.

There was a previous attempt to define a metric registry RFC 4148 [RFC4148]. However, it was obsoleted by RFC 6248 [RFC6248] because it was "found to be insufficiently detailed to uniquely identify IPPM metrics... [there was too much] variability possible when characterizing a metric exactly" which led to the RFC4148 registry having "very few users, if any".

Our approach learns from this, by tightly defining each entry in the registry with only a few parameters open for each. The idea is that the entries in the registry represent different measurement tests, whilst the parameters set things like source and destination addresses that don't change the fundamental nature of the test. The downside of this approach is that it could result in an explosion in the number of entries in the registry. We believe that less is more in this context - it is better to have a reduced set of useful metrics rather than a large set of metrics with questionable usefulness. Therefore this document defines that the registry only includes commonly used metrics that are well defined; hence we require both specification required AND expert review policies for the assignment of values in the registry.

There are a couple of side benefits of having such registry. First the registry could serve as an inventory of useful and used metrics, that are normally supported by different implementations of measurement agents. Second, the results of the metrics would be comparable even if they are performed by different implementations and in different networks, as the metric is properly defined.

This version of the document defines a set of independent registries, that limits the explosion of registry entries by allowing arbitrary combinations of entries in the different entries. The downside is that the list of useful metrics is less defined, as any combination would be defined. Which approach is better is up for discussion.

The registry forms part of a Measurement Plan {do you prefer the term 'Characterization Plan', 'control framework' or 'test schedule'?}. It describes various factors that need to be set by the party controlling the measurements, for example: specific values for the parameters associated with the selected registry entry (for instance, source and destination addresses); and how often the measurement is made. The Measurement Plan might look something like: "Dear measurement agent: Please start test DNS(example.com) and RTT(server.com,150) every day at 2000 GMT. Run the DNS test 5 times and the RTT test 50 times. Do that when the network is idle. Generate both raw results and 99th percentile mean. Send measurement results to collector.com in IPFIX format". The Measurement Plan depends on the requirements of the controlling party. For instance the broadband consumer might want a one-off measurement made immediately to one specific server; a regulator might want the same measurement made once a day until further notice to the 'top 10' servers; whilst an operator might want a varying series of tests (some of which will be beyond those defined in the registry) as determined from time to time by their operational support system. While the registries defined in this document help to define the Measurement Plan its full specification falls outside the scope of this document.

2. The commonly used metrics registry

In this section we define the registry for commonly used metrics. It is composed by the following sub-registries:

- o Scheduling registry
- o Environment registry
- o Output-type registry
- o Metric registry

The rationale for the registry structure is to allow flexibility but yet precise definition of metrics. The metric registry defines the metric itself while the other registries define additional aspects that are needed for the measurement plan and that are needed to fully specify a measurement request from a controller to a measurement agent.

2.1. The metrics registry

Each entry for the metrics registry contain the following information:

- o Value: A text string that uniquely identifies the metric
- o Reference: The specification where the metric is defined

The policy for the assignments in the metric registry is both

specification required AND expert review. This means that in order to create an entry for the metric value a specification defining the metric is required and when that happens, the request for allocation will be reviewed by an expert.

The specification must define the input parameters for the metric as well as the output of the metric. The metric must be well defined, in the sense that two independent implementations must produce uniform and comparable results.

The expert review must make sure that the proposed metric is operationally useful. This means that the metric has proven to be useful in operational/real scenarios.

2.2. The Scheduling registry

Each entry for the scheduling registry contain the following information:

- o Value: The name of the scheduling
- o Reference: the specification where the scheduling is defined

The scheduling defines the scheduling strategy for the metric. Simplest is Singleton scheduling, where an atomic measurement is made. Other strategies make a series of atomic measurements in a "sample" or "stream", with the schedule defining the timing between each distinct measurement. Each atomic measurement could consist of sending a single packet (such as a DNS request) or sending several packets (for example a webpage). A scheduling strategy requires input parameter(s). Assignment in this registry follows the specification required policy.

2.3. The Environment registry

Each entry for the environment registry contain the following information:

- o Value: The name of the environment
- o Reference: the specification where the environment is defined

The environment defines the conditions where the metric is expected to be used. It does not define the metric itself, but the context where the metric is executed. Assignment in this registry follows the specification required policy.

2.4. The Output type registry

Each entry for the output type registry contain the following information:

- o Value: The name of the output type
- o Reference: the specification where the output type is defined

The output type define the type of output that the metric produces. It can be the raw results or it can be some form of statistic. Assignment in this registry follows the specification required policy. The specification of the output type must define the format of the output.

3. Initial assignment for the Scheduling registry

3.1. Common parameter definitions

Although each IPPM RFC defines individual parameters and uses them consistently, the parameter names are not completely consistent across the RFC set. For example, the variable "dT" is used in several different ways. This memo uses one set of parameter names, and the reader is cautioned to map the names according to their definitions.

We define some parameters that are used by several types of scheduling:

- o T0: time to begin a test
 - o Tf: time to end a test
- T0 and Tf are both in seconds and use the date (yyyy-mm-dd) and NTP 64 bit timestamp. T0 includes any control handshaking before the test stream or singleton. Tf is the time the last test data is sent.

As a result, we have:

- o Time when test devices may close the test socket: Tf + Waiting Time (the time to wait before declaring a packet lost is fixed for each metric)
- o Total duration of the test: Tf - T0 + Waiting Time

3.2. Poisson scheduling

The values for this entry are as follows:

- o Value: Poisson
- o Reference: draft-bagnulo-ippm-new-registry

The Poisson scheduling is defined in section 11.1.1 of RFC 2330 [RFC2330] and needs input parameters:

- o T0 and Tf: defined above
- o lambda: the parameter defining the Poisson distribution. Lambda is the mean number of distinct measurements per second in the sample.

3.3. Periodic scheduling

The values for this entry are as follows:

- o Value: Periodic
- o Reference: draft-bagnulo-ippm-new-registry

The Periodic sampling is defined in RFC 3432 [RFC3432]. The additional input parameters for the metric required by Periodic scheduling are:

- o T0 and Tf: defined above
 - * Note that with Periodic sampling, T0 MUST NOT be strictly periodic with other tests of the same type. RFC 3432 [RFC3432] requires randomized start times and describes one way to accomplish this. Also, the duration of the test MUST be limited.
- o incT: the time in seconds between one distinct event and the next, where events typically result in repeating singleton measurements of various types (illustrated below).
 - * for a periodic stream this is the time between packets in the sample, first bit to first bit
 - * for measurements on a process this is the time between the first packets of the process, for example first bit to first bit of the SYN in a TCP 3-way handshake

3.4. Singleton scheduling

The values for this entry are as follows:

- o Value: singleton
- o Reference: draft-bagnulo-ippm-new-registry

The singleton scheduling covers the case when an atomic metric is performed as per RFC 2330 [RFC2330]. The additional input parameter for the metric required by Singleton scheduling is:

- o T0: defined above

4. Initial assignments for the Output Type registry

4.1. Raw

The values for this entry are as follows:

- o Value: Raw
- o Reference: draft-bagnulo-ippm-new-registry

The results of the metric are delivered in the exact way they are produced by the measurements without any further processing or filtering.

4.2. Xth percentile interval

The values for this entry are as follows:

- o Value: Xth-percentile
- o Reference: draft-bagnulo-ippm-new-registry

The additional input parameter for the metric is:

- o X: the percentile (e.g, if the X input parameter is 99, then the output will be the 99th percentile interval.)

The output when using this Output type will be a couple of values, expressed in the same units as the raw output, that is the Xth percentile interval, as defined in section 1.3 of RFC 2330 [RFC2330].

4.3. Xth percentile mean

The values for this entry are as follows:

- o Value: Xth-percentile-mean
- o Reference: draft-bagnulo-ippm-new-registry

The additional input parameter for the metric is

- o X: the percentile (e.g, if the X input parameter is 99, then the output will be the 99th percentile mean.)

The output when using this Output type will be a single value, expressed in the same units as the raw output, that is the mean of the sample only considering the values contained in the Xth percentile interval, as defined in RFC 2330 [RFC2330].

5. Initial assignments for the Environment registry

5.1. Undefined

The values for this entry are as follows:

- o Value: Undefined
- o Reference: draft-bagnulo-ippm-new-registry

The undefined environment is the case where no additional environment settings are defined to perform the metric.

5.2. No cross traffic

The values for this entry are as follows:

- o Value: No-cross-traffic
- o Reference: draft-bagnulo-ippm-new-registry

It is often important that there is no other traffic than the one generated by the measurement itself while doing the measurement. The reasons for this are two-folded, first, it is sometimes important that the traffic created by the measurement doesn't impact the experience of the users of the measured resource. Second it is sometimes important that no other traffic interferes with the measurement. This can be ensured by checking that the level of user traffic is either zero or low enough to be confident that it won't impact or be impacted by the measurement.

The "No cross traffic" condition is satisfied when, during the 5 seconds preceding measurement of the metric:

- o the level of traffic flowing through the interface that will be used to send measurement packets in either direction is less than a threshold value of 1% of the line rate of the aforementioned interface.

The "cross traffic" measurement is made at the interface, associated with the measurement agent, that user traffic flows across. For example, if the probe is attached to the home gateway, then the interface is the service demarcation point where the subscriber connects their private equipment or network to the subscribed service.

Note that the No-cross traffic condition is defined only for the link directly attached to the measurement agent initiating the measurement. There is nothing mentioned about cross traffic on other parts of the path used by measurement packets. In the case the bottleneck of the path is other link than the one directly attached to the device running the measurement agent, it may affect and be affected by the measurement even if the No cross traffic as defined here holds.

DISCUSSION

- o It is not clear we need a registry for this. If the only thing we are going to define is the No cross traffic condition, we can simply set it as an input parameter in each metric.
- o clarify whether traffic for each direction is less than threshold, or the sum

- o current SamKnows probes measure cross-traffic before the measurement of the metric. Another approach would be to measure cross-traffic during the time the metric is measured. Or a hybrid approach. These would either be separate environment entries, or parameterise the existing one.
- o current SamKnows probes define a fixed threshold. it could be a parameter
- o could ignore broadcast traffic (think SamKnows includes)
- o It would be possible to define this a bit more precisely as follows:
 - * The "No cross-traffic" condition is defined for active measurements. The measurement agent runs in a device that has one or more interfaces. In active measurements, the measurement agent sends one or more packets. Lets call if0 the interface through with the packets resulting from the measurement are sent through. The no cross traffic condition is fulfilled when during the 5 seconds prior sending each of the packets of the measurement:
 - + The traffic incoming through if0 that does not belong to the measurement is lower than 1% of the line rate of if0
 - + The traffic coming through the rest of the interfaces towards if0 is less than 1% of the line rate of if0.

6. Initial assignments for the Metric registry

6.1. Comment

Need to work through that we only define T0 and Tf (and not T, dT).

6.2. UDP related metrics

RFC 2681 [RFC2681] defines a Round-trip delay metric and RFC 6673 [RFC6673] defines a Round-trip packet loss metric. We build on these two metrics by specifying several of the open parameters to precisely define several metrics for measuring UDP latency and packet loss. All the UDP related metrics defined in this section use the following:

P-Type:

- o IPv4 header values:
 - * DSCP: set to 0
 - * TTL set to 255
 - * Protocol: Set to 17 (UDP)
- o UDP header values:
 - * Checksum: the checksum must be calculated

- o Payload
 - * Sequence number: 8-byte integer
 - * Timestamp: 8 byte integer. Expressed as 64-bit NTP timestamp as per section 6 of RFC 5905 [RFC5905]
 - * No padding

Timeout: 3 seconds

6.2.1. UDP latency metric

We define the UDP latency metric as follows:

- o Value: UDP_Latency
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-Round-trip-Delay- in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Time

The output of this metric is the couple of values formed by the timestamp of the sent packet and the time when the echo was received. They are expressed in seconds and use the date (yyyy-mm-dd) and NTP 64 bit timestamp

6.2.2. UDP packet-loss metric

We define the UDP packet-loss metric as follows:

- o Value: UDP_packet_loss
- o Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-Round-trip-Loss in RFC 6673 [RFC6673] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Source UDP port
- o Destination UDP port
- o Time T

The output of this metric is a single value 0 (packet was lost) or 1 (packet has arrived before timeout)

7. ICMP related metrics

RFC 6673 [RFC6673] defines a Round-trip packet loss metric. We build on that metrics by specifying several of the open parameters to precisely define a metric for measuring ICMP packet loss. The ICMP related metric defined in this document use the following:

P-Type:

- o IPv4 header values:
 - * DSCP: set to 0
 - * TTL set to 255
 - * Protocol: Set to 1 (ICMP)
- o ICMP header values:
 - * Type: 8 (Echo request)
 - * Code: 0

Observation: reply packets will contain an ICMP type of 0 Echo reply.

Timeout: 3 seconds

7.1. ICMP packet-loss metric

We define the ICMP packet-loss metric as follows:

- o Value: ICMP_Packet_Loss
- o Reference: draft-bagnulo-ippm-new-registry

This metric is defined as Type-P-Round-trip-Loss in RFC 6673 [RFC6673] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address
- o Time T

The output of this metric is a single value 0 (packet was lost) or 1 (packet has arrived before timeout)

8. DNS related metrics

RFC 2681 [RFC2681] defines a Round-trip delay metric. We build on that metric by specifying several of the open parameters to precisely define a metric for measuring DNS latency. The metric uses the following parameters:

P-Type:

- o IPv4 header values:
 - * DSCP: set to 0
 - * TTL set to 255
 - * Protocol: Set to 17 (UDP)
- o UDP header values:
 - * Source port: 53
 - * Destination port: 53
 - * Checksum: the checksum must be calculated
- o Payload: The payload contains a DNS message as defined in RFC 1035 [RFC1035] with the following values:
 - * The DNS header section contains:
 - + QR: set to 0 (Query)
 - + OPCODE: set to 0 (standard query)
 - + AA: not set
 - + TC: not set
 - + RD: set to one (recursion desired)
 - + RA: not set
 - + RCODE: not set
 - + QDCOUNT: set to one (only one entry)
 - + ANCOUNT: not set
 - + NSCOUNT: not set
 - + ARCOUNT: not set
 - * The Question section contains:
 - + QNAME: the FQDN provided as input for the test
 - + QTYPE: the query type provided as input for the test
 - + QCLASS: set to IN
 - * The other sections do not contain any Resource Records.

Observation: reply packets will contain a DNS response and may contain RRs.

Timeout: 3 seconds

8.1. DNS latency metric

We define the DNS latency metric as follows:

- o Value: DNS_Latency
- o Reference: draft-bagnulo-ippm-new-registry

The methodology for this metric is defined as Type-P-Round-trip-Delay in RFC 2681 [RFC2681] using the P-Type and Timeout defined above.

The input parameters for this metric are:

- o Source IP Address
- o Destination IP Address (the address of the DNS server to be tested)

- o QTYPE: A RR
- o FQDN: a valid FQDN that will be queried for.
- o Time T

The output of this metric is the timestamp when the packet was sent and the delay that it took to receive a response. Please note that any DNS response is valid, including no records in the answer. (Should we be more explicit about what is the output when there is no reply packet received?)

9. Some examples of measurement plans

A measurement plan will be characterized by the following tuple: (Metric, environment, scheduling, output format). We will next present some measurement plans that are currently used.

A measurement plan for measuring the 99th percentile interval of the UDP latency without cross traffics, using a Poisson stream with rate 1 pkts/sec, stating at time T0 and ending at Tf seconds, between source IP address IPs and source port Ps and destination IP address IPd and destination port Pd would be expressed as:

```
(UDP_Latency(IPs,Ps,IPd,Pd), No-cross-traffic, Poisson(T0,Tf,1),  
Xth-percentile(99))
```

A measurement plan for measuring the UDP packet loss ration without cross traffics, using a Poisson stream with rate 1 pkts/sec, stating at time T0 and ending at Tf seconds, between source IP address IPs and source port Ps and destination IP address IPd and destination port Pd would be expressed as:

```
(UDP_Packet_Loss(IPs,Ps,IPd,Pd), No-cross-traffic,  
Poisson(T0,Tf,1), Xth-percentile-mean(100))
```

A measurement plan for measuring the ICMP packet loss ratio, using a Periodic stream s second between packets, stating at time T0 and ending at Tf seconds, between source IP address IPs and destination IP address IPd would be expressed as:

```
(ICMP_Packet_Loss(IPs,IPd), Undefined, Periodic(T0,Tf,s), Xth-  
percentile-mean(100))
```

A measurement plan for measuring the DNS latency for resolving FQDN foo.com between a resolver in IP address IPs and a server with address IPd at time T would be expressed as:

```
(DNS_Latency(IPs,IPd,foo.com), Undefined, Singleton(T), raw)
```

10. Security considerations

TBD

11. IANA Considerations

TBD

12. Acknowledgments

We would like to thank Henning Schulzrinne for many constructive comments and input on early versions of this document.

13. References

13.1. Normative References

- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, November 2002.
- [RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.
- [RFC6673] Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, August 2012.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.
- [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.
- [RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation

Metric for IP Performance Metrics (IPPM)", RFC 3393, November 2002.

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

13.2. Informative References

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

[RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, April 2011.

Authors' Addresses

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6249500
Email: marcelo@it.uc3m.es
URI: <http://www.it.uc3m.es>

Trevor Burbridge
British Telecom
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: trevor.burbridge@bt.com

Sam Crawford
SamKnows

Email: sam@samknows.com

Philip Eardley
British Telecom
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: philip.eardley@bt.com

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

Email: acmorton@att.com

IPPM
Internet-Draft
Intended status: Standards Track
Expires: August 29, 2013

Y. Cui
E. Bi
K. Pentikousis, Ed.
Huawei Technologies
February 25, 2013

Network Performance Measurement for IPsec
draft-bi-ippm-ipsec-01.txt

Abstract

IPsec is a mature technology with several interoperable implementations. Indeed, the use of IPsec tunnels is increasingly gaining popularity in several deployment scenarios, not the least in what used to be solely areas of traditional telecommunication protocols. Wider deployment calls for mechanisms and methods that enable tunnel end-users, as well as operators, to measure one-way and two-way network performance. Unfortunately, however, standard IP performance measurement security mechanisms cannot be readily used with IPsec. This document makes the case for employing IPsec to protect O/TWAMP and proposes a method which combines IKEv2 and O/TWAMP as defined in RFC 4656 and RFC 5357, respectively. This specification aims, on the one hand, to ensure that O/TWAMP can be secured, while on the other hand, it extends the applicability of O/TWAMP to networks that have already deployed IPsec.

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 29, 2013.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
2. Terminology used in this document	4
3. Motivation	4
3.1. O/TWAMP-Control Security	5
3.2. O/TWAMP-Test Security	6
3.3. O/TWAMP Security Root	6
3.4. Co-existence of O/TWAMP and IPsec	6
4. O/TWAMP over IPsec	7
5. Others	11
6. Security Considerations	11
7. IANA Considerations	11
8. Acknowledgments	11
9. References	11
9.1. Normative References	11
9.2. Informative References	12
Authors' Addresses	12

1. Introduction

The active measurement protocols OWAMP [RFC4656] and TWAMP [RFC5357] can be used to measure network performance parameters, such as latency, bandwidth, and packet loss by sending probe packets and monitoring their experience in the network. In order to guarantee the accuracy of network measurement results, security aspects must be considered, otherwise, attacks may occur and authenticity may be violated. For example, if no protection is provided, an adversary in the middle may modify packet timestamps, thus altering the measurement results.

Cryptographic security mechanisms, such as IPsec, have been considered during the early stage of working towards the definition of the two protocols mentioned above. However, due to several reasons, it was preferred to avoid tying the development and deployment of O/TWAMP protocols to such security mechanisms. In practice, for many networks, the issues listed in [RFC4656], Sec. 6.6 with respect to IPsec are still valid. However, we expect that in the near future IPsec will be deployed in many more hosts and networks than today. For example, IPsec tunnels may be used to secure wireless channels. In this case, what we are interested in is measuring network performance specifically for the traffic carried by the tunnel, not in general over of the wireless channel. Therefore, in this document we attempt to make the case that for networks where wide deployment of IPsec and other security mechanisms is mandatory for a variety of reasons, there are increasingly more use cases in which IPsec and O/TWAMP protocols are needed simultaneously. In other words, we argue that it is now time to specify how O/TWAMP can be used in a network environment where IPsec is already deployed. In such an environment, measuring IP performance over IPsec tunnels with O/TWAMP is an important tool for operators.

Another advantage of IPsec key exchange protocol may be that it is not necessary to use distinct keys in OWAMP-Control and OWAMP-Test layers. One key for encryption and another for authentication is sufficient for both the Control and Test layers. This obviates the need to generate two keys and could reduce the complexity of O/TWAMP protocols in this environment. This observation comes from the fact that separate session keys in Control and Test layers are designed for preventing reflection attacks when employing the current mechanism. Once IPsec is employed, such a potential threat is alleviated. Note that this will be very useful in the environments where IPsec capability has been supported.

2. Terminology used in this document

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

3. Motivation

Let us first consider why the reasons originally listed in [RFC4656] Sec. 6.6 may not apply today in many cases. First, the argument made is that partial authentication in O/TWAMP authentication mode is not possible with IPsec. IPsec indeed cannot authenticate only a part of a packet. However, in an environment where IPsec is already deployed and actively used, partial authentication of O/TWAMP contradicts the operational reasons dictating the use of IPsec. At the same time, this limits the applicability and use of O/TWAMP in networks using IPsec.

The second argument made is the need to keep separate deployment paths between O/TWAMP and IPsec. In several currently deployed types of networks, IPsec is widely used to protect the data and signaling planes. For example, in mobile telecommunication networks, the deployment rate of IPsec exceeds 95% with respect to the LTE serving network. In older technology cellular networks, such as UMTS and GSM, IPsec use penetration is lower, but still quite significant. Additionally, there is a great number of IPsec-based VPN applications which are widely used in business applications to provide end-to-end, or host-to-host security over IEEE 802.11 wireless LANs. At the same time, lots of standardized protocols make use of IPsec/IKE, including MIPv4/v6, HIP, SCTP, BGP, NAT and SIP, just to name a few.

Third, with respect to the support of IPsec in lightweight embedded devices, nowadays, a large number of limited-resource and low-cost devices, such as Ethernet switches, DSL modems, and other such devices come with support for IPsec "out of the box". Therefore concerns about implementation, although likely valid a decade ago, are not well founded today.

Fourth, everyday use of IPsec applications by field technicians, on the one hand, and good understanding of the IPsec API by many programmers, on the other, should not be anymore a reason for concern. On the contrary: By now, IPsec open source code is available for anyone who wants to use it. Therefore, although IPsec does need a certain level of expertise to deal with it, in practice, most competent technical personnel and programmers have no problems using it on a daily basis.

O/TWAMP actually consists of two inter-related protocols: O/TWAMP-Control and O/TWAMP-Test. O/TWAMP-Control is used to initiate, start, and stop test sessions and to fetch their results, whereas O/TWAMP-Test is used to exchange test packets between two measurement nodes. In the following subsections we consider security for each one separately and then make the case for using them over IPsec.

3.1. O/TWAMP-Control Security

O/TWAMP uses a simple cryptographic protocol which relies on AES-CBC for confidentiality and on HMAC-SHA1 truncated to 128 bits for message authentication. Three modes of operation are supported: unauthenticated, authenticated, and encrypted. The authenticated and encrypted modes require that endpoints possess a shared secret, typically a passphrase. The secret key is derived from the passphrase using a password-based key derivation function PBKDF2 (PKCS#5) [RFC2898].

In the unauthenticated mode, the security parameters are left unused. In the authenticated and encrypted modes, security parameters are negotiated during the control connection establishment. Before the client can send commands to a server, it has to establish a connection to the server. Then, the client opens a TCP connection to the server on the well-known port number 861. The server responds with a server greeting, which contains the Challenge, Mode, Salt and Count. If the client wants to establish the connection, it responds with a Set-Up-Response message, wherein the KeyID, Token and Client IV are included. The Token is the concatenation of a 16-octet challenge, a 16-octet AES Session-key used for encryption, and a 32-octet HMAC-SHA1 Session-key used for authentication. The token itself is encrypted using AES in Cipher Block Chaining (AES-CBC).

Encryption is performed using a key derived from the shared secret associated with KeyID. In the authenticated and encrypted modes, all further communications are encrypted using the AES Session-key and authenticated with the HMAC Session-key. The client encrypts everything it transmits through the just-established O/TWAMP-Control connection using stream encryption with Client-IV as the IV. Correspondingly, the server encrypts its side of the connection using Server-IV as the IV. The IVs themselves are transmitted in cleartext. Encryption starts with the block immediately following the block containing the IV.

The AES Session-key and HMAC Session-key are generated randomly by the client. The HMAC Session-key is communicated along with the AES Session-key during O/TWAMP-Control connection setup. The HMAC Session-key is derived independently of the AES Session-key.

3.2. O/TWAMP-Test Security

The O/TWAMP-Test protocol runs over UDP, using sender and receiver IP and port numbers negotiated during the Request-Session exchange. As with O/TWAMP-Control, O/TWAMP-Test has three modes: unauthenticated, authenticated, and encrypted. All O/TWAMP-Test sessions that are spawned by an O/TWAMP-Control session inherit its mode.

The O/TWAMP-Test packet format is the same in authenticated and encrypted modes. The encryption and authentication operations are, however, different. Similarly with the respective O/TWAMP-Control session, each O/TWAMP-Test session has two keys: an AES Session-key and an HMAC Session-key. However, there is a difference in how the keys are obtained. In the case of O/TWAMP-Control, the keys are generated by the client and communicated (as part of the Token) during connection setup through the Set-Up-Response message. In the case of O/TWAMP-Test, the keys are derived from the O/TWAMP-Control keys and the session identifier (SID), as inputs of the key derivation function (KDF). The O/TWAMP-Test AES Session-key is generated by using the O/TWAMP-Control AES Session-key, with the 16-octet session identifier (SID), for encrypting and decrypting the packets of the particular O/TWAMP-Test session. The O/TWAMP-Test HMAC Session-key is generated by using the O/TWAMP-Control HMAC Session-key, with the 16-octet session identifier (SID), for authenticating the packets of the particular O/TWAMP-Test session.

3.3. O/TWAMP Security Root

As discussed above, the AES Session-key and HMAC Session-key used in the O/TWAMP-Test protocol are derived from the AES Session-key and HMAC Session-key which are used in O/TWAMP-Control protocol. The AES Session-key and HMAC Session-key used in the O/TWAMP-Control protocol are generated randomly by the client, and encrypted with the shared secret associated with KeyID. Therefore, the security root is the shared secret key. Thus, key provision and management are complicated and need to be taken care of appropriately. Comparatively, a certificate-based approach in IKEv2/IPsec can automatically manage the security root and solve this problem.

3.4. Co-existence of O/TWAMP and IPsec

According to [RFC4656] "[t]he deployment paths of IPsec and OWAMP could be separate if OWAMP does not depend on IPsec." The problem may occur in practice is that the security mechanism of O/TWAMP and IPsec cannot co-exist at the same time. IPsec provides confidentiality and data integrity to IP datagrams. Distinct protocols are provided: Authentication Header (AH), Encapsulating Security Payload (ESP) and Internet Key Exchange (IKE v1/v2). Only

integrity protection can be provided with AH. Both integrity and encryption can be provided with ESP. The IKE Protocol is used for dynamical key negotiation and automatic key management.

When the sender and receiver implement O/TWAMP over IPsec, they need to agree on a shared key during the establishment of the IPsec tunnel; subsequently all IP packets sent by the sender are protected. If the AH protocol is used, IP packets are transmitted in plaintext. The authentication part covers the entire packet. So all test information, such as UDP port number, and the test results will be visible to any attacker, which can intercept these test packets, and introduce errors or forge packets that may be injected during the transmission. In order to avoid this attack, the receiver must validate the integrity of these packets with the negotiated secret key. If ESP is used, IP packets are encrypted, and hence no other than the receiver can use the IPsec secret key and decrypt the IP packet, and then it can obtain the test data to assess the IP network performance based on the measurements. So both the sender and receiver must support IPsec to generate the security secret key of IPsec.

In the current implementation of O/TWAMP, after the test packets are received by the receiver, it cannot execute active measurement over IPsec. That is because the receiver knows only the shared secret key but not the IPsec key, while the test packets are protected by the IPsec key ultimately. Therefore, it needs to be considered how to measure IP network performance in an IPsec tunnel with O/TWAMP. Without this functionality, the use of OWAMP and TWAMP over IPsec is hindered.

Of course, backward compatibility should be considered, as well. That is, the intrinsic security method based on shared key as specified in the O/TWAMP standards can also fit the other platforms. There should be no impact on the current security mechanisms defined in O/TWAMP for other use cases. This document describes a possible solution to this problem which takes advantage of the secret key derived by IPsec, to provision the key needed in RFC 4656 and RFC 5357.

4. O/TWAMP over IPsec

A security method based on a shared secret key has been defined in O/TWAMP [RFC4656][RFC5357]. In this section, in order to employ O/TWAMP over IPsec, a method of binding O/TWAMP and IKEv2 is described, for those both the sender and receiver supporting the IPsec protocols. The shared key used in the security of O/TWAMP is derived from IPsec [RFC5996]. If the AH protocol is used, the IP

packets are transmitted in plaintext. All of O/TWAMP is integrity-protected by IPsec. Even if the peers choose to opt for the unauthenticated mode, IPsec integrity protection is extended to O/TWAMP. In the authenticated and encrypted modes, the shared secret can be derived from IKE SA or IPsec SA. If the shared secret key is derived from IKE SA, SKEYSEED must be generated firstly. SKEYSEED and its derivatives are computed as per [RFC5996], where prf is a pseudorandom function:

$$\text{SKEYSEED} = \text{prf}(\text{Ni} \parallel \text{Nr}, g^{\text{ir}})$$

Ni and Nr are nonces, negotiated during initial exchange. g^{ir} is the shared secret from the ephemeral Diffie-Hellman exchange and is represented as a string of octets. SKEYSEED can be used as the shared secret key directly, then the shared key is equal to SKEYSEED. Alternatively, the shared secret key can be generated as follows:

Shared secret key=PRF{ SKEYSEED, Session ID}

wherein the session ID is the SID agreed during the O/TWAMP-Test protocol.

If the shared secret key is derived from IPsec SA, the shared secret key can be equal to KEYMAT, wherein

$$\text{KEYMAT} = \text{prf}+(\text{SK}_d, \text{Ni} \parallel \text{Nr})$$

The term "prf+" describes a function that outputs a pseudorandom stream based on the inputs to a prf [RFC5996]; or the shared secret key can be generated as follows:

Shared secret key=PRF{ KEYMAT, Session ID}

wherein the session ID is the SID agreed during the O/TWAMP-Test protocol.

There are some cases for rekeying IKE SA and IPsec SA, after which the corresponding key of SA is updated. Generally ESP and AH SAs always exist in pairs, with one SA in each direction. If the SA is deleted, the key generated from the IKE SA or IPsec SA should also be updated.

As discussed above, a binding association between the key generated from IPsec and the shared secret key needs to be considered. SA can be identified by SPI and protocol uniquely for a given sender and a receiver. So these parameters should be agreed upon during the O/TWAMP protocol. When the sender and receiver execute O/TWAMP protocol to negotiate integrity key, the IPsec protocol and SPI

should be checked. Only if two parameters are matched with the information of IPsec, should the O/TWAMP connection be established. As illustrated in Fig. 1, the SPI and protocol type are included in the server greeting of the O/TWAMP-Control protocol. After the client receives the greeting, it closes the connection if it receives a greeting with an erroneous SPI and protocol value. Otherwise, the client responds with the following Set-Up-Response message and generates the shared secret key. This message exchange flow is illustrated as Fig. 1.

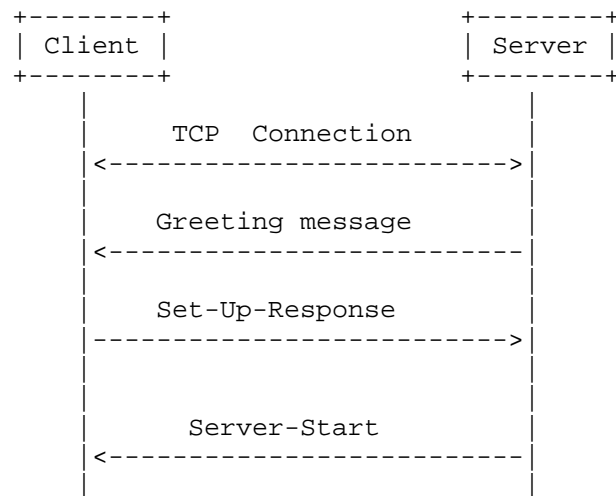


Figure 1. The procedure of O/TWAMP-Control

The format of server greeting is illustrated in Fig. 2. The unused 12 octets are used to carry the new parameter: protocol and SPIs.

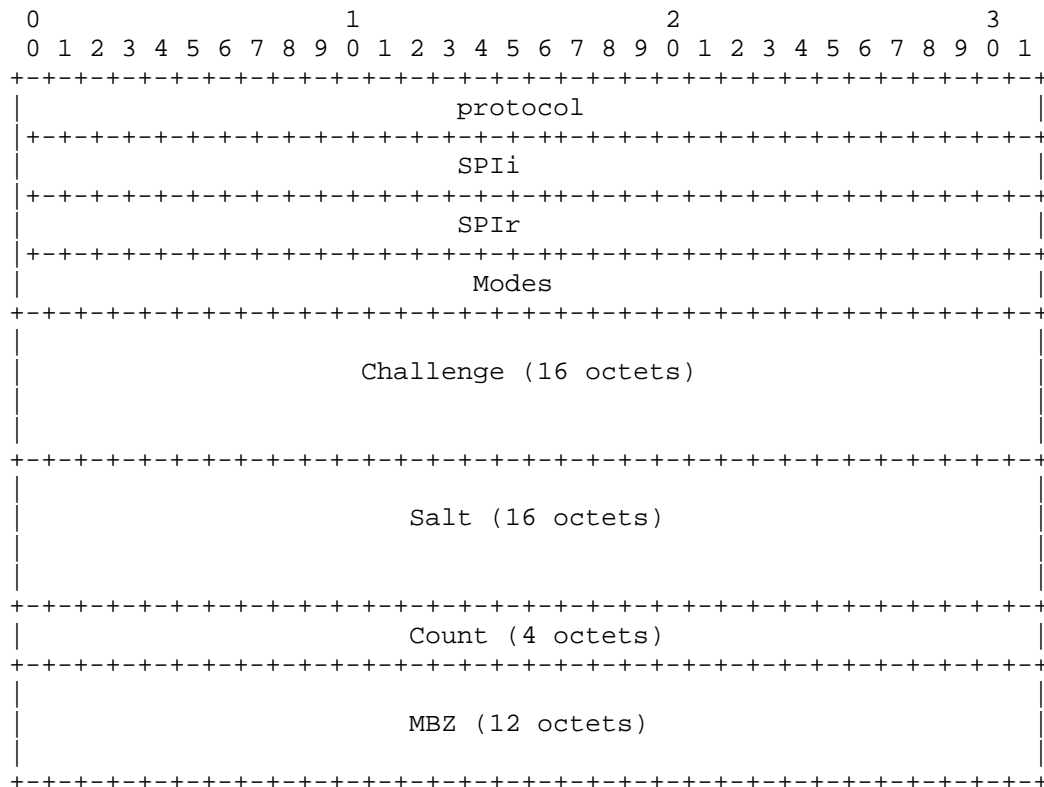


Figure 2. The format of server greeting

In ESP, when the IP packets are encrypted, no other than the receiver can use the IPsec key and decrypt the IP packets. It gains the test data to process measurement IP performance. In this case, the IPsec tunnel between the sender and receiver provides additional security. Even if the peers choose the unauthenticated mode, IPsec encryption and integrity protection is provided to O/TWAMP. If the sender and receiver also want to use authenticated or encrypted mode, the shared secret can be also derived from IKE SA or IPsec SA. The method of key generation and binding association is the same as AH protocol mode.

Besides, there is encryption-only configuration in ESP, though not recommended due to its limitations. Since it does not produce integrity key in this case, either encryption-only ESP should be prohibited for O/TWAMP, or a decryption failure should be distinguished due to possible integrity attack.

5. Others

The community may want to revisit the arguments listed in [RFC4656], Sec. 6.6. Other widely-used Internet security mechanisms, such as TLS and DTLS, may also be considered for future use over and above of what is already specified in O/TWAMP.

6. Security Considerations

As the shared secret key is derived from IPsec, the key derivation algorithm strength and limitations are as per [RFC5996]. The strength of a key derived from a Diffie-Hellman exchange using any of the groups defined here depends on the inherent strength of the group, the size of the exponent used, and the entropy provided by the random number generator employed. The strength of all keys and implementation vulnerabilities, particularly DoS attacks are as defined in [RFC5996].

7. IANA Considerations

There may be IANA considerations for allocating additional value for these options. The values of the protocol field needed to be assigned from the numbering space.

8. Acknowledgments

We would like to thank Eric Chen and Yakov Stein for their comments, and Al Morton for pointing to previous work discussed in IPPM WG.

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

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen,
"Internet Key Exchange Protocol Version 2 (IKEv2)",
RFC 5996, September 2010.

9.2. Informative References

[RFC2898] Kaliski, B., "PKCS #5: Password-Based Cryptography
Specification Version 2.0", RFC 2898, September 2000.

Authors' Addresses

Yang Cui
Huawei Technologies
Huawei Building, Q20, No.156, Rd. BeiQing
Haidian District, Beijing 100095
P. R. China

Email: cuiyang@huawei.com

Emily Bi
Huawei Technologies
Huawei Building, Q20, No.156, Rd. BeiQing
Haidian District, Beijing 100095
P. R. China

Phone: +86-10-82881907
Email: bixiaoyu@huawei.com

Kostas Pentikousis (editor)
Huawei Technologies
Carnotstrasse 4
10587 Berlin
Germany

Email: k.pentikousis@huawei.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 5, 2013

A. Morton
AT&T Labs
February 1, 2013

Rate Measurement Test Protocol Problem Statement
draft-ietf-ippm-rate-problem-02

Abstract

There is a rate measurement scenario which has wide-spread attention of Internet access subscribers and seemingly all industry players, including regulators. This memo presents an access rate-measurement problem statement for test protocols to measure IP Performance Metrics. Key test protocol aspects require the ability to control packet size on the tested path and enable asymmetrical packet size testing in a controller-responder architecture.

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 5, 2013.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	3
2. Purpose and Scope	3
3. Active Rate Measurement	5
4. Measurement Method Categories	7
5. Test Protocol Control & Generation Requirements	8
6. Security Considerations	9
7. IANA Considerations	9
8. Acknowledgements	9
9. Appendix	9
10. References	10
10.1. Normative References	10
10.2. Informative References	10
Author's Address	11

1. Introduction

There are many possible rate measurement scenarios. This memo describes one rate measurement problem and presents a rate-measurement problem statement for test protocols to measure IP Performance Metrics (IPPM).

The access-rate scenario or use case has wide-spread attention of Internet access subscribers and seemingly all Internet industry players, including regulators. This problem is being approached with many different measurement methods. This memo

2. Purpose and Scope

The scope and purpose of this memo is to define the measurement problem statement for test protocols conducting access rate measurement on production networks. Relevant test protocols include [RFC4656] and [RFC5357]), but the problem is stated in a general way so that it can be addressed by any existing test protocol, such as [RFC6812].

This memo discusses possibilities for methods of measurement, but does not specify exact methods which would normally be part of the solution, not the problem.

We characterize the access rate measurement scenario as follows:

- o The Access portion of the network is the focus of this problem statement. The user typically subscribes to a service with bi-directional access partly described by rates in bits per second. The rates may be expressed as raw capacity or restricted capacity as described in [RFC6703]. These are the quantities that must be measured according to one or more standard metrics for which methods must also be agreed as a part of the solution.
- o Referring to the reference path defined in [I-D.morton-ippm-lmap-path], possible measurement points include a Subscriber's host (mp000), the access service demarcation point (mp100), Intra IP access where a globally routable address is present (mp150), or the gateway between the measured access network and other networks (mp190).
- o Rates at the edge of the network are several orders of magnitude less than aggregation and core portions.
- o Asymmetrical ingress and egress rates are prevalent.

- o Extremely large scale of access services requires low complexity devices participating at the user end of the path.

Today, the majority of widely deployed access services achieve rates less than 100 Mbit/s, and this is the order of magnitude for which a solution is sought now.

This problem statement assumes that the most-likely bottleneck device or link is adjacent to the remote (user-end) measurement device, or is within one or two router/switch hops of the remote measurement device.

Other use cases for rate measurement involve situations where the packet switching and transport facilities are leased by one operator from another and the actual capacity available cannot be directly determined (e.g., from device interface utilization). These scenarios could include mobile backhaul, Ethernet Service access networks, and/or extensions of layer 2 or layer 3 networks. The results of rate measurements in such cases could be employed to select alternate routing, investigate whether capacity meets some previous agreement, and/or adapt the rate of traffic sources if a capacity bottleneck is found via the rate measurement. In the case of aggregated leased networks, available capacity may also be asymmetric. In these cases, the tester is assumed to have a sender and receiver location under their control. We refer to this scenario below as the aggregated leased network case.

Support of active measurement methods will be addressed here, consistent with the IPPM working group's traditional charter. Active measurements require synthetic traffic dedicated to testing, and do not use user traffic.

The actual path used by traffic may influence the rate measurement results for some forms of access, as it may differ between user and test traffic if the test traffic has different characteristics, primarily in terms of the packets themselves (the Type-P described in [RFC2330]).

There are several aspects of Type-P where user traffic may be examined and directed to special treatment that may affect transmission rates. The possibilities include:

- o Packet length
- o IP addresses used
- o Transport protocol used (where TCP packets may be routed differently from UDP)

- o Transport Protocol port numbers used

This issue requires further discussion when specific solutions/methods of measurement are proposed, but for this problem statement it is sufficient to Identify the problem and indicate that the solution may require an extremely close emulation of user traffic, in terms of the factors above.

Although the user may have multiple instances of network access available to them, the primary problem scope is to measure one form of access at a time. It is plausible that a solution for the single access problem will be applicable to simultaneous measurement of multiple access instances, but discussion of this is beyond the current scope.

A key consideration is whether active measurements will be conducted with user traffic present (In-Service testing), or not present (Out-of-Service testing), such as during pre-service testing or maintenance that interrupts service temporarily. Out-of-Service testing includes activities described as "service commissioning", "service activation", and "planned maintenance". Opportunistic In-Service testing when there is no user traffic present throughout the test interval is essentially equivalent to Out-of-Service testing. Both In-Service and Out-of-Service testing are within the scope of this problem.

It is a non-goal to solve the measurement protocol specification problem in this memo.

It is a non-goal to standardize methods of measurement in this memo. However, the problem statement will mandate that support for one or more categories of rate measurement methods and adequate control features for the methods in the test protocol.

3. Active Rate Measurement

This section lists features of active measurement methods needed to measure access rates in production networks.

Test coordination between source and destination devices through control messages and other basic capabilities described in the methods of IPPM RFCs [RFC2679][RFC2680] are taken as given (these could be listed later, if desired).

Most forms of active testing intrude on user performance to some degree. One key tenet of IPPM methods is to minimize test traffic effects on user traffic in the production network. Section 5 of

[RFC2680] lists the problems with high measurement traffic rates, and the most relevant for rate measurement is the tendency for measurement traffic to skew the results, followed by the possibility of introducing congestion on the access link. Obviously, categories of rate measurement methods that use less active test traffic than others with similar accuracy SHALL be preferred for In-Service testing.

On the other hand, Out-of-Service tests where the test path shares no links with In-Service user traffic have none of the congestion or skew concerns, but these tests must address other practical concerns such as conducting measurements within a reasonable time from the tester's point of view. Out-of-Service tests where some part of the test path is shared with In-Service traffic MUST respect the In-Service constraints.

The ****intended metrics to be measured**** have strong influence over the categories of measurement methods required. For example, using the terminology of [RFC5136], it may be possible to measure a Path Capacity Metric while In-Service if the level of background (user) traffic can be assessed and included in the reported result.

The measurement ***architecture*** MAY be either of one-way (e.g., [RFC4656]) or two-way (e.g., [RFC5357]), but the scale and complexity aspects of end-user or aggregated access measurement clearly favor two-way (with low-complexity user-end device and round-trip results collection, as found in [RFC5357]). However, the asymmetric rates of many access services mean that the measurement system MUST be able to evaluate performance in each direction of transmission. In the two-way architecture, it is expected that both end devices MUST include the ability to launch test streams and collect the results of measurements in both (one-way) directions of transmission (this requirement is consistent with previous protocol specifications, and it is not a unique problem for rate measurements).

The following paragraphs describe features for the roles of test packet SENDER, RECEIVER, and results REPORTER.

SENDER:

Generate streams of test packets with various characteristics as desired (see Section 4). The SENDER may be located at the user end of the access path, or may be located elsewhere in the production network, such as at one end of an aggregated leased network segment.

RECEIVER:

Collect streams of test packets with various characteristics (as

described above), and make the measurements necessary to support rate measurement at the other end of an end-user access or aggregated leased network segment.

REPORTER:

Use information from test packets and local processes to measure delivered packet rates.

4. Measurement Method Categories

The design of rate measurement methods can be divided into two phases: test stream design and measurement (SENDER and RECEIVER), and a follow-up phase for analysis of the measurement to produce results (REPORTER). The measurement protocol that addresses this problem MUST only serve the test stream generation and measurement functions.

For the purposes of this problem statement, we categorize the many possibilities for rate measurement stream generation as follows:

1. Packet pairs, with fixed intra-pair packet spacing and fixed or random time intervals between pairs in a test stream.
2. Multiple streams of packet pairs, with a range of intra-pair spacing and inter-pair intervals.
3. One or more packet ensembles in a test stream, using a fixed ensemble size in packets and one or more fixed intra-ensemble packet spacings (including zero spacing).
4. One or more packet chirps, where intra-packet spacing typically decreases between adjacent packets in the same chirp and each pair of packets represents a rate for testing purposes.

For all categories, the test protocol MUST support:

1. Variable payload lengths among packet streams
2. Variable length (in packets) among packet streams or ensembles
3. Variable IP header markings among packet streams
4. Choice of UDP transport and variable port numbers, OR, choice of TCP transport and variable port numbers for two-way architectures only, OR BOTH.

5. Variable number of packets-pairs, ensembles, or streams used in a test session

The items above are additional variables that the test protocol MUST be able to identify and control.

The test protocol SHALL support test packet ensemble generation (category 3), as this appears to minimize the demands on measurement accuracy. Other stream generation categories are OPTIONAL.

>>>>>>

Note: For measurement systems employing TCP Transport protocol, the ability to generate specific stream characteristics requires a sender with the ability to establish and prime the connection such that the desired stream characteristics are allowed. See Mathis' work in progress for more background [I-D.mathis-ippm-model-based-metrics]. The general requirement statements needed to describe an "open-loop" TCP sender require some additional discussion.

It may also be useful to specify a control for Bulk Transfer Capacity measurement with fully-specified TCP senders and receivers, as envisioned in [RFC3148], but this would be a brute-force assessment which does not follow the conservative tenets of IPPM measurement [RFC2330].

>>>>>>

Measurements for each test packet transferred between SENDER and RECEIVER MUST be compliant with the singleton measurement methods described in IPPM RFCs [RFC2679][RFC2680] (these could be listed later, if desired). The time-stamp information or loss/arrival status for each packet MUST be available for communication to the protocol entity that collects results.

5. Test Protocol Control & Generation Requirements

Essentially, the test protocol MUST support the measurement features described in the sections above. This requires:

1. Communicating all test variables to the Sender and Receiver
2. Results collection in a one-way architecture
3. Remote device control for both one-way and two-way architectures

4. Asymmetric and/or pseudo-one-way test capability in a two-way measurement architecture

The ability to control packet size on the tested path and enable asymmetrical packet size testing in a two-way architecture are REQUIRED.

The test protocol SHOULD enable measurement of the [RFC5136] Capacity metric, either Out-of-Service, In-Service, or both. Other [RFC5136] metrics are OPTIONAL.

6. Security Considerations

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

There may be a serious issue if a proprietary Service Level Agreement involved with the access network segment provider were somehow leaked in the process of rate measurement. To address this, test protocols SHOULD NOT convey this information in a way that could be discovered by unauthorized parties.

7. IANA Considerations

This memo makes no requests of IANA.

8. Acknowledgements

Dave McDysan provided comments and text for the aggregated leased use case. Yaakov Stein suggested many considerations to address, including the In-Service vs. Out-of-Service distinction and its implication on test traffic limits and protocols. Bill Cervený and Marcelo Bagnulo have contributed insightful, clarifying comments that made this a better draft.

9. Appendix

This Appendix was proposed to briefly summarize previous rate measurement experience. (There is a large body of research on rate measurement, so there is a question of what to include and what to omit. Suggestions are welcome.)

10. References

10.1. Normative References

- [RFC1305] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation", RFC 1305, March 1992.
- [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.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.
- [RFC5618] Morton, A. and K. Hedayat, "Mixed Security Mode for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5618, August 2009.
- [RFC5938] Morton, A. and M. Chiba, "Individual Session Control Feature for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5938, August 2010.
- [RFC6038] Morton, A. and L. Ciavattone, "Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features", RFC 6038, October 2010.
- [RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View", RFC 6703, August 2012.

10.2. Informative References

- [I-D.mathis-ippm-model-based-metrics]
Mathis, M., "Model Based Internet Performance Metrics",

draft-mathis-ippm-model-based-metrics-00 (work in progress), October 2012.

[I-D.morton-ippm-lmap-path]

Bagnulo, M., Burbridge, T., Crawford, S., Eardley, P., and A. Morton, "A Reference Path and Measurement Points for LMAP", draft-morton-ippm-lmap-path-00 (work in progress), January 2013.

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

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

[RFC6812] Chiba, M., Clemm, A., Medley, S., Salowey, J., Thombare, S., and E. Yedavalli, "Cisco Service-Level Assurance Protocol", RFC 6812, January 2013.

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: August 21, 2013

L. Ciavattone
AT&T Labs
R. Geib
Deutsche Telekom
A. Morton
AT&T Labs
M. Wieser
Technical University Darmstadt
February 17, 2013

Test Plan and Results for Advancing RFC 2680 on the Standards Track
draft-ietf-ippm-testplan-rfc2680-02

Abstract

This memo proposes to advance a performance metric RFC along the standards track, specifically RFC 2680 on One-way Loss 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 2680.

In this version, the results are presented in the R-tool output form. Beautification is future work.

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 21, 2013.

Copyright Notice

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

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

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

Table of Contents

1. Introduction	4
1.1. RFC 2680 Coverage	5
2. A Definition-centric metric advancement process	5
3. Test configuration	6
4. Error Calibration, RFC 2680	10
4.1. Clock Synchronization Calibration	10
4.2. Packet Loss Determination Error	10
5. Pre-determined Limits on Equivalence	11
6. Tests to evaluate RFC 2680 Specifications	12
6.1. One-way Loss, ADK Sample Comparison	12
6.1.1. 340B/Periodic Cross-imp. results	13
6.1.2. 64B/Periodic Cross-imp. results	14
6.1.3. 64B/Poisson Cross-imp. results	15
6.1.4. Conclusions on the ADK Results for One-way Packet Loss	16
6.2. One-way Loss, Delay threshold	16
6.2.1. NetProbe results for Loss Threshold	17
6.2.2. Perfas Results for Loss Threshold	18
6.2.3. Conclusions for Loss Threshold	18
6.3. One-way Loss with Out-of-Order Arrival	18
6.4. Poisson Sending Process Evaluation	19
6.4.1. NetProbe Results	20
6.4.2. Perfas Results	21
6.4.3. Conclusions for Goodness-of-Fit	23
6.5. Implementation of Statistics for One-way Delay	23
7. Conclusions for RFC 2680bis	23
8. Security Considerations	24
9. IANA Considerations	24
10. Acknowledgements	24
11. References	24
11.1. Normative References	24
11.2. Informative References	26
Authors' Addresses	26

1. Introduction

The IETF (IP Performance Metrics working group, IPPM) has considered how to advance their metrics along the standards track since 2001.

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 consensus [RFC6576] 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 [RFC2680].

In particular, this memo documents consensus 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.

Although the conclusion reached through testing is that [RFC2680] should be advanced on the Standards Track with modifications, the revised text of RFC 2680bis is not yet ready for review. Therefore, this memo documents the information to support [RFC2680] advancement,

and the approval of RFC2680bis is left for future action.

1.1. RFC 2680 Coverage

This plan is intended to cover all critical requirements and sections of [RFC2680].

Note that there are only five instances of the requirement term "MUST" in [RFC2680] outside of the boilerplate and [RFC2119] reference.

Material may be added as it is "discovered" (apparently, not all requirements use requirements language).

2. A Definition-centric metric advancement process

The process described in Section 3.5 of [RFC6576] 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.

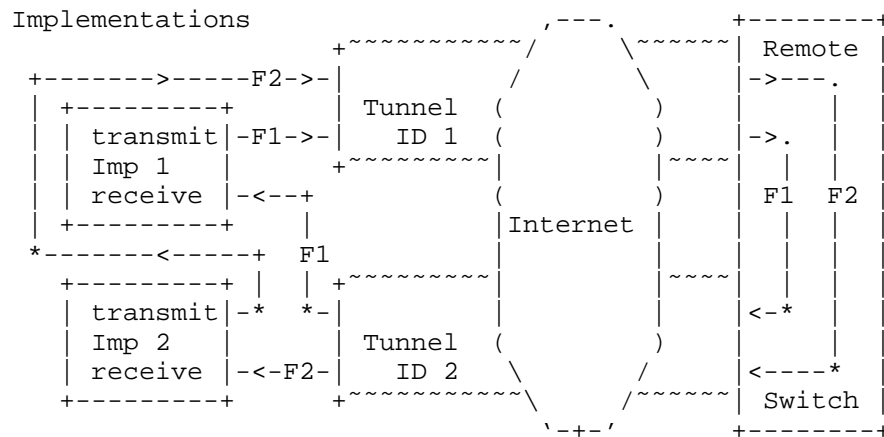
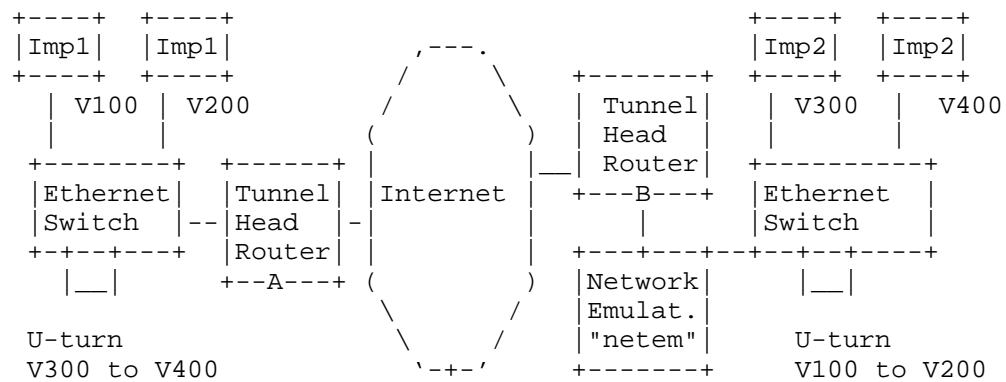
Finally, all the findings MUST be documented in a report that can support advancement on the standards track, similar to those described in [RFC5657]. The list of measurement devices used in testing satisfies the implementation requirement, while the test results provide information on the quality of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

3. Test configuration

One metric implementation used was NetProbe version 5.8.5, (an earlier version is used in the WIPM system and deployed world-wide [WIPM]). NetProbe uses UDP packets of variable size, and can produce test streams with Periodic [RFC3432] or Poisson [RFC2330] sample distributions.

The other metric implementation used was Perfas+ version 3.1, developed by Deutsche Telekom [Perfas]. Perfas+ uses UDP unicast packets of variable size (but supports also TCP and multicast). Test streams with periodic, Poisson or uniform sample distributions may be used.

Figure 1 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 [RFC6576].



Illustrations of a test setup with a bi-directional tunnel. The upper diagram emphasizes the VLAN connectivity and geographical location. The lower diagram shows example flows traveling between two measurement implementations (for simplicity, only two flows 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, one pair of 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 14 Linux [<http://fedoraproject.org/>] with IP forwarding enabled and the "netem" Network emulator as part of the Fedora Kernel 2.6.35.11 [<http://www.linuxfoundation.org/collaborate/workgroups/networking/netem>] loaded and operating. Connectivity across the netem/Fedora host was accomplished by bridging Ethernet VLAN interfaces together with "brctl" commands (e.g., eth1.100 <-> eth2.100). The netem emulator was activated on one interface (eth1) and only operates on test streams traveling in one direction. In some tests, independent netem instances operated separately on each VLAN.

The links between the netem emulator host and router and switch were found to be 100baseTx-HD (100Mbps half duplex) as reported by "mii-tool" when the testing was complete. Use of Half Duplex was not intended, but probably added a small amount of delay variation that could have been avoided in full duplex mode.

Each individual test was run with common packet rates (1 pps, 10pps) Poisson/Periodic distributions, and IP packet sizes of 64, 340, and 500 Bytes.

For these tests, a stream of at least 300 packets were sent from Source to Destination in each implementation. Periodic streams (as per [RFC3432]) with 1 second spacing were used, except as noted.

As required in Section 2.8.1 of [RFC2680], packet Type-P must be reported. The packet 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.

With the L2TPv3 tunnel in use, 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-Packet-Loss-<StreamType>-Stream

With (Section 3.2. [RFC2680]) Metric Parameters:

- + Src, the IP address of a host (12.3.167.16 or 193.159.144.8)
- + Dst, the IP address of a host (193.159.144.8 or 12.3.167.16)
- + T0, a time

- + Tf, a time
- + lambda, a rate in reciprocal seconds
- + Thresh, a maximum waiting time in seconds (see Section 2.8.2 of [RFC2680]) and (Section 3.8. [RFC2680])

Metric Units: A sequence of pairs; the elements of each pair are:

- + T, a time, and
- + L, either a zero or a one

The values of T in the sequence are monotonic increasing. Note that T would be a valid parameter to the *singleton* Type-P-One-way-Packet-Loss, and that L would be a valid value of Type-P-One-way-Packet Loss (see Section 2 of [RFC2680]).

Also, Section 2.8.4 of [RFC2680] 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 traceroute is conducted in parallel at the outset of measurements.

Perfas+ does not support traceroute.

```
IPLGW#traceroute 193.159.144.8
```

```
Type escape sequence to abort.
```

```
Tracing the route to 193.159.144.8
```

```
 1 12.126.218.245 [AS 7018] 0 msec 0 msec 4 msec
 2 cr84.n54ny.ip.att.net (12.123.2.158) [AS 7018] 4 msec 4 msec
   cr83.n54ny.ip.att.net (12.123.2.26) [AS 7018] 4 msec
 3 cr1.n54ny.ip.att.net (12.122.105.49) [AS 7018] 4 msec
   cr2.n54ny.ip.att.net (12.122.115.93) [AS 7018] 0 msec
   cr1.n54ny.ip.att.net (12.122.105.49) [AS 7018] 0 msec
 4 n54ny02jt.ip.att.net (12.122.80.225) [AS 7018] 4 msec 0 msec
   n54ny02jt.ip.att.net (12.122.80.237) [AS 7018] 4 msec
 5 192.205.34.182 [AS 7018] 0 msec
   192.205.34.150 [AS 7018] 0 msec
   192.205.34.182 [AS 7018] 4 msec
 6 da-rg12-i.DA.DE.NET.DTAG.DE (62.154.1.30) [AS 3320] 88 msec 88 msec
88 msec
 7 217.89.29.62 [AS 3320] 88 msec 88 msec 88 msec
 8 217.89.29.55 [AS 3320] 88 msec 88 msec 88 msec
 9 * * *
```

It was only possible to conduct the traceroute for the measured path on one of the tunnel-head routers (the normal trace facilities of the measurement systems are confounded by the L2TPv3 tunnel encapsulation).

4. Error Calibration, RFC 2680

An implementation is required to report calibration results on clock synchronization in Section 2.8.3 of [RFC2680] (also required in Section 3.7 of [RFC2680] for sample metrics).

Also, it is recommended to report the probability that a packet successfully arriving at the destination network interface is incorrectly designated as lost due to resource exhaustion in Section 2.8.3 of [RFC2680].

4.1. Clock Synchronization Calibration

For NetProbe and Perfas+ clock synchronization test results, refer to Section 4 of [RFC6808].

4.2. Packet Loss Determination Error

Since both measurement implementations have resource limitations, it is theoretically possible that these limits could be exceeded and a

packet that arrived at the destination successfully might be discarded in error.

In previous test efforts [I-D.morton-ippm-advance-metrics], NetProbe produced 6 multicast streams with an aggregate bit rate over 53 Mbit/s, in order to characterize the 1-way capacity of a NISTNet-based emulator. Neither the emulator nor the pair of NetProbe implementations used in this testing dropped any packets in these streams.

The maximum load used here between any 2 NetProbe implementations was be 11.5 Mbit/s divided equally among 3 unicast test streams. We conclude that steady resource usage does not contribute error (additional loss) to the measurements.

5. Pre-determined Limits on Equivalence

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) [ADK] comparisons, the required confidence factor for the cross-implementation comparisons SHALL be the smallest of:

- o 0.95 confidence factor at 1 packet resolution, or
- o the smallest confidence factor (in combination with resolution) of the two same-implementation comparisons for the same test conditions (if the number of streams is sufficient to allow such comparisons).

For Anderson-Darling Goodness-of-Fit (ADGoF) [Radgof] comparisons, the required level of significance for the same-implementation Goodness-of-Fit (GoF) SHALL be 0.05 or 5%, as specified in Section 11.4 of [RFC2330]. This is equivalent to a 95% confidence factor.

6. Tests to evaluate RFC 2680 Specifications

This section describes some results from production network (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 similar contained in Appendix A.1 of [RFC6576] for One-way Delay.

6.1. One-way Loss, ADK Sample Comparison

This test determines if implementations produce results that appear to come from a common packet loss distribution, as an overall evaluation of Section 3 of [RFC2680], "A Definition for Samples of One-way Packet Loss". 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 2, 3, and 4 of [RFC2680].

By testing the extent to which the counts of one-way packet loss counts on different test streams of two [RFC2680] implementations appear to be from the same loss process, we reduce comparison steps because comparing the resulting summary statistics (as defined in Section 4 of [RFC2680]) would require a redundant set of equivalence evaluations. We can easily check whether the single statistic in Section 4 of [RFC2680] was implemented, and report on that fact.

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 packet loss singletons with 2 or more implementations, using identical options and network emulator settings (if used).
3. Measure a sample of one-way packet loss singletons with *four or more* instances of the *same* implementations, using identical options, noting that connectivity differences SHOULD be the same as for the cross implementation testing.
4. If less than ten test streams are available, skip to step 7.
5. Apply the ADK comparison procedures (see Appendix C of [RFC6576]) and determine the resolution and confidence factor for

distribution equivalence of each same-implementation comparison and each cross-implementation comparison.

6. 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.
7. Compare the cross-implementation ADK performance with the equivalence threshold determined in step 5 to determine if equivalence can be declared.

The common parameters used for tests in this section are:

The cross-implementation comparison uses a simple ADK analysis [Rtool] [Radk], where all NetProbe loss counts are compared with all Perfas+ loss results.

In the result analysis of this section:

- o All comparisons used 1 packet resolution.
- o No Correction Factors were applied.
- o The 0.95 confidence factor (1.960 for cross-implementation comparison) was used.

6.1.1. 340B/Periodic Cross-imp. results

Tests described in this section used:

- o IP header + payload = 340 octets
- o Periodic sampling at 1 packet per second
- o Test duration = 1200 seconds (during April 7, 2011, EDT)

The netem emulator was set for 100ms constant delay, with 10% loss ratio. In this experiment, the netem emulator was configured to operate independently on each VLAN and thus the emulator itself is a potential source of error when comparing streams that traverse the test path in different directions.


```
A07bps_loss <- c(114, 175, 138, 142, 181, 105) (NetProbe)
A07per_loss <- c(115, 128, 136, 127, 139, 138) (Perfas)

> A07bps_loss <- c(114, 175, 138, 142, 181, 105)
> A07per_loss <- c(115, 128, 136, 127, 139, 138)
>
> A07cross_loss_ADK <- adk.test(A07bps_loss, A07per_loss)
> A07cross_loss_ADK
Anderson-Darling k-sample test.
```

```
Number of samples: 2
Sample sizes: 6 6
Total number of values: 12
Number of unique values: 11
```

```
Mean of Anderson Darling Criterion: 1
Standard deviation of Anderson Darling Criterion: 0.6569
```

```
T = (Anderson Darling Criterion - mean)/sigma
```

Null Hypothesis: All samples come from a common population.

	t.obs	P-value	extrapolation
not adj. for ties	0.52043	0.20604	0
adj. for ties	0.62679	0.18607	0

The cross-implementation comparisons pass the ADK criterion.

6.1.2. 64B/Periodic Cross-imp. results

Tests described in this section used:

- o IP header + payload = 64 octets
- o Periodic sampling at 1 packet per second
- o Test duration = 300 seconds (during March 24, 2011, EDT)

The netem emulator was set for 0ms constant delay, with 10% loss ratio.

```
> M24per_loss <- c(42,34,35,35)          (Perfas)
> M24apd_23BC_loss <- c(27,39,29,24)      (NetProbe)
> M24apd_loss23BC_ADK <- adk.test(M24apd_23BC_loss,M24per_loss)
> M24apd_loss23BC_ADK
Anderson-Darling k-sample test.
```

```
Number of samples: 2
Sample sizes: 4 4
Total number of values: 8
Number of unique values: 7
```

```
Mean of Anderson Darling Criterion: 1
Standard deviation of Anderson Darling Criterion: 0.60978
```

```
T = (Anderson Darling Criterion - mean)/sigma
```

Null Hypothesis: All samples come from a common population.

	t.obs	P-value	extrapolation
not adj. for ties	0.76921	0.16200	0
adj. for ties	0.90935	0.14113	0

Warning: At least one sample size is less than 5.
p-values may not be very accurate.

The cross-implementation comparisons pass the ADK criterion.

6.1.3. 64B/Poisson Cross-imp. results

Tests described in this section used:

- o IP header + payload = 64 octets
- o Poisson sampling at $\lambda = 1$ packet per second
- o Test duration = 20 minutes (during April 27, 2011, EDT)

The netem configuration was 0ms delay and 10% loss, but there were two passes through an emulator for each stream, and loss emulation was present for 18 minutes of the 20 minute test .

```
A27aps_loss <- c(91,110,113,102,111,109,112,113) (NetProbe)
A27per_loss <- c(95,123,126,114) (Perfas)
```

```
A27cross_loss_ADK <- adk.test(A27aps_loss, A27per_loss)
```

```
> A27cross_loss_ADK
Anderson-Darling k-sample test.
```

```
Number of samples: 2
Sample sizes: 8 4
Total number of values: 12
Number of unique values: 11
```

```
Mean of Anderson Darling Criterion: 1
Standard deviation of Anderson Darling Criterion: 0.65642
```

```
T = (Anderson Darling Criterion - mean)/sigma
```

```
Null Hypothesis: All samples come from a common population.
```

	t.obs	P-value	extrapolation
not adj. for ties	2.15099	0.04145	0
adj. for ties	1.93129	0.05125	0

```
Warning: At least one sample size is less than 5.
p-values may not be very accurate.
>
```

The cross-implementation comparisons barely pass the ADK criterion at 95% = 1.960 when adjusting for ties.

6.1.4. Conclusions on the ADK Results for One-way Packet Loss

We conclude that the two implementations are capable of producing equivalent one-way packet loss measurements based on their interpretation of [RFC2680] .

6.2. One-way Loss, Delay threshold

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 2.8.2 of [RFC2680].

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 1.0 sec one-way constant delay in one direction of transmission.
3. measure (average) one-way delay with 2 or more implementations, using identical waiting time thresholds (Thresh) for loss set at 3 seconds.
4. configure the network emulator to add 3 sec one-way constant delay in one 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.

The common parameters used for tests in this section are:

- o IP header + payload = 64 octets
- o Poisson sampling at $\lambda = 1$ packet per second
- o Test duration = 900 seconds total (March 21)

The netem emulator was set to add constant delays as specified in the procedure above.

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 3 seconds, all packets with one-way delay >3 seconds are marked "Lost" and included in the Lost Packet list with their transmission time (as required in Section 3.3 of [RFC2680]). This resulted in 342 packets designated as lost in one of the test streams (with average delay = 3.091 sec).

6.2.2. Perfas Results for Loss Threshold

Perfas+ uses a fixed Loss Threshold which was not adjustable during this study. The Loss Threshold is approximately one minute, and emulation of a delay of this size was not attempted. However, it is possible to implement any delay threshold desired with a post-processing routine and subsequent analysis. Using this method, 195 packets would be declared lost (with average delay = 3.091 sec).

6.2.3. Conclusions for Loss Threshold

Both implementations assume that any constant delay value desired can be used as the Loss Threshold, since all delays are stored as a pair <Time, Delay> as required in [RFC2680]. This is a simple way to enforce the constant loss threshold envisioned in [RFC2680] (see specific section reference above). We take the position that the assumption of post-processing is compliant, and that the text of the RFC should be revised slightly to include this point.

6.3. One-way Loss with Out-of-Order Arrival

Section 3.6 of [RFC2680] indicates that implementations need to ensure that reordered packets are handled correctly using an uncapitalized "must". In essence, this is an implied requirement because the correct packet must be identified as lost if it fails to arrive before its delay threshold under all circumstances, and reordering is always a possibility on IP network paths. See [RFC4737] for the definition of reordering used in IETF standard-compliant measurements.

Using the procedure of section 6.1, the netem emulator was set to introduce significant delay (2000 ms) and delay variation (1000 ms), which was sufficient to produce packet reordering because each packet's emulated delay is independent from others, and 10% loss.

The tests described in this section used:

- o IP header + payload = 64 octets
- o Periodic sampling = 1 packet per second
- o Test duration = 600 seconds (during May 2, 2011, EDT)

```
> Y02aps_loss <- c(53,45,67,55)      (NetProbe)
> Y02per_loss <- c(59,62,67,69)      (Perfas)
> Y02cross_loss_ADK <- adk.test(Y02aps_loss, Y02per_loss)
> Y02cross_loss_ADK
Anderson-Darling k-sample test.
```

```
Number of samples: 2
Sample sizes: 4 4
Total number of values: 8
Number of unique values: 7
```

```
Mean of Anderson Darling Criterion: 1
Standard deviation of Anderson Darling Criterion: 0.60978
```

```
T = (Anderson Darling Criterion - mean)/sigma
```

```
Null Hypothesis: All samples come from a common population.
```

```

              t.obs P-value extrapolation
not adj. for ties 1.11282 0.11531          0
adj. for ties    1.19571 0.10616          0
```

```
Warning: At least one sample size is less than 5.
         p-values may not be very accurate.
>
```

The test results indicate that extensive reordering was present. Both implementations capture the extensive delay variation between adjacent packets. In NetProbe, packet arrival order is preserved in the raw measurement files, so an examination of arrival packet sequence numbers also indicates reordering.

Despite extensive continuous packet reordering present in the transmission path, the distributions of loss counts from the two implementations pass the ADK criterion at 95% = 1.960.

6.4. Poisson Sending Process Evaluation

Section 3.7 of [RFC2680] indicates that implementations need to ensure that their sending process is reasonably close to a classic Poisson distribution when used. Much more detail on sample distribution generation and Goodness-of-Fit testing is specified in Section 11.4 of [RFC2330] and the Appendix of [RFC2330].

In this section, each implementation's Poisson distribution is compared with an idealistic version of the distribution available in the base functionality of the R-tool for Statistical Analysis[Rtool],

and performed using the Anderson-Darling Goodness-of-Fit test package (ADGofTest) [Radgof]. The Goodness-of-Fit criterion derived from [RFC2330] requires a test statistic value $AD \leq 2.492$ for 5% significance. The Appendix of [RFC2330] also notes that there may be difficulty satisfying the ADGofTest when the sample includes many packets (when 8192 were used, the test always failed, but smaller sets of the stream passed).

Both implementations were configured to produce Poisson distributions with $\lambda = 1$ packet per second.

6.4.1. NetProbe Results

Section 11.4 of [RFC2330] suggests three possible measurement points to evaluate the Poisson distribution. The NetProbe analysis uses "user-level timestamps made just before or after the system call for transmitting the packet".

The statistical summary for two NetProbe streams is below:

```
> summary(a27ms$s1[2:1152])
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0100 0.2900 0.6600 0.9846 1.3800 8.6390
> summary(a27ms$s2[2:1152])
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.010  0.280  0.670  0.979  1.365  8.829
```

We see that both the Means are near the specified $\lambda = 1$.

The results of ADGoF tests for these two streams is shown below:

```
> ad.test( a27ms$s1[2:101], pexp, 1)
```

```
Anderson-Darling GoF Test
```

```
data: a27ms$s1[2:101] and pexp  
AD = 0.8908, p-value = 0.4197  
alternative hypothesis: NA
```

```
> ad.test( a27ms$s1[2:1001], pexp, 1)
```

```
Anderson-Darling GoF Test
```

```
data: a27ms$s1[2:1001] and pexp  
AD = 0.9284, p-value = 0.3971  
alternative hypothesis: NA
```

```
> ad.test( a27ms$s2[2:101], pexp, 1)
```

```
Anderson-Darling GoF Test
```

```
data: a27ms$s2[2:101] and pexp  
AD = 0.3597, p-value = 0.8873  
alternative hypothesis: NA
```

```
> ad.test( a27ms$s2[2:1001], pexp, 1)
```

```
Anderson-Darling GoF Test
```

```
data: a27ms$s2[2:1001] and pexp  
AD = 0.6913, p-value = 0.5661  
alternative hypothesis: NA
```

We see that both 100 and 1000 packet sets from two different streams (s1 and s2) all passed the AD ≤ 2.492 criterion.

6.4.2. Perfas Results

Section 11.4 of [RFC2330] suggests three possible measurement points to evaluate the Poisson distribution. The Perfas+ analysis uses "wire times for the packets as recorded using a packet filter". However, due to limited access at the Perfas+ side of the test setup, the captures were made after the Perfas+ streams traversed the production network, adding a small amount of unwanted delay variation to the wire times (and possibly error due to packet loss).

The statistical summary for two Perfas+ streams is below:


```
> summary(a27pe$p1)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.004   0.347   0.788   1.054   1.548   4.231
> summary(a27pe$p2)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.0010  0.2710  0.7080  0.9696  1.3740  7.1160
```

We see that both the Means are near the specified $\lambda = 1$.

The results of ADGoF tests for these two streams is shown below:

```
> ad.test(a27pe$p1, pexp, 1 )
```

Anderson-Darling GoF Test

```
data: a27pe$p1 and pexp
AD = 1.1364, p-value = 0.2930
alternative hypothesis: NA
```

```
> ad.test(a27pe$p2, pexp, 1 )
```

Anderson-Darling GoF Test

```
data: a27pe$p2 and pexp
AD = 0.5041, p-value = 0.7424
alternative hypothesis: NA
```

```
> ad.test(a27pe$p1[1:100], pexp, 1 )
```

Anderson-Darling GoF Test

```
data: a27pe$p1[1:100] and pexp
AD = 0.7202, p-value = 0.5419
alternative hypothesis: NA
```

```
> ad.test(a27pe$p1[101:193], pexp, 1 )
```

Anderson-Darling GoF Test

```
data: a27pe$p1[101:193] and pexp
AD = 1.4046, p-value = 0.201
alternative hypothesis: NA
```

```
> ad.test(a27pe$p2[1:100], pexp, 1 )
```

Anderson-Darling GoF Test

```
data: a27pe$p2[1:100] and pexp
```

```
AD = 0.4758, p-value = 0.7712
alternative hypothesis: NA
```

```
> ad.test(a27pe$p2[101:193], pexp, 1 )
```

```
Anderson-Darling GoF Test
```

```
data: a27pe$p2[101:193] and pexp
AD = 0.3381, p-value = 0.9068
alternative hypothesis: NA
```

```
>
```

We see that both 193, 100, and 93 packet sets from two different streams (p1 and p2) all passed the AD <= 2.492 criterion.

6.4.3. Conclusions for Goodness-of-Fit

Both NetProbe and Perfas+ implementations produce adequate Poisson distributions when according to the Anderson-Darling Goodness-of-Fit at the 5% significance (1-alpha = 0.05, or 95% confidence level).

6.5. Implementation of Statistics for One-way Delay

We check which statistics were implemented, and report on those facts, noting that Section 4 of [RFC2680] does not specify the calculations exactly, and gives only some illustrative examples.

	NetProbe	Perfas
4.1. Type-P-One-way-Packet-Loss-Average (this is more commonly referred to as loss ratio)	yes	yes

Implementation of Section 4 Statistics

We note that implementations refer to this metric as a loss ratio, and this is an area for likely revision of the text to make it more consistent with wide-spread usage.

7. Conclusions for RFC 2680bis

This memo concludes that [RFC2680] should be advanced on the standards track, and recommends the following edits to improve the text (which are not deemed significant enough to affect maturity).

- o Revise Type-P-One-way-Packet-Loss-Ave to Type-P-One-way-Delay-Packet-Loss-Ratio
- o Regarding implementation of the loss delay threshold (section 6.2), the assumption of post-processing is compliant, and the text of RFC 2680bis should be revised slightly to include this point.
- o The IETF has reached consensus on guidance for reporting metrics in [RFC6703], and this memo should be referenced in RFC2680bis to incorporate recent experience where appropriate.

We note that there are at least two Erratta on [RFC2680] and these should be processed as part of the editing process.

8. Security Considerations

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

9. IANA Considerations

This memo makes no requests of IANA, and the authors hope that IANA personnel will be able to use their valuable time in other worthwhile pursuits.

10. Acknowledgements

The authors thank Lars Eggert for his continued encouragement to advance the IPPM metrics during his tenure as AD Advisor.

Nicole Kowalski supplied the needed CPE router for the NetProbe side of the test set-up, and graciously managed her testing in spite of issues caused by dual-use of the router. Thanks Nicole!

The "NetProbe Team" also acknowledges many useful discussions on statistical interpretation with Ganga Maguluri.

11. References

11.1. Normative References

- [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.
- [RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, November 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.
- [RFC6576] Geib, R., Morton, A., Fardid, R., and A. Steinmitz, "IP Performance Metrics (IPPM) Standard Advancement Testing", BCP 176, RFC 6576, March 2012.
- [RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View", RFC 6703, August 2012.

- [RFC6808] Ciavattone, L., Geib, R., Morton, A., and M. Wieser, "Test Plan and Results Supporting Advancement of RFC 2679 on the Standards Track", RFC 6808, December 2012.

11.2. 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.
- [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.
- [Perfas] Heidemann, C., "Qualitaet in IP-Netzen Messverfahren", published by ITG Fachgruppe, 2nd meeting 5.2.3 (NGN) http://www.itg523.de/oeffentlich/01nov/Heidemann_QOS_Messverfahren.pdf , November 2001.
- [RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.
- [Radgof] Bellosta, C., "ADGofTest: Anderson-Darling Goodness-of-Fit Test. R package version 0.3.", <http://cran.r-project.org/web/packages/ADGofTest/index.html>, December 2011.
- [Radk] Scholz, F., "adk: Anderson-Darling K-Sample Test and Combinations of Such Tests. R package version 1.0.", , 2008.
- [Rtool] R Development Core Team, "R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org/>", , 2011.
- [WIPM] "AT&T Global IP Network", <http://ipnetwork.bgtmo.ip.att.net/pws/index.html>, 2012.

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 58 12747
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
Technical University Darmstadt
Darmstadt,
Germany

Phone:
Email: matthias_michael.wieser@stud.tu-darmstadt.de

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2013

K. Ko
ADTRAN
February 18, 2013

Model-Based Estimation of Streaming Performance
draft-ko-ippm-streaming-performance-00.txt

Abstract

This memo defines metrics plus a methodology for post-measurement processing of sample metrics to evaluate network path performance relative to streaming video traffic. The metrics are based on established methodologies for TCP throughput testing. The post-processing methodology is based on a model of streaming traffic that allows a given sample metric to be evaluated against multiple sets of parameters representing different streaming rates, delays and buffering values. The results of the post-processing methodology are derived metrics that are suitable for statistical analysis.

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on August 18, 2013.

Copyright Notice

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

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

Table of Contents

1. Introduction.....	3
2. Conventions used in this document.....	4
3. Transmission and Buffering of Streaming Data.....	4
4. Methodology.....	5
5. Streaming model.....	6
5.1. Model parameters.....	6
5.2. Model behavior.....	7
6. Metrics.....	9
6.1. A Singleton Definition for Short Term TCP Throughput.....	9
6.1.1. Metric Name.....	9
6.1.2. Metric Parameters.....	9
6.1.3. Metric Units.....	9
6.1.4. Definition.....	10
6.1.5. Discussion.....	10
6.2. A Definition of Sample for Short Term TCP Throughput.....	11
6.2.1. Metric Name.....	11
6.2.2. Metric Parameters.....	11
6.2.3. Metric Units.....	11
6.2.4. Definition.....	12
6.2.5. Discussion.....	12
6.3. A Definition of Sample for Dejitter Buffer Fill.....	12
6.3.1. Metric Name.....	13
6.3.2. Metric Parameters.....	13
6.3.3. Metric Units.....	13
6.3.4. Definition.....	14
6.3.5. Discussion.....	14
6.3.6. Methodologies.....	15
6.4. Some Statistics Definitions for Dejitter-Buffer-Fill.....	15
6.4.1. Initial-Streaming-Delay.....	15
6.4.2. Percentage-Viewing-Time.....	16

6.4.3. Minimum-Buffer-Depth.....	17
7. Additional topics.....	17
8. Security Considerations.....	17
9. IANA Considerations.....	17
10. References.....	17
10.1. Normative References.....	17
10.2. Informative References.....	18
11. Acknowledgments.....	18

1. Introduction

As the rates at which residential users access the Internet have increased over time, the types of traffic accessed by those users have evolved. Dialup access at rates of up to 56 kbps supported little more than email and non-real time traffic such as static web pages. The introduction of DSL and cable modems helped drive the trend towards more complex web pages with higher information content, as well as the emergence of real-time traffic such as VoIP and near-real time traffic such as streaming audio and video at low speeds. With the introduction of Fiber To The Premises (FTTP) as well as increasingly high DSL, cable and mobile wireless access rates, streaming video has become the largest category of consumer traffic on the Internet. According to one source [Cis2012], 49% of all consumer Internet traffic in 2011 was streaming video in some form and the percentage is continuing to increase.

The applications that display streaming video can be sensitive to variations in throughput and delay in the network. If the dejitter buffer in the destination host doesn't receive a steady enough stream of packets at the required rate it may underflow, causing a noticeable freeze in the video being played out. While application providers can mitigate the frequency and severity of these freezes, doing so typically involves trading off both the perceived quality of the video and the delay before it starts against the possibility of performance issues in mid-playout.

Recent large scale performance trials have included dedicated tests to measure video streaming performance [FCC2012]. These tests are designed to measure streaming at a defined performance point, with a new dedicated test required for each new performance point. This document proposes a different approach in which metrics resulting from TCP throughput performance testing are used to evaluate the network path's ability to support streaming at multiple performance points.

This document defines metrics plus a methodology for post-measurement processing of sample metrics to determine network path

performance relative to the requirements for streaming video traffic. The metrics are based on established methodologies for TCP throughput testing [RFC6349]. The post-processing methodology is based on a model of streaming traffic that allows a given sample metric to be evaluated against multiple sets of parameters representing different streaming rates, delays and buffering values. The results of the post-processing methodology are derived metrics that are suitable for statistical analysis.

2. Conventions used in this document

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

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

Whenever a technical term from the IPPM Framework document [RFC2330] is first used in this memo, it will be tagged with a trailing asterisk. For example, "term*" indicates that "term" is defined in the Framework.

3. Transmission and Buffering of Streaming Data

Streaming video is a near-real time application, the performance of which is dependent upon many factors including: the throughput and delay characteristics of the network; the rate, susceptibility to lost packets and other characteristics of the encoded content; and the protocols and application parameters used to deliver and display the content. In a typical implementation, packets are sent from a content serving host to a destination host where they are stored in a de jitter buffer. The buffered packets are read from the buffer, decoded, and displayed by a video display application as needed in real time. The de jitter buffer stores the packets from the time of arrival at the destination host until they are needed by the display application. This de jitter buffer compensates for variability in the delivery of packets across the network and also allows the content serving host to send packets on a schedule that can differ from the timing required by the display application. For example, content serving hosts commonly schedule transmission of streaming video packets in bursts separated by idle time such that the average rate of transmission approximates the rate required for display, but the momentary rate at a given point in time is either much higher than the display rate or it is zero.

The amount of data present in the dejitter buffer at a given point in time is a function of the difference between the packet arrival activity and the packet read activity. The display application typically waits until the dejitter buffer has stored some minimum amount of received data before starting to read it, to allow the momentary read rate to exceed the arrival rate without exhausting the buffer contents. As long as there are unread packets available in the dejitter buffer, the display application can read, decode and display the video content without interruption. However, if packet reads empty the buffer without the corresponding arrival of new packets, the resulting underflow will interrupt the display of video. When this happens, video freezes until the buffer has once again received and stored a minimum amount of data.

Nearly all video streaming traffic is carried over TCP. TCP is a connection oriented protocol which responds to packet loss by reducing its congestion window. As a result, the momentary throughput for the protocol varies, which can lead to the buffer exhaustion and video interruptions described above. In the methodology described below, the variation in throughput is measured directly during the course of TCP throughput testing and the measured results are applied to a model of streaming video behavior to estimate the ability of the measured network path to support streaming video at a given combination of parameters. The same measured results can be tested against multiple sets of streaming parameters, generating a picture of how well the measured performance would support streaming video at different rates and with different amounts of delay.

4. Methodology

This methodology is intended for operational IP networks. The metrics measured in the first step rely on the framework for TCP throughput testing defined in RFC6349.

The methodology is described in the following sequence of steps:

1. Using the methodology in RFC6349, perform TCP throughput testing on the network path under test. During the throughput testing, collect the sample metric Short-term-TCP-throughput-constant-stream defined in Section 6.2. This metric may be collected on its own in a TCP throughput test dedicated to streaming video performance, or it may be collected in addition to the metrics defined in RFC6349 as part of a larger series of performance tests.

2. Define the model parameter values described in Section 5.1. for the streaming model of Section 5.
3. Apply the streaming model to the sample metric Short-term-TCP-throughput-constant-stream using the model behavior defined in Section 5.2.
4. Step 3 may be performed as many times as desired defining different sets of model parameter values in step 2. Each iteration of the process generates results indicating how well the network path performance at the time of the throughput test would support streaming content under a different set of conditions.

5. Streaming model

In this section we define a generalized model for the reception, storage and subsequent reading of a media stream. The streaming model generates a sequence of values representing the amount of data stored in the dejitter buffer at discrete times during the simulated reception and display of streaming media content. The input to the model is the sample metric Short-term-TCP-throughput-constant-stream defined in Section 6.2. The output of the model is the derived sample metric Dejitter-Buffer-Fill defined in Section 6.3.

5.1. Model parameters

The following parameters are used in the streaming model:

- o The average encoded media rate *Ravg*, in bits per second. This is the average rate at which data is read from the dejitter buffer by the streaming media application for decoding and display when the model is in state *FILL_PLAY* or *MAINTAIN*. It is also the maximum rate at which data is written to the buffer in state *MAINTAIN*.
- o The initial streaming rate *Rinit*, in bits per second where *Rinit* > *Ravg*. This is the maximum rate at which data is written to the dejitter buffer when the model is in state *FILL_NOPLAY* or *FILL_PLAY*. *Rinit* may be set to an arbitrarily high value to allow the buffer to be filled as quickly as can be supported by the network.

- o The buffer depth Binit, in bytes, at which the streaming display application starts to read content out of the dejitter buffer for encoding and display. The combination of Rinit and Binit determines the minimum time delay between the beginning of streaming over the network and the beginning of content decoding and display at the destination.
- o The target buffer depth Btarget, in bytes where $B_{target} \geq B_{init}$. As long as the fill level of the dejitter buffer remains at or above this depth, the average streaming rate is reduced to the encoded rate Ravg and the buffer depth remains constant until the end of the simulation. If the buffer depth drops below this value due to a reduction in TCP throughput, the streaming rate can increase as high as Rinit until the buffer depth once again reaches Btarget.
- o The time interval I, in seconds. This is the interval I used to generate the value of the sample metric Short-term-TCP-throughput-constant-stream used as an input to the simulation.

5.2. Model behavior

The streaming model is stateful. When the model is used to simulate streaming performance using a value of sample Short-term-TCP-throughput-constant-stream, it iteratively updates a number of variables which are then used to update the model's state:

- o The fill rate F at which data is written to the buffer. The units for F are bytes per time interval I. When the model is in state FILL_NOPLAY or FILL_PLAY, the value of F for an interval k is calculated from the minimum of Rinit (normalized to bytes per interval I) or the short-term TCP throughput for the interval R(k). When the model is in state MAINTAIN, the value of F for an interval k is calculated from the minimum of Ravg (normalized to bytes per interval I) or R(k). Note that F may have a non-integer value.
- o The playout rate P at which data is read from the buffer. The units for P are bytes per time interval I. When the model is in state FILL_NOPLAY, the value of P is zero. When the model is in state FILL_PLAY or MAINTAIN, the value of P is calculated from the average encoded rate Ravg. Note that P may have a non-integer value.

- o The buffer depth B , in bytes. The value of B in each time interval is iteratively calculated from the value of B from the previous interval and the fill and playout rates for the current interval. Note that since F and P may have non-integer values, B may also have a non-integer value.

The following pseudocode specifies the model algorithm:

```
// Initialize the parameters for time  $T(0)$ 

 $B(0) = 0$  // Initial buffer state
 $T(0) = T(1) - I$  //  $T(0) = T_0$ 
 $F_{init} = R_{init} / 8 * I$  // Initial fill rate
 $F_{maint} = R_{avg} / 8 * I$  // Maintenance fill rate
 $P = R_{avg} / 8 * I$  // Playout rate
 $Model\_state = FILL\_NOPLAY$  // Initial state

// Run the simulation for each time interval  $T(k)$ 

For  $k = 1$  to  $k_{max}$ 
  Switch( $Model\_state$ )
    // Buffer filling, no playout
    Case  $FILL\_NOPLAY$ :
       $B(k) = B(k-1) + \min(F_{init}, R(k))$ 
      If  $B(k) \geq B_{target}$  then
         $Model\_state = MAINTAIN$ 
      Else If  $B(k) \geq B_{init}$  then
         $Model\_state = FILL\_PLAY$ 
      End if
    // Buffer filling, media playing out
    Case  $FILL\_PLAY$ :
       $B(k) = B(k-1) + \min(F_{init}, R(k)) - P$ 
      If  $B(k) \geq B_{target}$  then
         $Model\_state = MAINTAIN$ 
      Else If  $B(k) \leq 0$  then
         $B(k) = 0$ 
         $Model\_state = FILL\_NOPLAY$ 
      End if
    // Buffer at target, media playing out
    Case  $MAINTAIN$ :
       $B(k) = B(k-1) + \min(F_{maint}, R(k)) - P$ 
      If  $B(k) \leq 0$  then
         $B(k) = 0$ 
         $Model\_state = FILL\_NOPLAY$ 
      Else If  $B(k) < B_{target}$  then
         $Model\_state = FILL\_PLAY$ 
      End if
```

End switch
Next k

6. Metrics

The structure of this section is as follows:

- o A 'singleton' analytic metric, called Short-term-TCP-throughput, will be introduced to measure a single observation of short-term TCP throughput.
- o Using this singleton metric, a 'sample', called Short-term-TCP-throughput-constant-stream, will be introduced to measure a sequence of singleton short-term TCP throughputs measured at regular intervals.
- o Using this sample and the streaming model defined in Section 5. , a derived sample metric called Dejitter-Buffer-Fill will be defined and discussed.
- o Using this derived sample metric, several 'statistics' will be defined and discussed.

This progression from singleton to sample to derived sample to statistics, with clear separation among them, is important.

6.1. A Singleton Definition for Short Term TCP Throughput

6.1.1. Metric Name

Short-term-TCP-throughput

6.1.2. Metric Parameters

- o Src, the IP address of a host
- o Dst, the IP address of a host
- o T, a time
- o I, a time interval

6.1.3. Metric Units

The value of Short-term-TCP-throughput is an integer number.

6.1.4. Definition

>>The *Short-term-TCP-throughput* from Src to Dst at time T over the interval I is R<< means that the number of new bytes sent over TCP from Src which are acknowledged by Dst during the time interval I preceding wire-time* T is R.

6.1.5. Discussion

Short-term-TCP-throughput is measured using the framework for TCP testing described in RFC6349. Like the metrics in that document, this metric should be measured in the TCP Equilibrium state [see RFC6349 Section 1.3].

Short-term-TCP-throughput can be measured at either the Src or Dst host. If measured at Dst, the first step in generating the metric is to record the highest Acknowledgement Numbers generated by Dst for the TCP connection being measured at the beginning and end of the time interval I preceding time T.

- o Assign $A(k)$ = the highest Acknowledgement Number generated by Dst at time T
- o Assign $A(k-1)$ = the highest Acknowledgement Number generated by Dst at time $(T - I)$

If the metric is measured at Src, then

- o Assign $A(k)$ = the highest Acknowledgement Number received by Src at time T
- o Assign $A(k-1)$ = the highest Acknowledgement Number received by Src at time $(T - I)$

Short-term-TCP-throughput is then calculated in bytes per interval I as

$$R = A(k) - A(k-1)$$

Calculation of the difference $A(k) - A(k-1)$ must account for the TCP window scaling option and for the potential rollover of the Acknowledgement Number from $(2^{32} - 1)$ to 0 between the beginning and the end of a measurement interval.

The streaming video applications described in Section 3. typically allow at least several seconds worth of content to be buffered before beginning to stream data, and then continue to send and store

data faster than it is being read for display until a target level of buffer fill is reached that may support several tens of seconds worth of content. To effectively apply the streaming model from Section 5. with an acceptable level of accuracy, we require short term throughput values at intervals on the order of two orders of magnitude smaller than the buffering intervals described above. A measurement interval I on the order of 100 milliseconds is suggested.

TCP throughput testing using the methodology described in RFC6349 can make use of multiple TCP connections operating concurrently between the same Src and Dst. If this is the case, the process described above should be used to calculate the short term throughput at time T for each TCP connection separately, and then the values should be summed to generate Short-term-TCP-throughput.

6.2. A Definition of Sample for Short Term TCP Throughput

Given the singleton metric Short-term-TCP-throughput, 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 I , and then define a sample of values of parameter T . The means for defining the values of T is to select a beginning time T_0 , a final time T_f , and an interval I , then define a series of regularly spaced time values T whose values fall between T_0 and T_f . The time interval between successive values of T will have the constant value I .

6.2.1. Metric Name

Short-term-TCP-throughput-constant-stream

6.2.2. Metric Parameters

- o Src, the IP address of a host
- o Dst, the IP address of a host
- o T_0 , a time
- o T_f , a time
- o I , a time interval

6.2.3. Metric Units

A sequence of pairs; the elements of each pair are:

- o T, a time, and
- o R, an integer number

The values of T in the sequence are monotonic and increasing at constant intervals I. Note that T and I would be valid parameters to Short-term-TCP-throughput, and that R would be a valid value of Short-term-TCP-throughput. Note also that since T can be determined from T0, I and the position of the value in the sequence, it is possible to alternatively define the metric units as: a time T0, a time interval I; and a sequence of integer numbers R.

6.2.4. Definition

Given T0, Tf, and I, we compute a series of time values T(k) where

$$T(k) = T0 + (k) * I, \text{ for } 1 \leq k \leq (Tf - T0) / I.$$

The first time value in the sequence occurs at $T(1) = T0 + I$, and successive time values are regularly spaced with the constant interval I between successive values of T. The last time value in the sequence occurs in the interval $Tf - I < T(k_{\max}) \leq Tf$. At each of the times in this sequence, we obtain the value of Short-term-TCP-throughput at this time using interval I. The value of the sample is made up of the resulting <time, rate> pairs. If there are no such pairs, the sequence is of length zero and the sample is said to be empty.

6.2.5. Discussion

The <time, rate> pairs that make up the value of Short-term-TCP-throughput-constant-stream provide a continuous view of the momentary throughput in the network path from Src to Dst from time T0 to time Tf. By defining a single value of I as a common input parameter to both Short-term-TCP-throughput and Short-term-TCP-throughput-constant-stream, the 'final' highest Acknowledgement Number A(k) used to calculate the kth singleton value in the sequence becomes the 'initial' highest value A(j-1) used to calculate the next successive (jth, where $j = k+1$) value in the sequence. In this way, each new segment acknowledged from time T0 until the end of the final interval preceding time Tf contributes to exactly one of the singleton values in the sample.

6.3. A Definition of Sample for Dejitter Buffer Fill

Given the sample metric Short-term-TCP-throughput-constant-stream, we now define one additional sample derived from such sample. The

idea of the derived sample is to select a particular binding of the model parameters `Ravg`, `Rinit`, `Binit`, `Btarget` and `I` defined in Section 5.1. , and then to apply the streaming model defined in Section 5. to the value of the sample `Short-term-TCP-throughput-constant-stream`. The buffer fill depths per time interval generated by the streaming model then form the value for the derived sample metric.

6.3.1. Metric Name

`Dejitter-Buffer-Fill`

6.3.2. Metric Parameters

- o `Rseq`, a value of type `Short-term-TCP-throughput-constant-stream`

The following model parameters are defined in Section 5.1. :

- o `Rinit`, a real number
- o `Ravg`, a real number
- o `Binit`, a real number
- o `Btarget`, a real number
- o `I`, a time interval

6.3.3. Metric Units

A sequence of pairs; the elements of each pair are:

- o `T`, a time, and
- o `B`, a real number

The values of `T` in the sequence are monotonic and increasing at constant intervals `I`. Note that if the input parameter `Rseq` is defined using parameters `T0` and `I` and a sequence of integer numbers `R`, `Dejitter-Buffer-Fill` may likewise be defined as: a time `T0`; a time interval `I`; and a set of real numbers `B`. Note also that the number of pairs in a value of `Dejitter-Buffer-Fill` will be one more than the number of pairs in the input parameter `Rseq`.

6.3.4. Definition

Given R_{seq} and the model parameters from Section 5.1. , we compute a series of buffer fill values $B(k)$ at regular time intervals $T(k)$ using the model algorithm defined in Section 5.2. The value of the sample is made up of the resulting $\langle T(k), B(k) \rangle$ pairs.

The first time value in the sequence occurs at $T(0) = T_0$, and successive time values are regularly spaced with the constant interval I between successive values of T . The last time value in the sequence occurs in the interval $T_f - I < T(k_{max}) \leq T_f$. The number of pairs in a value of the sample metric is one greater than the number of pairs in the input sample R_{seq} . If R_{seq} is an empty sample, then the value of Dejitter-Buffer-Fill is defined to be of length zero and the sample is said to be empty.

6.3.5. Discussion

The $\langle T(k), B(k) \rangle$ pairs that make up the value of Dejitter-Buffer-Fill approximate the variation in fill depth over time of a dejitter buffer for a streaming media application. As new data arrives from the network, it is written to the buffer and the fill depth increases. As data is read from the buffer to be decoded and displayed, the fill depth decreases. In any given time interval, any combination of write and read operations may occur, and both writes and reads may occur multiple times.

When a streaming session is initiated, data is sent from source to destination at a high rate (R_{init}) to fill the buffer as quickly as possible. For the same reason, read operations are disabled during this initiation period. Once the buffer has reached an initial fill level (B_{init}), read operations are enabled and the streaming media application begins to read, decode and display the streamed content. The data may continue streaming at a high rate until the buffer reaches a target fill level (B_{target}).

Once the buffer has reached its target fill level, the rate at which data is streamed is reduced to a value equal to the rate at which data is being read from the dejitter buffer. So long as the short-term throughput in the network supports this streaming rate, the dejitter buffer is in equilibrium, with the rates at which data is being written and read at the same average value. If the short-term throughput falls below the streaming rate, the buffer fill depth will fall below the target value. Once this occurs, the maximum streaming rate is increased to R_{init} until the buffer fill depth reaches B_{target} once more.

(Note that the increase in the maximum streaming rate does not imply either the presence or the absence of an explicit feedback mechanism between the buffer fill depth and the source of the streaming traffic. In some cases there may be such a mechanism, for example by varying the value of the advertised window in TCP. In other cases, allowing the rate at which data is written to the buffer to increase beyond R_{avg} is simply an acknowledgement that the remote source has continued to stream the data at the average rate but it has not all been successfully received, hence there is extra data in the network pipeline that may be received at higher than the average rate until the shortfall is made up.)

If the short-term throughput falls and remains below the streaming rate for some time, the buffer fill depth may drop to zero. At this point the buffer is exhausted, an underflow condition occurs, and read operations are disabled until the buffer once again reaches the initial fill level. Of course if this occurs, it signals an error condition in which the decoding and display of the streaming media is interrupted.

6.3.6. Methodologies

A single value R_{seq} of Short-term-TCP-throughput-constant-stream can be used with different model parameters to generate many values of Dejitter-Buffer-Fill. Each resulting value of Dejitter-Buffer-Fill represents the response of the jitter buffer to streaming content with different characteristics and/or displayed with different parameters over the same network conditions as recorded in R_{seq} .

6.4. Some Statistics Definitions for Dejitter-Buffer-Fill

Given the sample metric Dejitter-Buffer-Fill, we now offer several statistics of that sample. These statistics are offered mostly to be illustrative of what could be done.

6.4.1. Initial-Streaming-Delay

Given a Dejitter-Buffer-Fill and the model parameters used to derive it, the time between the time of the first pair T_0 and the first time at which the buffer fill depth $B(k)$ is equal to or greater than B_{init} . This is the delay between the initiation of streaming and the time at which the streaming media application first starts to decode and display the streamed content.

6.4.2. Percentage-Viewing-Time

Given a Dejitter-Buffer-Fill and the model parameters used to derive it, the total time during which data is being read from the dejitter buffer divided by the total time after the initial transition from state FILL_NOPLAY to another state. This represents the percentage of time spent viewing media content once the display starts, as opposed to observing an interruption in the content and waiting for it to resume. The metric can be calculated from Dejitter-Buffer-Fill using the algorithm specified by the following pseudo code:

```
// Initialization

Total_time = 0           // Total time after start of playout
View_time = 0            // Viewing time after start of playout
Model_state = INITIAL_FILL // Initial state

// Run iteratively for each time interval T(k)

For k = 1 to k_max
  Switch(Model_state)
    // initial buffer fill not counted toward total
    Case INITIAL_FILL:
      If B(k) >= Binit then
        Model_state = PLAYING
      End if
    // Buffer filling, no playout
    Case FILL_NOPLAY:
      Total_time = Total_time + I
      If B(k) >= Binit then
        Model_state = PLAYING
      End if
    // media playing out
    Case PLAYING:
      Total_time = Total_time + I
      View_time = View_time + I
      If B(k) = 0 then
        Model_state = FILL_NOPLAY
      End if
  End switch
Next k

If Total_time > 0
  Percentage-Viewing-Time = View_time / Total_time
Else
  Percentage-Viewing-Time = 0
End if
```

6.4.3. Minimum-Buffer-Depth

Given a Dejitter-Buffer-Fill and the model parameters used to derive it, the minimum dejitter buffer depth at any point in time after Btarget is initially reached. A minimum value close to zero indicates that the buffer was close to exhaustion at some point in the simulation.

7. Additional topics

The following topics are candidates for inclusion but have not yet been addressed in this Internet-Draft.

- o Discussion of the different ways in which streams are scheduled and why the streaming model approximates this scheduling behavior.
- o Discussion of the differences between model behavior and network streaming behavior. Limitations of the model.
- o Discussion of short-term streaming rates that vary around the average.
- o Extension to adaptive streaming. At least, discussion of how the model can be extended to adaptive streaming.
- o Addition of explanatory figures.

8. Security Considerations

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

9. IANA Considerations

This memo makes no requests of IANA

10. References

10.1. Normative References

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

- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.
- [RFC6349] Constantine, B., Forget, G., Geib, R., and Schrage, R., "Framework for TCP Throughput Testing," RFC 6349, August 2011.

10.2. Informative References

- [Cis2012] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2011-2016," May 30, 2012.
- [FCC2012] FCC, "Measuring Broadband America: Technical Appendix," July 2012.

11. Acknowledgments

The authors wish to thank the following people for reading and commenting on this draft <TBD>.

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

Authors' Addresses

Ken Ko
ADTRAN
901 Explorer Blvd.
Huntsville, AL 35806
USA

Email: ken.ko@adtran.com

IP Performance Working Group
Internet-Draft
Intended status: Experimental
Expires: April 18, 2013

M. Mathis
Google, Inc
Oct 15, 2012

Model Based Internet Performance Metrics
draft-mathis-ippm-model-based-metrics-00.txt

Abstract

We introduce a new class of model based Internet metrics designed to determine if a long path can be expected to meet a predefined end-to-end application performance target by applying a suite of single property tests to successive sections of the long path. In many cases these single property tests are based on existing IPPM metrics, with the addition of success and validity criteria. The sub-path at a time tests are designed to eliminate all known conditions that might prevent the full path from meeting the target performance.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <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 18, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Background	4
3. Common Models and Parameters	5
3.1. End-to-end parameters	5
3.2. Per sub-path parameters	7
3.3. Common Calculations for Single Property Tests	7
3.4. Parameter Derating	8
3.5. Single Property Tests Results	9
4. Single Property Tests	9
4.1. Verify the absence of cross traffic	9
4.1.1. Parameter Calculation	10
4.1.2. Cross traffic Measurement	10
4.2. Full Data Rate Loss Rate Tests	10
4.2.1. Loss Rate Measurement	11
4.3. Background Loss Rate Tests	11
4.3.1. Background Loss Rate Measurement	11
4.4. Queue Capacity Test	12
4.4.1. Model Calculation	12
4.4.2. Queue Capacity Measurement	12
4.5. AQM Test	12
4.5.1. Model Calculation	12
4.5.2. AQM Measurement	12
4.6. Reordering Test	12
4.6.1. Model Calculation	12
4.6.2. Reordering Measurement	12
5. Calibration	12
6. References	13
6.1. Normative References	13
6.2. Informative References	13
Appendix A. Model Derivations	13
Appendix B. Old text from an earlier document	13
Author's Address	15

1. Introduction

We introduce a new class of model based metrics designed to determine if a long path can be expected to meet a predefined application end-to-end performance target by applying a suite of single property tests to successive sections of the long path. In many cases these single property tests are based on existing IPPM metrics, with the addition of success and validity criteria. The sub-path at a time tests are designed to eliminate all known conditions that might prevent the full path from meeting the target performance. The end-to-end target performance must be specified in advance, in order to be able to open-loop the control systems (such as congestion control) that are present in all Internet transport protocols and applications. Since a singleton (see [RFC2330]) is only a pass/fail measurement of a sub-path, these metrics are most useful in composition over large pools of samples, such as across a collection of paths or a time interval [RFC5835].

For Bulk Transport Capacity (BTC) the target performance to be measured is a data rate. TCP's ability to compensate for less than ideal network conditions is fundamentally affected by the RTT and MTU of the end-to-end Internet path that it traverses. Since the minimum RTT and maximum MTU are both fixed properties of the path, they are also taken as parameters. The target values for these three parameters, Data Rate, RTT and MTU, are determined by the application, its intended use and the physical infrastructure over which it traverses. They are described in more detail in Section 3 together with the models used to infer the required performance of the underlying Internet fabric.

Traditional end-to-end BTC metrics have proven to be difficult or unsatisfactory for the reasons described in Section 2. Rather than testing the end-to-end path with TCP or other some other BTC, each sub-path is evaluated using suite of far simpler and more predictable single property tests described in Section 4. For BTC the following tests are sufficient: raw data rate, background loss rate, queue burst capacity, reordering extent, onset of congestion/AQM and return path quality. If every sub-path passes all of these tests, then an end-to-end application using any reasonably modern TCP or similar protocol should be able to attain the specified target data rate, over the full end-to-end path at the specified RTT and MTU.

There exists the potential that model based metric might fail, in the sense that every sub-path of an end-to-end path passes every single property test and yet a application might still fail to attain its performance target. If so, then a traditional BTC needs to be used to validate the tests for each sub-path, as described in Section 5.

Future text (or a more likely a future document) will describe model based metrics for real time traffic. The salient point will be that concurrently meeting the goals of both RT and throughput maximizing traffic implicitly requires some form of traffic segregation, such that the two traffic classes are not placed in the same queue. Some technique as simple as SFQ[SFQ] might be a sufficient alternative to full QoS.

TODO:

Add discussion of protocol overhead: MSS vs IP MTU vs link MTU

2. Background

(Fragments of earlier text)

The holy grail of IPPM has been BTC measurement, but it has proven to be a very hard problem for a number of reasons:

TCP is a control systems - everything affects performance, including components that are explicitly not part of the the test. Congestion control is an equilibrium process, transport protocols change the network (raise loss probability and/or RTT) to conform to their behavior.

TCP's ability to compensate for network flaws is directly proportional to the number of round trips per second (e.g. inversely proportional to the RTT). As a consequence a flawed link that passes a local test is likely to completely fail when the path is extended by a perfect network to some larger RTT. TCP has a meta Heisenberg problem - Measurement and cross traffic interact in unknown and ill defined ways. The situation is actually worse than the traditional physics problem where you can at least estimate the relative masses of the measurement and measured particles. For network measurement you can not in general determine the relative "masses" of the measurement traffic and cross traffic, so you can not even gage the relative magnitude of their effects on each other.

The new approach is to "open loop" congestion control. Defeat CC, typically by throttling TCP to a lower rate, such that it does not respond to network conditions. In this mode the measurement software explicitly controls TCP's state variables (e.g. cwnd) to create controlled traffic patterns, which are manipulated to measure the network.

Models are used to determine the actual test parameters (loss rate, etc) from the target parameters. The basic method is to use models to estimate simple network properties required to sustain a given transport flow (or set of flows), and using a suite of simpler

metrics to confirm that the network meets the required properties. For example a network can sustain a Bulk TCP flow of a given data rate, MTU and RTT when 4 (and probably more) conditions are met:

- The raw link rate is higher than the target data rate.

- The raw packet loss rate is lower than required by a suitable TCP performance model

- There is sufficient buffering at any bottleneck smooth bursts.

- When the link is overfilled (congested), the onset of packet loss is progressive.

These condition can all be verified with simple tests, using model parameters and acceptance thresholds derived from the target data rate, MTU and RTT. Note that this procedure is not invertible: a singleton measurement is a pass/fail evaluation of a given path or subpath at a given performance. Measurements to confirm that a link passes at one particular performance may not be generally be useful to predict if the link will pass at a different performance.

Although they are not invertible, they do have several other valuable properties, such as natural ways to define several different composition metrics.

3. Common Models and Parameters

Transport performance models are used to derive the test parameters for each single property test from the end-to-end target parameters and additional ancillary parameters.

It is envisioned that the modeling phase (to compute the test parameters) and testing phases will be decoupled. This section covers common derived parameters, used by multiple single property tests. For some tests, additional modeling is described with the tests.

Since some aspects of the models are very conservative, the modeling framework permits some latitude in derating test parameters, as described in Section 3.4.

For certain sub-paths (e.g. common types of access links) it would be appropriate for the single property test parameters to be documented as a "measurement profile" together with the modeling assumptions and derating factors described in Section 3.3 and Section 3.4.

3.1. End-to-end parameters

These parameters are determined by the needs of the application or the ultimate end user and the end-to-end Internet path.

Target Data Rate: The application or ultimate user's performance goal (in aggregate across all connections).

Permitted Number of Connections: The target rate can be more easily obtained by dividing the traffic across more than one connection. In general the number of concurrent connections is determined by the application, however see the comments below on multiple connections.

Target RTT (Round Trip Time): For fundamental reasons a long path makes it more difficult for TCP or other transport protocol to meet the target rate. The target RTT must be representative for the actual applications expected to use the network. This parameter may be subject to a future convention (e.g. continental scale paths should be assumed to be some fixed RTT, such as 100 ms) or alternatively be an property of an ISP's topology (e.g. a ISP with richer and better placed peering may actually have lower RTTs for typical users.)

Target MTU (Maximum Transmission Unit): Assume 1500 Bytes unless otherwise specified. If some sub-path forces a smaller MTU, then all sub-paths must be tested with the same smaller MTU.

The use of multiple connections has been very controversial since the beginning of the World-Wide-Web[first complaint]. Modern browsers open many connections [BScope]. Experts in the IETF transport area have frequently spoken against this practice [long list]. It is not inappropriate to assume some small number of concurrent connections (e.g. 4 or 6), to compensate for limitation in TCP. However, choosing too large a number is at risk of being taken as a signal by the web browser community that this practice has been embraced by the Internet community. It may not be desirable to send such a signal.

The following optional parameters apply for testing generalized end-to-end paths that include subpaths with known specific types of behaviors that are not well represented by simple queueing models:

Bottleneck link clock rate: This applies to links that are using virtual queues or other techniques to police or shape users traffic at lower rates full link rate. The bottleneck link clock rate should be representative of queue drain times for short bursts of packets on an otherwise unloaded link.

Channel hold time: For channels that have relatively expensive channel arbitration algorithms, this is the typical (maximum?) time that data and or ACKs are held pending acquiring the channel. While under heavy load, the RTT may be inflated by this parameter, unless it is built into the target RTT

Preload traffic volume: If the user's traffic is shaped on the basis of average traffic volume, this is volume necessary to invoke "heavy hitter" policies.

Unloaded traffic volume: If the user's traffic is shaped on the basis of average traffic volume, this is the maximum traffic volume that a test can use and stay within a "light user" policies.

Note on a ConEx enabled network [ConEx], the word "traffic" in the last two items should be replaced by "congestion" i.e. "preload congestion volume" and "unloaded congestion volume".

3.2. Per sub-path parameters

Some single parameter tests also need parameter of the sub-path.

sub-path RTT: RTT of the sub-path under test.

sub-path link clock rate: If different than the Bottleneck link clock rate

3.3. Common Calculations for Single Property Tests

The most important derived parameter is `target_pipe_size` (in packets), which is the number of packets needed exactly meet the target rate, with no cross traffic for the specified RTT and MTU. It is given by:

$$\text{target_pipe_size} = \text{target_rate} * \text{target_RTT} / \text{target_MTU}$$

If the transport protocol (e.g. TCP) average window size is smaller than this, the link will be under filled.

If `target_data_rate` is equal to `bottleneck link_data_rate`, then `target_pipe_size` also predicts the onset of queueing. If the transport protocol (e.g. TCP) average window size is larger than the `target_pipe_size`, the excess packets will be in a standing queue at the bottleneck.

If the transport protocol is using Reno congestion control [RFC5681], then there must be `target_pipe_size` roundtrips between losses. Otherwise the multiplicative window reduction triggered by a loss would cause the network to be underfilled. Following [MSM097], we derive the losses must be no more frequent than every 1 in $(3/2)(\text{target_pipe_size}^2)$ packets. This provides the reference value for `target_run_length` which is typically the number of packets that must be delivered between loss episodes in the tests below:

$$\text{reference_target_run_length} = (3/2)(\text{target_pipe_size}^2)$$

Note that this calculation is based on a number of assumptions that may not apply. Appendix A discusses these assumptions and provides

some alternative models. The actual method for computing `target_run_length` MUST be published along with the rationale for the underlying assumptions and the ratio of chosen `target_run_length` to `reference_target_run_length`.

Although this document gives a lot of latitude for calculating `target_run_length` people specifying profiles for suites of single property tests need to consider the effect of their choices on the ongoing conversation and tussle about the relevance of "TCP friendliness" as an appropriate model for capacity allocation. Choosing a `target_run_length` that is substantially smaller than `reference_target_run_length` is equivalent to saying that it is appropriate for the research community to abandon "TCP friendliness" as a fairness model and to develop more aggressive Internet transport protocols, and for applications to continue (or even increase) the number of connections that they open.

The calculations for individual parameters are presented with the each single property test. In general these calculations are permitted some as described in Section 3.4

3.4. Parameter Derating

Since some aspects of the models are very conservative, the modeling framework permits some latitude in derating some specific test parameters, as indicated in Section 4. For example classical performance models suggest that in order to be sure that a single TCP stream can fill a link, it needs to have a full bandwidth-delay-product worth of buffering at the bottleneck[`QueueSize`]. In real networks with real applications this is sometimes overly conservative. Rather than trying to formalize more complicated models we permit some test parameters to be relaxed as long as they meet some additional procedural constraints:

- The method used compute and justify the derated metrics is published in such a way that it becomes a matter of public record.
- The calibration procedures described in Section 5 are used to demonstrate the feasibility of meeting the performance targets with the derated test parameters.

- The calibration process itself is documented in such a way that other researchers can duplicate the experiments and validate the results.

Note that some single property test parameters are not permitted to be derated.

3.5. Single Property Tests Results

TBD Define: Pass, Fail and inconclusive test results.

The inconclusive outcome is needed to address the case where a test failed to attain the specified test conditions. This is important to the extent that the tests themselves have built in control systems which might interfere with some aspect of the test. It is required for example to use TCP for testing.

4. Single Property Tests

The single property tests confirm that each sub-path can sustain the normal traffic patterns caused by TCP running at the specified target performance. Specifically they confirm that each sub-path has: sufficient raw capacity (e.g. sufficient data rate); low enough background loss rate where mandatory congestion control stays out of the way; large enough queue space to absorb TCP's normal bursts; does not cause unreasonable packet reordering; progressive AQM to appropriately invoke congestion control. Appropriately invoking congestion control requires that packet losses or ECN marks start progressively before TCP creates an excessive sustained queues[BufferBloat] or excessively bursty losses. The return path must also subject to a similar suite of tests, although potentially with different test parameters.

Note that many of the sub-path tests resemble metrics that have already been defined in the IPPM context, with the addition of criteria for passing or failing the test. The models used to derive the test parameters make specific assumptions about network conditions, a test is deemed "inconclusive" (as opposed to failing) if tester does not meet the underlying assumption. For example a loss rate test at a specified data rate is inconclusive if the tester fails to send data at the specified rate for some reason. This concept of an inconclusive test is necessary to build tests out of protocols or technologies that they themselves have built in or implicit control systems.

Some single property test can be combined, since their parameters are not mutually exclusive.

4.1. Verify the absence of cross traffic

Use a passive packet or SNMP monitoring to verify that the traffic volume on the sub-path agrees with the traffic generated by each test. Ideally this should be performed before during and after each test.

The goal is provide quality assurance on the overall measurement process, and specifically to detect the following measurement failure: a user observes unexpectedly poor application performance, the ISP observes that the access link is running at the rated capacity. Both fail to observe that the user's computer has been infected by a virus which is spewing traffic as fast as it can.

Parameters:

Maximum Cross Traffic Data Rate The amount of excess traffic permitted. Note that this might be different for different tests.
Maximum Data Rate underage The permitted amount that the traffic can be less than predicted for the current test. Normally this would just be a statement of the maximum permitted measurement error, however it might also detect cases where the passive and active tests are misaligned: testing different subscriber lines. This is important because the vantage points are so different: in-band active measurement vs out-of-band passive measurement.

4.1.1. Parameter Calculation

TBA

4.1.2. Cross traffic Measurement

One possible method is an adaptation of: [www-didc.lbl.gov/papers/SCNM-PAM03.pdf](http://www.didc.lbl.gov/papers/SCNM-PAM03.pdf) D Agarwal etal. "An Infrastructure for Passive Network Monitoring of Application Data Streams". Use the same technique as that paper to trigger the capture of SNMP statistics for the link.

4.2. Full Data Rate Loss Rate Tests

We propose two versions of the loss rate test. One, performed at data full rate, is intrusive and recommend for infrequent testing, such as when a service is first turned up or as part of an auditing process. Note that this test also implicitly confirms that sub_path has sufficient capacity to carry the target_data_rate.

The second background loss rate, described below, is designed for ongoing monitoring for change in sub-path quality.

Parameters:

Run Length Same as target_run_lenght
Data Rate Same as target_data_rate

Note that these parameters MUST NOT be derated. If the default parameters are too stringent use an alternate model for target_data_rate as described in Appendix A.

4.2.1. Loss Rate Measurement

Data is sent at the specified `data_rate`. The receiver accumulates the total data delivered and packets lost [and ECN marks, which are nominally treated as losses by conforming transport protocols]. The observed `average_run_lenght` is computed from `total_data_delivered` divided by the `total_loss_rate`. A [TBD] statistical test is applied to determine when or if the `average_run_lenght` is larger than `target_run_lenght`.

TODO: add language about monitoring cross traffic.

The test is deemed to have passed only if the observed data rate matches the `target_data_rate` and it is statistically significant that the `average_run_lenght` is larger than `target_run_lenght`. It is deemed inconclusive if: the statistical test is inconclusive; there is too much background load; or the `target_data_rate` could not be attained.

4.3. Background Loss Rate Tests

The background loss rate is designed for ongoing monitoring for change in sub-path quality. It should be used in conjunction with the above full rate test.

Parameters:

Run Length Same as `target_run_lenght`

Data Rate Some small fraction of `target_data_rate`, such as 1%.

4.3.1. Background Loss Rate Measurement

The receiver accumulates the total data delivered and packets losses [and ECN marks, which are nominally treated as losses by conforming transport protocols]. The observed `average_run_lenght` is computed from `total_data_delivered` divided by the `total_loss_rate`. A [TBD] statistical test is applied to determine when or if the `average_run_lenght` is larger than `target_run_lenght`.

TODO: add language about monitoring cross traffic.

The test is deemed to have passed if it is statistically significant that the `average_run_lenght` is larger than `target_run_lenght`. It is deemed inconclusive if there is too much background traffic or the statistical test is inconclusive.

4.4. Queue Capacity Test

Parameters:

TBA TBA

4.4.1. Model Calculation

TBA

4.4.2. Queue Capacity Measurement

TBA

4.5. AQM Test

Parameters:

TBA TBA

4.5.1. Model Calculation

TBA

4.5.2. AQM Measurement

TBA

4.6. Reordering Test

Parameters:

TBA TBA

4.6.1. Model Calculation

TBA

4.6.2. Reordering Measurement

TBA

5. Calibration

If using derated metrics, or when something goes wrong, the results must be calibrated against a traditional BTC.....

6. References

6.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

6.2. Informative References

- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
 - [RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.
 - [RFC5835] Morton, A. and S. Van den Berghe, "Framework for Metric Composition", RFC 5835, April 2010.
 - [MSMO97] Mathis, M., Semke, J., Mahdavi, J., and T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", Computer Communications Review volume 27, number3, July 1997.
 - [BScope] Browserscope, "Browserscope Network tests", Sept 2012, <<http://www.browserscope.org/?category=network>>.
- See Max Connections column

Appendix A. Model Derivations

This appendix describes several different ways to calculate `target_run_length` and the implication of the chosen calculation.

Rederive MSMO97 under two different assumptions: `target_rate = link_rate` and `target_rate < 2 * link_rate`.

Show equivalent derivation for CUBIC.

Commentary on the consequence of the choice.

Appendix B. Old text from an earlier document

To be moved, removed or absorbed

Step 0: select target end-to-end parameters: a target rate and target RTT. The primary test will be to confirm that the link quality is sufficient to meet the specified target rate for the link under test, when extended to the target RTT by an ideal network. The target rate must be below the actual link rate and nominally the target RTT would be longer than the link RTT. There should probably be a convention for the relationship between link and target rates (e.g. 85%).

For example on a 10 Mb/s link, the target rate might be 1 MBytes/s, at an RTT of 100 mS (a typical continental scale path).

Step 1: On the basis of the target rate and RTT and your favorite TCP performance model, compute the "required run length", which is the required number of consecutive non-losses between loss episodes. The run length resembles one over the loss probability, if clustered losses only count as a single event. Also select "test duration" and "test rate". The latter would nominally be the same as the target rate, but might be different in some situations. There must be documentation connecting the test rate, duration and required run length, to the target rate and RTT selected in step 0.

Continuing the above example: Assuming a 1500 Byte MTU. The calculated model loss rate for a single TCP stream is about 0.01% (1 loss in 1E4 packets).

Step 2, the actual measurement proceeds as follows: Start an unconstrained bulk data flow using any modern TCP (with large buffers and/or autotuning). During the first interval (no rate limits) observe the slowstart (e.g. tcpdump) and measure: Peak burst size; link clock rate (delivery rate for each round); peak data rate for the fastest single RTT interval; fraction of segments lost at the end of slow start. After the flow has fully recovered from the slowstart (details not important) throttle the flow down to the test rate (by clamping cwnd or application pacing at the sender or receiver). While clamped to the test rate, observe the losses (run length) for the chosen test duration. The link passes the test if the slowstart ends with less than approximately 50% losses and no timeouts, the peak rate is at least the target rate, and the measured run length is better than the required run length. There will also need to be some ancillary metrics, for example to discard tests where the receiver closes the window, invalidating the slowstart test. [This needs to be separated into multiple subtests]

Optional step 3: In some cases it might make sense to compute an "extrapolated rate", which is the minimum of the observed peak rate, and the rate computed from the specified target RTT and the observed run length by using a suitable TCP performance model. The extrapolated rate should be annotated to indicate if it was run

length or peak rate limited, since these have different predictive values.

Other issues:

If the link RTT is not substantially smaller than the target RTT and the actual run length is close to the target rate, a standards compliant TCP implementation might not be effective at accurately controlling the data rate. To be independent of the details of the TCP implementation, failing to control the rate has to be treated as a spoiled measurement, not a infrastructure failure. This can be overcome by "stiffening" TCP by using a non-standard congestion control algorithm. For example if the rate controlling by clamping cwnd then use "relentless TCP" style reductions on loss, and lock ssthresh to the cwnd clamp. Alternatively, implement an explicit rate controller for TCP. In either case the test must be abandoned (aborted) if the measured run length is substantially below the target run length.

If the test is run "in situ" in a production environment, there also needs to be baseline tests using alternate paths to confirm that there are no bottlenecks or congested links between the test end points and the link under test.

It might make sense to run multiple tests with different parameters, for example infrequent tests with test rate equal to the target rate, and more frequent, less disruptive tests with the same target rate but the test rate equal to 1% of the target rate. To observe the required run length, the low rate test would take 100 times longer to run.

Returning to the example: a full rate test would entail sending 690 pps (1 MByte/s) for several tens of seconds (e.g. 50k packets), and observing that the total loss rate is below 1:1e4. A less disruptive test might be to send at 6.9 pps for 100 times longer, and observing

Formatted: Mon Oct 15 16:00:51 PDT 2012

Author's Address

Matt Mathis
Google, Inc
1600 Amphitheater Parkway
Mountain View, California 93117
USA

Email: mattmathis@google.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 26, 2013

J. Fabini
Vienna University of Technology
A. Morton
AT&T Labs
February 22, 2013

Advanced Stream and Sampling Framework for IPPM
draft-morton-ippm-2330-update-01

Abstract

To obtain repeatable results in modern networks, test descriptions need an expanded stream parameter framework that also augments aspects specified as Type-P for test packets. This memo proposes to update the IP Performance Metrics (IPPM) Framework with advanced considerations for measurement methodology and testing. The existing framework mostly assumes deterministic connectivity, and that a single test stream will represent the characteristics of the path when it is aggregated with other flows. Networks have evolved and test stream descriptions must evolve with them, otherwise unexpected network features may dominate the measured performance. This memo describes new stream parameters for both network characterization and support of application design using IPPM metrics.

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 26, 2013.

Copyright Notice

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

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

Table of Contents

1. Introduction	3
1.1. Definition: Reactive Network Behavior	3
2. Scope	4
3. New Stream Parameters	4
3.1. Test Packet Type-P	5
3.1.1. Test Packet Length	6
3.1.2. Test Packet Payload Content Optimization	6
3.2. Packet History	6
3.3. Access Technology Change	7
3.4. Time-Slotted Network Paths	7
4. Conclusions	8
5. Security Considerations	9
6. IANA Considerations	9
7. Acknowledgements	9
8. References	9
8.1. Normative References	9
8.2. Informative References	10
Authors' Addresses	10

1. Introduction

The IETF IP Performance Metrics (IPPM) working group first created a framework for metric development in [RFC2330]. This framework has stood the test of time and enabled development of many fundamental metrics, while only being updated once in a specific area [RFC5835].

The IPPM framework [RFC2330] generally relies on several assumptions, one of which is not explicitly stated but assumed: the network behaves (halfway) deterministic and without state/history-less (with some exceptions, firewalls are mentioned). However, this does not hold true for many modern network technologies, such as reactive networks (those with demand-driven resource allocation) and links with time-slotted operation. Per-flow state can be observed on test packet streams, and such treatment will influence network characterization if it is not taken into account. Flow history will also affect the performance of applications and be perceived by their users.

Moreover, Sections 4 and 6.2 of [RFC2330] explicitly recommend repeatable measurement metrics and methodologies. Measurements in today's access networks illustrate that methodological guidelines of [RFC2330] must be extended to capture the reactive nature of these networks. Although the proposed extensions can support methodologies to fulfill the continuity requirement stated in section 6.2 of [RFC2330], there is no guarantee. Practical measurements confirm that some link types exhibit distinct responses to repeated measurements with identical stimulus, i.e., identical traffic patterns. If feasible, appropriate fine-tuning of measurement traffic patterns can improve measurement continuity and repeatability for these link types as shown in [IBD].

1.1. Definition: Reactive Network Behavior

Reactive behavior is present when link-or host-internal sensing of packet arrival for the flow of interest indicates that traffic is absent or present, or that traffic during an assessment interval is above or below a threshold, and the results of one or more successive assessments cause one or more network components to process future packets using a different mode of operation than for other assessment outcomes.

Reactive network behavior must be observable by the measurement flow as a repeatable phenomenon where packet transfer performance characteristics **change** according to prior node- or link-internal observations of the packet flow of interest. Therefore, reactive network behavior is deterministic with respect to the flow of interest. Other flows or traffic load conditions may result in

additional performance-affecting reactions, but these are external to the characteristics of the flow of interest.

Other than the size of the payload at the layer of interest and the header itself, packet content does not influence the measurement. Reactive behavior at the IP layer is not influenced by the TCP ports in use, for example. Therefore, the finding of reactive behavior must specify the layer at which assessment leading to reaction appears to be taking place. In any case, the full packet Type-P used in measurement should always be specified.

Examples include links with Active/In-active state detectors, and network devices or links that revise their traffic serving and forwarding rates (up or down) based on packet arrival history.

A network or network path is said to be reactive when at least one of the links or hosts comprising the path exhibits reactive behavior.

2. Scope

The scope of this memo is to describe useful stream parameters in addition to the information in Section 11.1 of [RFC2330] and described in [RFC3432] for periodic streams. The purpose is to foster repeatable measurement results in modern networks by highlighting the key aspects of test streams and packets and make them part of the IPPM performance metric framework.

3. New Stream Parameters

There are several areas where measurement methodology definition and test result interpretation will benefit from an increased understanding of the stream characteristics and the (possibly unknown) network condition that influence the measured metrics.

1. Network treatment depends on the fullest extent on the "packet of Type-P" definition in [RFC2330], and has for some time.

- * State is often maintained on the per-flow basis at various points in the network, where "flows" are determined by IP and other layers. Significant treatment differences occur with the simplest of Type-P parameters: packet length.
- * Payload content optimization (compression or format conversion) in intermediate segments. This breaks the convention of payload correspondence when correlating measurements made at different points in a path.

2. Packet history (instantaneous or recent test rate or inactivity, also for non-test traffic) profoundly influences measured performance, in addition to all the Type-P parameters described in [RFC2330].
3. Access technology may change during testing. A range of transfer capacities and access methods may be encountered during a test session. When different interfaces are used, the host seeking access will be aware of the technology change which differentiates this form of path change from other changes in network state. Section 14 of [RFC2330] treats the possibility that a host may have more than one attachment to the network, and also that assessment of the measurement path (route) is valid for some length of time (in Section 5 and Section 7 of [RFC2330]). Here we combine these two considerations under the assumption that changes may be more frequent and possibly have greater consequences on performance metrics.
4. Paths including links or nodes with time-slotted service opportunities represent several challenges to measurement (when service time period is appreciable):
 - * Random/unbiased sampling is not possible beyond one such link in the path.
 - * The above encourages a segmented approach to end to end measurement, as described in [RFC6049] for Network Characterization (as defined in [RFC6703]) to understand the full range of delay and delay variation on the path. Alternatively, if application performance estimation is the goal (also defined in [RFC6703]), then a stream with un-biased or known-bias properties [RFC3432] may be sufficient.
 - * Multi-modal delay variation makes central statistics unimportant, others must be used instead.

Each of these topics is treated in detail below.

3.1. Test Packet Type-P

We recommend two Type-P parameters to be added to the factors which have impact on network performance measurements, namely packet length and payload type. Carefully choosing these parameters can improve measurement methodologies in their continuity and repeatability when deployed in reactive networks.

3.1.1. Test Packet Length

Many instances of network characterization using IPPM metrics have relied on a single test packet length. When testing to assess application performance or an aggregate of traffic, benchmarking methods have used a range of fixed lengths and frequently augmented fixed size tests with a mixture of sizes, or IMIX as described in [I-D.ietf-bmwg-imix-genome].

Test packet length influences delay measurements, in that the IPPM one-way delay metric [RFC2679] includes serialization time in its first-bit to last bit time stamping requirements. However, different sizes can have a larger effect on link delay and link delay variation than serialization would explain alone. This effect can be non-linear and change instantaneous or future network performance.

Repeatability is a main measurement methodology goal as stated in section 6.2 of [RFC2330]. To eliminate packet length as a potential measurement uncertainty factor, successive measurements must use identical traffic patterns. In practice a combination of random payload and random start time can yield representative results as illustrated in [IRR].

3.1.2. Test Packet Payload Content Optimization

The aim for efficient network resource use has resulted in a series of "smart" networks to deploy server-only or client-server lossless or lossy payload compression techniques on some links or paths. These optimizers attempt to compress high-volume traffic in order to reduce network load. Files are analyzed by application-layer parsers and parts (like comments) might be dropped. Although typically acting on HTTP or JPEG files, compression might affect measurement packets, too. In particular measurement packets are qualified for efficient compression when they use standard plain-text payload.

IPPM-conforming measurements should add packet payload content as a Type-P parameter which can help to improve measurement determinism. Some packet payloads are more susceptible to compression than others, but optimizers in the measurement path can be out ruled by using incompressible packet payload. This payload content could be either generated by a random device or by using part of a compressed file (e.g., a part of a ZIP compressed archive).

3.2. Packet History

Recent packet history and instantaneous data rate influence measurement results for reactive links supporting on-demand capacity allocation. Measurement uncertainty may be reduced by knowledge of

measurement packet history and total host load. Additionally, small changes in history, e.g., because of lost packets along the path, can be the cause of large performance variations.

For instance delay in reactive 3G networks like High Speed Packet Access (HSPA) depends to a large extent on the test traffic data rate. The reactive resource allocation strategy in these networks affects the uplink direction in particular. Small changes in data rate can be the reason of more than 200% increase in delay, depending on the specific packet size.

The reactive behavior of a network under test may require to "pre-load" paths with traffic, or to separate early traffic that captures the reactive network performance signature from steady conditions that follow.

3.3. Access Technology Change

[RFC2330] discussed the scenario of multi-homed hosts. If hosts become aware of access technology changes (e.g., because of IP address changes or lower layer information) and make this information available, measurement methodologies can use this information to improve measurement representativeness and relevance.

However, today's various access network technologies can present the same physical interface to the host. A host may or may not become aware when its access technology changes on such an interface. Measurements for networks which support on-demand capacity allocation are therefore challenging in that it is difficult to differentiate between access technology changes (e.g., because of mobility) and reactive network behavior (e.g., because of data rate change).

3.4. Time-Slotted Network Paths

Time-Slotted operation of network entities - interfaces, routers or links - in a network path is a particular challenge for measurements, especially if the time slot period is substantial. The central observation as an extension to Poisson stream sampling in [RFC2330] is that the first such time-slotted component cancels unbiased measurement stream sampling. In the worst case, time-slotted operation converts an unbiased, random measurement packet stream into a periodic packet stream. Being heavily biased, these packets may interact with periodic network behavior of subsequent time-slotted network entities.

Practical measurements confirm that such interference limits delay measurement variation to a sub-set of theoretical value range. Measurement samples for such cases can aggregate on artificial

limits, generating multi-modal distributions as demonstrated in [IRR]. In this context, the desirable measurement sample statistics differentiate between multi-modal delay distributions caused by reactive network behavior and the ones due to time-slotted interference.

The amount of measurement bias is determined by the relative offset between allocated time-slots in subsequent network entities, delay variation in these networks, and other sources of variation. Measurement results might change over time, depending on how accurately the sending host, receiving host, and time-slotted components in the measurement path are synchronized to each other and to global time. If network segments maintain flow state, flow parameter change or flow re-allocations can cause substantial variation in measurement results.

Measurement methodology selection for time-slotted paths depends to a large extent on the respective viewpoint. End-to-end metrics can provide accurate measurement results for short-term sessions and low likelihood of flow state modifications. Applications or services which aim at approximating network performance for a short time interval (in the order of minutes) and expect stable network conditions should therefore prefer end-to-end metrics. Here stable network conditions refer to any kind of global knowledge concerning measurement path flow state and flow parameters.

However, if long-term forecast of time-slotted network performance is the main measurement goal, a segmented approach relying on hop-by-hop metrics is preferred. Re-generating unbiased measurement traffic at any hop can help to unleash the true range of network performance for all network segments.

4. Conclusions

Safeguarding continuity and repeatability as key properties of measurement methodologies is highly challenging and sometimes impossible in reactive networks. Measurements in networks with demand-driven allocation strategies must use a prototypical application packet stream to infer a specific application's performance. Measurement repetition with unbiased network and flow states (e.g., by rebooting measurement hosts) can help to avoid interference with periodic network behavior, randomness being a mandatory feature for avoiding correlation with network timing. Inferring from one measurement session or packet stream onto network performance for alternative streams is highly discouraged in reactive networks because of the huge set of global parameters which influence on instantaneous network performance.

5. Security Considerations

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

6. IANA Considerations

This memo makes no requests of IANA.

7. Acknowledgements

The authors thank Rudiger Geib and Matt Mathis for their helpful comments on this draft.

8. References

8.1. Normative References

- [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.
- [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.
- [RFC5835] Morton, A. and S. Van den Berghe, "Framework for Metric Composition", RFC 5835, April 2010.
- [RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, January 2011.
- [RFC6576] Geib, R., Morton, A., Fardid, R., and A. Steinmitz, "IP Performance Metrics (IPPM) Standard Advancement Testing", BCP 176, RFC 6576, March 2012.
- [RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View", RFC 6703, August 2012.

8.2. Informative References

- [I-D.ietf-bmwg-imix-genome] Morton, A., "IMIX Genome: Specification of variable packet sizes for additional testing", draft-ietf-bmwg-imix-genome-04 (work in progress), December 2012.
- [IBD] Fabini, J., "The Illusion of Being Deterministic - Application-Level Considerations on Delay in 3G HSPA Networks", Lecture Notes in Computer Science, Springer, Volume 5550, 2009, pp 301-312 , May 2009.
- [IRR] Fabini, J., "The Importance of Being Really Random: Methodological Aspects of IP-Layer 2G and 3G Network Delay Assessment", ICC'09 Proceedings of the 2009 IEEE International Conference on Communications, doi: 10.1109/ICC.2009.5199514, June 2009.

Authors' Addresses

Joachim Fabini
Vienna University of Technology
Favoritenstrasse 9/E389
Vienna, 1040
Austria

Phone: +43 1 58801 38813
Fax: +43 1 58801 38898
Email: Joachim.Fabini@tuwien.ac.at
URI: <http://www.tc.tuwien.ac.at/about-us/staff/joachim-fabini/>

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
Obsoletes: 2679 (if approved)
Intended status: Standards Track
Expires: April 24, 2013

G. Almes
Texas A&M
S. Kalidindi
Ixia
M. Zekauskas
Internet2
A. Morton, Ed.
AT&T Labs
October 21, 2012

A One-Way Delay Metric for IPPM
draft-morton-ippm-2679-bis-01

Abstract

This memo (RFC 2679 bis) defines a metric for one-way delay of packets across Internet paths. It builds on notions introduced and discussed in the IPPM Framework document, RFC 2330; the reader is assumed to be familiar with that document.

Requirements Language

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

Status of this Memo

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

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

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

This Internet-Draft will expire on April 24, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. RFC 2679 bis	4
2. Introduction	4
2.1. Motivation	5
2.2. General Issues Regarding Time	6
3. A Singleton Definition for One-way Delay	7
3.1. Metric Name:	7
3.2. Metric Parameters:	7
3.3. Metric Units:	8
3.4. Definition:	8
3.5. Discussion:	8
3.6. Methodologies:	9
3.7. Errors and Uncertainties:	10
3.7.1. Errors or uncertainties related to Clocks	10
3.7.2. Errors or uncertainties related to Wire-time vs Host-time	12
3.7.3. Calibration	12
3.8. Reporting the metric:	14
3.8.1. Type-P	14
3.8.2. Loss Threshold	15
3.8.3. Calibration Results	15
3.8.4. Path	15
4. A Definition for Samples of One-way Delay	15
4.1. Metric Name:	16
4.2. Metric Parameters:	16
4.3. Metric Units:	16
4.4. Definition:	16
4.5. Discussion:	17
4.6. Methodologies:	17
4.7. Errors and Uncertainties:	18
4.8. Reporting the metric:	18
5. Some Statistics Definitions for One-way Delay	18
5.1. Type-P-One-way-Delay-Percentile	18
5.2. Type-P-One-way-Delay-Median	19
5.3. Type-P-One-way-Delay-Minimum	20
5.4. Type-P-One-way-Delay-Inverse-Percentile	20
6. Security Considerations	20
7. IANA Considerations	21
8. Acknowledgements	21
9. Refetrences (temporary)	21
10. References	22
10.1. Normative References	22
10.2. Informative References	23
Authors' Addresses	23

1. RFC 2679 bis

The following text constitutes RFC 2769 bis proposed for advancement on the IETF Standards Track.

[I-D.ietf-ippm-testplan-rfc2679] (now approved) provides the test plan and results supporting [RFC2679] advancement along the standards track, according to the process in [RFC6576]. The conclusions of [I-D.ietf-ippm-testplan-rfc2679] list four minor modifications for inclusion:

1. Section 6.2.3 of [I-D.ietf-ippm-testplan-rfc2679] asserts that the assumption of post-processing to enforce a constant waiting time threshold is compliant, and that the text of the RFC should be revised slightly to include this point (see the last list item of section 3.6, below).
2. Section 6.5 of [I-D.ietf-ippm-testplan-rfc2679] indicates that Type-P-One-way-Delay-Inverse-Percentile statistic has been ignored in both implementations, so it is a candidate for removal or deprecation in RFC2679bis (this small discrepancy does not affect candidacy for advancement) (see section 5.4, below).
3. The IETF has reached consensus on guidance for reporting metrics in [RFC6703], and this memo should be referenced in RFC2679bis to incorporate recent experience where appropriate (see the last list item of section 3.6, section 3.8, and section 5 below).
4. There is currently one erratum with status "Held for document update" for [RFC2679], and it appears this minor revision and additional text should be incorporated in RFC2679bis (see section 5.1).

A small number of updates to the [RFC2679] text have been proposed (by the current Editor) in the text below, principally to reference key IPPM RFCs that were approved after [RFC2679].

2. Introduction

This memo defines a metric for one-way delay of packets across Internet paths. It builds on notions introduced and discussed in the IPPM Framework document, RFC 2330 [1]; the reader is assumed to be familiar with that document.

This memo is intended to be parallel in structure to a companion document for Packet Loss ("A One-way Packet Loss Metric for IPPM") [2].

Although RFC 2119 was written with protocols in mind, the key words are used in this document for similar reasons. They are used to ensure the results of measurements from two different implementations are comparable, and to note instances when an implementation could perturb the network.

The structure of the memo is as follows:

- + A 'singleton' analytic metric, called Type-P-One-way-Delay, will be introduced to measure a single observation of one-way delay.

- + Using this singleton metric, a 'sample', called Type-P-One-way-Delay-Poisson-Stream, will be introduced to measure a sequence of singleton delays measured at times taken from a Poisson process.

- + Using this sample, several 'statistics' of the sample will be defined and discussed. This progression from singleton to sample to statistics, with clear separation among them, is important.

Whenever a technical term from the IPPM Framework document is first used in this memo, it will be tagged with a trailing asterisk. For example, "term*" indicates that "term" is defined in the Framework.

2.1. Motivation

One-way delay of a Type-P* packet from a source host* to a destination host is useful for several reasons:

- + Some applications do not perform well (or at all) if end-to-end delay between hosts is large relative to some threshold value.

- + Erratic variation in delay makes it difficult (or impossible) to support many real-time applications.

- + The larger the value of delay, the more difficult it is for transport-layer protocols to sustain high bandwidths.

- + The minimum value of this metric provides an indication of the delay due only to propagation and transmission delay.

- + The minimum value of this metric provides an indication of the delay that will likely be experienced when the path* traversed is lightly loaded.

- + Values of this metric above the minimum provide an indication of the congestion present in the path.

The measurement of one-way delay instead of round-trip delay is

motivated by the following factors:

- + In today's Internet, the path from a source to a destination may be different than the path from the destination back to the source ("asymmetric paths"), such that different sequences of routers are used for the forward and reverse paths. Therefore round-trip measurements actually measure the performance of two distinct paths together. Measuring each path independently highlights the performance difference between the two paths which may traverse different Internet service providers, and even radically different types of networks (for example, research versus commodity networks, or ATM versus packet-over-SONET).

- + Even when the two paths are symmetric, they may have radically different performance characteristics due to asymmetric queueing.

- + Performance of an application may depend mostly on the performance in one direction. For example, a file transfer using TCP may depend more on the performance in the direction that data flows, rather than the direction in which acknowledgements travel.

- + In quality-of-service (QoS) enabled networks, provisioning in one direction may be radically different than provisioning in the reverse direction, and thus the QoS guarantees differ. Measuring the paths independently allows the verification of both guarantees.

It is outside the scope of this document to say precisely how delay metrics would be applied to specific problems.

2.2. General Issues Regarding Time

{Comment: the terminology below differs from that defined by ITU-T documents (e.g., G.810, "Definitions and terminology for synchronization networks" and I.356, "B-ISDN ATM layer cell transfer performance"), but is consistent with the IPPM Framework document. In general, these differences derive from the different backgrounds; the ITU-T documents historically have a telephony origin, while the authors of this document (and the Framework) have a computer systems background. Although the terms defined below have no direct equivalent in the ITU-T definitions, after our definitions we will provide a rough mapping. However, note one potential confusion: our definition of "clock" is the computer operating systems definition denoting a time-of-day clock, while the ITU-T definition of clock denotes a frequency reference.}

Whenever a time (i.e., a moment in history) is mentioned here, it is understood to be measured in seconds (and fractions) relative to UTC.

As described more fully in the Framework document, there are four distinct, but related notions of clock uncertainty:

synchronization*

measures the extent to which two clocks agree on what time it is. For example, the clock on one host might be 5.4 msec ahead of the clock on a second host. {Comment: A rough ITU-T equivalent is "time error".}

accuracy*

measures the extent to which a given clock agrees with UTC. For example, the clock on a host might be 27.1 msec behind UTC. {Comment: A rough ITU-T equivalent is "time error from UTC".}

resolution*

measures the precision of a given clock. For example, the clock on an old Unix host might tick only once every 10 msec, and thus have a resolution of only 10 msec. {Comment: A very rough ITU-T equivalent is "sampling period".}

skew*

measures the change of accuracy, or of synchronization, with time. For example, the clock on a given host might gain 1.3 msec per hour and thus be 27.1 msec behind UTC at one time and only 25.8 msec an hour later. In this case, we say that the clock of the given host has a skew of 1.3 msec per hour relative to UTC, which threatens accuracy. We might also speak of the skew of one clock relative to another clock, which threatens synchronization. {Comment: A rough ITU-T equivalent is "time drift".}

3. A Singleton Definition for One-way Delay

3.1. Metric Name:

Type-P-One-way-Delay

3.2. Metric Parameters:

- + Src, the IP address of a host
- + Dst, the IP address of a host
- + T, a time

3.3. Metric Units:

The value of a Type-P-One-way-Delay is either a real number, or an undefined (informally, infinite) number of seconds.

3.4. Definition:

For a real number dT , \gg the *Type-P-One-way-Delay* from Src to Dst at T is dT \ll means that Src sent the first bit of a Type-P packet to Dst at wire-time T and that Dst received the last bit of that packet at wire-time $T+dT$.

\gg The *Type-P-One-way-Delay* from Src to Dst at T is undefined (informally, infinite) \ll means that Src sent the first bit of a Type-P packet to Dst at wire-time T and that Dst did not receive that packet.

Suggestions for what to report along with metric values appear in Section 3.8 after a discussion of the metric, methodologies for measuring the metric, and error analysis.

3.5. Discussion:

Type-P-One-way-Delay is a relatively simple analytic metric, and one that we believe will afford effective methods of measurement.

The following issues are likely to come up in practice:

- + Real delay values will be positive. Therefore, it does not make sense to report a negative value as a real delay. However, an individual zero or negative delay value might be useful as part of a stream when trying to discover a distribution of a stream of delay values.

- + Since delay values will often be as low as the 100 usec to 10 msec range, it will be important for Src and Dst to synchronize very closely. GPS systems afford one way to achieve synchronization to within several 10s of usec. Ordinary application of NTP may allow synchronization to within several msec, but this depends on the stability and symmetry of delay properties among those NTP agents used, and this delay is what we are trying to measure. A combination of some GPS-based NTP servers and a conservatively designed and deployed set of other NTP servers should yield good results, but this is yet to be tested.

- + A given methodology will have to include a way to determine whether a delay value is infinite or whether it is merely very large (and the packet is yet to arrive at Dst). As noted by Mahdavi and Paxson [4],

simple upper bounds (such as the 255 seconds theoretical upper bound on the lifetimes of IP packets [5]) could be used, but good engineering, including an understanding of packet lifetimes, will be needed in practice. {Comment: Note that, for many applications of these metrics, the harm in treating a large delay as infinite might be zero or very small. A TCP data packet, for example, that arrives only after several multiples of the RTT may as well have been lost.}

- + If the packet is duplicated along the path (or paths) so that multiple non-corrupt copies arrive at the destination, then the packet is counted as received, and the first copy to arrive determines the packet's one-way delay.

- + If the packet is fragmented and if, for whatever reason, reassembly does not occur, then the packet will be deemed lost.

3.6. Methodologies:

As with other Type-P-* metrics, the detailed methodology will depend on the Type-P (e.g., protocol number, UDP/TCP port number, size, precedence).

Generally, for a given Type-P, the methodology would proceed as follows:

- + Arrange that Src and Dst are synchronized; that is, that they have clocks that are very closely synchronized with each other and each fairly close to the actual time.

- + At the Src host, select Src and Dst IP addresses, and form a test packet of Type-P with these addresses. Any 'padding' portion of the packet needed only to make the test packet a given size should be filled with randomized bits to avoid a situation in which the measured delay is lower than it would otherwise be due to compression techniques along the path.

- + At the Dst host, arrange to receive the packet.

- + At the Src host, place a timestamp in the prepared Type-P packet, and send it towards Dst.

- + If the packet arrives within a reasonable period of time, take a timestamp as soon as possible upon the receipt of the packet. By subtracting the two timestamps, an estimate of one-way delay can be computed. Error analysis of a given implementation of the method must take into account the closeness of synchronization between Src and Dst. If the delay between Src's timestamp and the actual sending of the packet is known, then the estimate could be adjusted by

subtracting this amount; uncertainty in this value must be taken into account in error analysis. Similarly, if the delay between the actual receipt of the packet and Dst's timestamp is known, then the estimate could be adjusted by subtracting this amount; uncertainty in this value must be taken into account in error analysis. See the next section, "Errors and Uncertainties", for a more detailed discussion.

+ If the packet fails to arrive within a reasonable period of time, the one-way delay is taken to be undefined (informally, infinite). Note that the threshold of 'reasonable' is a parameter of the methodology. These points are examined in detail in [RFC6703], including analysis preferences to assign undefined delay to packets that fail to arrive with the difficulties emerging from the informal "infinite delay" assignment, and an estimation of an upper bound on waiting time for packets in transit. Further, enforcing a specific constant waiting time on stored singletons of one-way delay is compliant with this specification and may allow the results to serve more than one reporting audience.

Issues such as the packet format, the means by which Dst knows when to expect the test packet, and the means by which Src and Dst are synchronized are outside the scope of this document. {Comment: We plan to document elsewhere our own work in describing such more detailed implementation techniques and we encourage others to as well.}

3.7. Errors and Uncertainties:

The description of any specific measurement method should include an accounting and analysis of various sources of error or uncertainty. The Framework document provides general guidance on this point, but we note here the following specifics related to delay metrics:

+ Errors or uncertainties due to uncertainties in the clocks of the Src and Dst hosts.

+ Errors or uncertainties due to the difference between 'wire time' and 'host time'.

In addition, the loss threshold may affect the results. Each of these are discussed in more detail below, along with a section ("Calibration") on accounting for these errors and uncertainties.

3.7.1. Errors or uncertainties related to Clocks

The uncertainty in a measurement of one-way delay is related, in part, to uncertainties in the clocks of the Src and Dst hosts. In

the following, we refer to the clock used to measure when the packet was sent from Src as the source clock, we refer to the clock used to measure when the packet was received by Dst as the destination clock, we refer to the observed time when the packet was sent by the source clock as T_{source} , and the observed time when the packet was received by the destination clock as T_{dest} . Alluding to the notions of synchronization, accuracy, resolution, and skew mentioned in the Introduction, we note the following:

- + Any error in the synchronization between the source clock and the destination clock will contribute to error in the delay measurement. We say that the source clock and the destination clock have a synchronization error of T_{synch} if the source clock is T_{synch} ahead of the destination clock. Thus, if we know the value of T_{synch} exactly, we could correct for clock synchronization by adding T_{synch} to the uncorrected value of $T_{dest} - T_{source}$.

- + The accuracy of a clock is important only in identifying the time at which a given delay was measured. Accuracy, per se, has no importance to the accuracy of the measurement of delay. When computing delays, we are interested only in the differences between clock values, not the values themselves.

- + The resolution of a clock adds to uncertainty about any time measured with it. Thus, if the source clock has a resolution of 10 msec, then this adds 10 msec of uncertainty to any time value measured with it. We will denote the resolution of the source clock and the destination clock as R_{source} and R_{dest} , respectively.

- + The skew of a clock is not so much an additional issue as it is a realization of the fact that T_{synch} is itself a function of time. Thus, if we attempt to measure or to bound T_{synch} , this needs to be done periodically. Over some periods of time, this function can be approximated as a linear function plus some higher order terms; in these cases, one option is to use knowledge of the linear component to correct the clock. Using this correction, the residual T_{synch} is made smaller, but remains a source of uncertainty that must be accounted for. We use the function $E_{synch}(t)$ to denote an upper bound on the uncertainty in synchronization. Thus, $|T_{synch}(t)| \leq E_{synch}(t)$.

Taking these items together, we note that naive computation $T_{dest} - T_{source}$ will be off by $T_{synch}(t) \pm (R_{source} + R_{dest})$. Using the notion of $E_{synch}(t)$, we note that these clock-related problems introduce a total uncertainty of $E_{synch}(t) + R_{source} + R_{dest}$. This estimate of total clock-related uncertainty should be included in the error/uncertainty analysis of any measurement implementation.

3.7.2. Errors or uncertainties related to Wire-time vs Host-time

As we have defined one-way delay, we would like to measure the time between when the test packet leaves the network interface of Src and when it (completely) arrives at the network interface of Dst, and we refer to these as "wire times." If the timings are themselves performed by software on Src and Dst, however, then this software can only directly measure the time between when Src grabs a timestamp just prior to sending the test packet and when Dst grabs a timestamp just after having received the test packet, and we refer to these two points as "host times".

To the extent that the difference between wire time and host time is accurately known, this knowledge can be used to correct for host time measurements and the corrected value more accurately estimates the desired (wire time) metric.

To the extent, however, that the difference between wire time and host time is uncertain, this uncertainty must be accounted for in an analysis of a given measurement method. We denote by Hsource an upper bound on the uncertainty in the difference between wire time and host time on the Src host, and similarly define Hdest for the Dst host. We then note that these problems introduce a total uncertainty of Hsource+Hdest. This estimate of total wire-vs-host uncertainty should be included in the error/uncertainty analysis of any measurement implementation.

3.7.3. Calibration

Generally, the measured values can be decomposed as follows:

$$\text{measured value} = \text{true value} + \text{systematic error} + \text{random error}$$

If the systematic error (the constant bias in measured values) can be determined, it can be compensated for in the reported results.

$$\text{reported value} = \text{measured value} - \text{systematic error}$$

therefore

$$\text{reported value} = \text{true value} + \text{random error}$$

The goal of calibration is to determine the systematic and random error generated by the instruments themselves in as much detail as possible. At a minimum, a bound ("e") should be found such that the reported value is in the range (true value - e) to (true value + e) at least 95 percent of the time. We call "e" the calibration error for the measurements. It represents the degree to which the values

produced by the measurement instrument are repeatable; that is, how closely an actual delay of 30 ms is reported as 30 ms. {Comment: 95 percent was chosen because (1) some confidence level is desirable to be able to remove outliers, which will be found in measuring any physical property; (2) a particular confidence level should be specified so that the results of independent implementations can be compared; and (3) even with a prototype user-level implementation, 95% was loose enough to exclude outliers.}

From the discussion in the previous two sections, the error in measurements could be bounded by determining all the individual uncertainties, and adding them together to form

$$E_{\text{synch}}(t) + R_{\text{source}} + R_{\text{dest}} + H_{\text{source}} + H_{\text{dest}}.$$

However, reasonable bounds on both the clock-related uncertainty captured by the first three terms and the host-related uncertainty captured by the last two terms should be possible by careful design techniques and calibrating the instruments using a known, isolated, network in a lab.

For example, the clock-related uncertainties are greatly reduced through the use of a GPS time source. The sum of $E_{\text{synch}}(t) + R_{\text{source}} + R_{\text{dest}}$ is small, and is also bounded for the duration of the measurement because of the global time source.

The host-related uncertainties, $H_{\text{source}} + H_{\text{dest}}$, could be bounded by connecting two instruments back-to-back with a high-speed serial link or isolated LAN segment. In this case, repeated measurements are measuring the same one-way delay.

If the test packets are small, such a network connection has a minimal delay that may be approximated by zero. The measured delay therefore contains only systematic and random error in the instrumentation. The "average value" of repeated measurements is the systematic error, and the variation is the random error.

One way to compute the systematic error, and the random error to a 95% confidence is to repeat the experiment many times - at least hundreds of tests. The systematic error would then be the median. The random error could then be found by removing the systematic error from the measured values. The 95% confidence interval would be the range from the 2.5th percentile to the 97.5th percentile of these deviations from the true value. The calibration error "e" could then be taken to be the largest absolute value of these two numbers, plus the clock-related uncertainty. {Comment: as described, this bound is relatively loose since the uncertainties are added, and the absolute value of the largest deviation is used. As long as the resulting

value is not a significant fraction of the measured values, it is a reasonable bound. If the resulting value is a significant fraction of the measured values, then more exact methods will be needed to compute the calibration error.}

Note that random error is a function of measurement load. For example, if many paths will be measured by one instrument, this might increase interrupts, process scheduling, and disk I/O (for example, recording the measurements), all of which may increase the random error in measured singletons. Therefore, in addition to minimal load measurements to find the systematic error, calibration measurements should be performed with the same measurement load that the instruments will see in the field.

We wish to reiterate that this statistical treatment refers to the calibration of the instrument; it is used to "calibrate the meter stick" and say how well the meter stick reflects reality.

In addition to calibrating the instruments for finite one-way delay, two checks should be made to ensure that packets reported as losses were really lost. First, the threshold for loss should be verified. In particular, ensure the "reasonable" threshold is reasonable: that it is very unlikely a packet will arrive after the threshold value, and therefore the number of packets lost over an interval is not sensitive to the error bound on measurements. Second, consider the possibility that a packet arrives at the network interface, but is lost due to congestion on that interface or to other resource exhaustion (e.g. buffers) in the instrument.

3.8. Reporting the metric:

The calibration and context in which the metric is measured MUST be carefully considered, and SHOULD always be reported along with metric results. We now present four items to consider: the Type-P of test packets, the threshold of infinite delay (if any), error calibration, and the path traversed by the test packets. This list is not exhaustive; any additional information that could be useful in interpreting applications of the metrics should also be reported (see [RFC6703] for extensive discussion of reporting considerations for different audiences).

3.8.1. Type-P

As noted in the Framework document [1], the value of the metric may depend on the type of IP packets used to make the measurement, or "type-P". The value of Type-P-One-way-Delay could change if the protocol (UDP or TCP), port number, size, or arrangement for special treatment (e.g., IP precedence or RSVP) changes. The exact Type-P

used to make the measurements MUST be accurately reported.

3.8.2. Loss Threshold

In addition, the threshold (or methodology to distinguish) between a large finite delay and loss MUST be reported.

3.8.3. Calibration Results

- + If the systematic error can be determined, it SHOULD be removed from the measured values.

- + You SHOULD also report the calibration error, e , such that the true value is the reported value plus or minus e , with 95% confidence (see the last section.)

- + If possible, the conditions under which a test packet with finite delay is reported as lost due to resource exhaustion on the measurement instrument SHOULD be reported.

3.8.4. Path

Finally, the path traversed by the packet SHOULD be reported, if possible. In general it is impractical to know the precise path a given packet takes through the network. The precise path may be known for certain Type-P on short or stable paths. If Type-P includes the record route (or loose-source route) option in the IP header, and the path is short enough, and all routers* on the path support record (or loose-source) route, then the path will be precisely recorded. This is impractical because the route must be short enough, many routers do not support (or are not configured for) record route, and use of this feature would often artificially worsen the performance observed by removing the packet from common-case processing. However, partial information is still valuable context. For example, if a host can choose between two links* (and hence two separate routes from Src to Dst), then the initial link used is valuable context. {Comment: For example, with Merit's NetNow setup, a Src on one NAP can reach a Dst on another NAP by either of several different backbone networks.}

4. A Definition for Samples of One-way Delay

Given the singleton metric Type-P-One-way-Delay, 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 T. The means for defining the values of T is to select a beginning time T_0 , a final time T_f ,

and an average rate λ , then define a pseudo-random Poisson process of rate λ , whose values fall between T_0 and T_f . The time interval between successive values of T will then average $1/\lambda$.

{Comment: Note that Poisson sampling is only one way of defining a sample. Poisson has the advantage of limiting bias, but other methods of sampling might be appropriate for different situations. We encourage others who find such appropriate cases to use this general framework and submit their sampling method for standardization.}

>>> Editor proposal: Add ref to RFC 3432 Periodic sampling above.

4.1. Metric Name:

Type-P-One-way-Delay-Poisson-Stream

4.2. Metric Parameters:

- + Src, the IP address of a host
- + Dst, the IP address of a host
- + T_0 , a time
- + T_f , a time
- + λ , a rate in reciprocal seconds

4.3. 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.

4.4. Definition:

Given T_0 , T_f , and λ , we compute a pseudo-random Poisson process beginning at or before T_0 , with average arrival rate λ , and ending at or after T_f . Those time values greater than or equal to T_0 and less than or equal to T_f are then selected. At each of the times

in this process, we obtain the value of Type-P-One-way-Delay at this time. The value of the sample is the sequence made up of the resulting <time, delay> pairs. If there are no such pairs, the sequence is of length zero and the sample is said to be empty.

4.5. Discussion:

The reader should be familiar with the in-depth discussion of Poisson sampling in the Framework document [1], which includes methods to compute and verify the pseudo-random Poisson process.

We specifically do not constrain the value of lambda, except to note the extremes. If the rate is too large, then the measurement traffic will perturb the network, and itself cause congestion. If the rate is too small, then you might not capture interesting network behavior. {Comment: We expect to document our experiences with, and suggestions for, lambda elsewhere, culminating in a "best current practices" document.}

Since a pseudo-random number sequence is employed, the sequence of times, and hence the value of the sample, is not fully specified. Pseudo-random number generators of good quality will be needed to achieve the desired qualities.

The sample is defined in terms of a Poisson process both to avoid the effects of self-synchronization and also capture a sample that is statistically as unbiased as possible. {Comment: there is, of course, no claim that real Internet traffic arrives according to a Poisson arrival process.} The Poisson process is used to schedule the delay measurements. The test packets will generally not arrive at Dst according to a Poisson distribution, since they are influenced by the network.

All the singleton Type-P-One-way-Delay metrics in the sequence will have the same values of Src, Dst, and Type-P.

Note also that, given one sample that runs from T_0 to T_f , and given new time values T_0' and T_f' such that $T_0 \leq T_0' \leq T_f' \leq T_f$, the subsequence of the given sample whose time values fall between T_0' and T_f' are also a valid Type-P-One-way-Delay-Poisson-Stream sample.

4.6. Methodologies:

The methodologies follow directly from:

+ the selection of specific times, using the specified Poisson arrival process, and

+ the methodologies discussion already given for the singleton Type-P-One-way-Delay metric.

Care must, of course, be given to correctly handle out-of-order arrival of test packets; it is possible that the Src could send one test packet at TS[i], then send a second one (later) at TS[i+1], while the Dst could receive the second test packet at TR[i+1], and then receive the first one (later) at TR[i].

>>> Editor proposal: Add ref to RFC 4737 Reordering metric above.

4.7. Errors and Uncertainties:

In addition to sources of errors and uncertainties associated with methods employed to measure the singleton values that make up the sample, care must be given to analyze the accuracy of the Poisson process with respect to the wire-times of the sending of the test packets. Problems with this process could be caused by several things, including problems with the pseudo-random number techniques used to generate the Poisson arrival process, or with jitter in the value of Hsource (mentioned above as uncertainty in the singleton delay metric). The Framework document shows how to use the Anderson-Darling test to verify the accuracy of a Poisson process over small time frames. {Comment: The goal is to ensure that test packets are sent "close enough" to a Poisson schedule, and avoid periodic behavior.}

4.8. Reporting the metric:

You MUST report the calibration and context for the underlying singletons along with the stream. (See "Reporting the metric" for Type-P-One-way-Delay.)

5. Some Statistics Definitions for One-way Delay

Given the sample metric Type-P-One-way-Delay-Poisson-Stream, we now offer several statistics of that sample. These statistics are offered mostly to be illustrative of what could be done. See [RFC6703] for additional discussion of statistics that are relevant to different audiences.

5.1. Type-P-One-way-Delay-Percentile

Given a Type-P-One-way-Delay-Poisson-Stream and a percent X between 0% and 100%, the Xth percentile of all the dT values in the Stream. In computing this percentile, undefined values are treated as infinitely large. Note that this means that the percentile could

thus be undefined (informally, infinite). In addition, the Type-P-One-way-Delay-Percentile is undefined if the sample is empty.

Example: suppose we take a sample and the results are:

```
Stream1 = <
<T1, 100 msec>
<T2, 110 msec>
<T3, undefined>
<T4, 90 msec>
<T5, 500 msec>
>
```

Then the 50th percentile would be 110 msec, since 90 msec and 100 msec are smaller and 500 msec and 'undefined' are larger. See Section 11.3 of [1] for computing percentiles.

Note that if the possibility that a packet with finite delay is reported as lost is significant, then a high percentile (90th or 95th) might be reported as infinite instead of finite.

5.2. Type-P-One-way-Delay-Median

Given a Type-P-One-way-Delay-Poisson-Stream, the median of all the dT values in the Stream. In computing the median, undefined values are treated as infinitely large. As with Type-P-One-way-Delay-Percentile, Type-P-One-way-Delay-Median is undefined if the sample is empty.

As noted in the Framework document, the median differs from the 50th percentile only when the sample contains an even number of values, in which case the mean of the two central values is used.

Example: suppose we take a sample and the results are:

```
Stream2 = < <T1, 100 msec> <T2, 110 msec> <T3, undefined> <T4, 90
msec> >
```

Then the median would be 105 msec, the mean of 100 msec and 110 msec, the two central values.

5.3. Type-P-One-way-Delay-Minimum

Given a Type-P-One-way-Delay-Poisson-Stream, the minimum of all the dT values in the Stream. In computing this, undefined values are treated as infinitely large. Note that this means that the minimum could thus be undefined (informally, infinite) if all the dT values are undefined. In addition, the Type-P-One-way-Delay-Minimum is undefined if the sample is empty.

In the above example, the minimum would be 90 msec.

5.4. Type-P-One-way-Delay-Inverse-Percentile

Note: This statistic is deprecated in this version of the memo because of lack of use.

Given a Type-P-One-way-Delay-Poisson-Stream and a time duration threshold, the fraction of all the dT values in the Stream less than or equal to the threshold. The result could be as low as 0% (if all the dT values exceed threshold) or as high as 100%. Type-P-One-way-Delay-Inverse-Percentile is undefined if the sample is empty.

In the above example, the Inverse-Percentile of 103 msec would be 50%.

6. 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 will 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 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. Unlike passive measurements, there can be no release of existing user data.

7. IANA Considerations

This memo makes no requests of IANA.

8. Acknowledgements

Special thanks are due to Vern Paxson of Lawrence Berkeley Labs for his helpful comments on issues of clock uncertainty and statistics. Thanks also to Garry Couch, Will Leland, Andy Scherrer, Sean Shapira, and Roland Wittig for several useful suggestions.

9. References (temporary)

- [1] Paxson, V., Almes, G., Mahdavi, J. and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [2] Almes, G., Kalidindi, S. and M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, September 1999.
- [3] Mills, D., "Network Time Protocol (v3)", RFC 1305, April 1992.
- [4] Mahdavi J. and V. Paxson, "IPPM Metrics for Measuring Connectivity", RFC 2678, September 1999.
- [5] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [6] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [7] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

10. References

10.1. Normative References

- [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.
- [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.
- [RFC5835] Morton, A. and S. Van den Berghe, "Framework for Metric Composition", RFC 5835, April 2010.
- [RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, January 2011.
- [RFC6576] Geib, R., Morton, A., Fardid, R., and A. Steinmitz, "IP Performance Metrics (IPPM) Standard Advancement Testing", BCP 176, RFC 6576, March 2012.
- [RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View",

RFC 6703, August 2012.

10.2. 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.
- [I-D.ietf-ippm-testplan-rfc2679]
Ciavattone, L., Geib, R., Morton, A., and M. Wieser, "Test Plan and Results Supporting Advancement of RFC 2679 on the Standards Track", draft-ietf-ippm-testplan-rfc2679-03 (work in progress), September 2012.
- [RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

Authors' Addresses

Guy Almes
Texas A&M

Phone:
Fax:
Email:
URI:

Sunil Kalidindi
Ixia

Phone:
Fax:
Email:
URI:

Matt Zekauskas
Internet2

Phone:
Fax:
Email: matt@internet2.edu
URI:

Al Morton (editor)
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
Obsoletes: 2680 (if approved)
Intended status: Standards Track
Expires: August 21, 2013

G. Almes
Texas A&M
S. Kalidindi
Ixia
M. Zekauskas
Internet2
A. Morton, Ed.
AT&T Labs
February 17, 2013

A One-Way Loss Metric for IPPM
draft-morton-ippm-2680-bis-00

Abstract

This memo (RFC 2680 bis) defines a metric for one-way loss of packets across Internet paths. It builds on notions introduced and discussed in the IPPM Framework document, RFC 2330; the reader is assumed to be familiar with that document.

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 21, 2013.

Copyright Notice

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

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

Table of Contents

1. Introduction	4
1.1. Motivation	4
1.2. General Issues Regarding Time	5
2. A Singleton Definition for One-way Packet Loss	7
2.1. Metric Name:	7
2.2. Metric Parameters:	7
2.3. Metric Units:	7
2.4. Definition:	7
2.5. Discussion:	7
2.6. Methodologies:	8
2.7. Errors and Uncertainties:	9
2.8. Reporting the metric:	10
2.8.1. Type-P	10
2.8.2. Loss Threshold	11
2.8.3. Calibration Results	11
2.8.4. Path	11
3. A Definition for Samples of One-way Packet Loss	11
3.1. Metric Name:	12
3.2. Metric Parameters:	12
3.3. Metric Units:	12
3.4. Definition:	12
3.5. Discussion:	13
3.6. Methodologies:	14
3.7. Errors and Uncertainties:	14
3.8. Reporting the metric:	14
4. Some Statistics Definitions for One-way Packet Loss	14
4.1. Type-P-One-way-Packet Loss-Average	15
5. Security Considerations	15
6. Acknowledgements	16
7. RFC 2680 bis	16
8. IANA Considerations	17
9. Acknowledgements	17
10. References (temporary)	17
11. References	18
11.1. Normative References	18
11.2. Informative References	19
Authors' Addresses	19

1. Introduction

This memo defines a metric for one-way packet loss across Internet paths. It builds on notions introduced and discussed in the IPPM Framework document, RFC 2330 [1]; the reader is assumed to be familiar with that document.

This memo is intended to be parallel in structure to a companion document for One-way Delay ("A One-way Delay Metric for IPPM") [2]; the reader is assumed to be familiar with that document.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [5]. Although RFC 2119 was written with protocols in mind, the key words are used in this document for similar reasons. They are used to ensure the results of measurements from two different implementations are comparable, and to note instances when an implementation could perturb the network.

The structure of the memo is as follows:

- + A 'singleton' analytic metric, called Type-P-One-way-Packet-Loss, is introduced to measure a single observation of packet transmission or loss.

- + Using this singleton metric, a 'sample', called Type-P-One-way-Packet-Loss-Poisson-Stream, is introduced to measure a sequence of singleton transmissions and/or losses measured at times taken from a Poisson process.

- + Using this sample, several 'statistics' of the sample are defined and discussed.

This progression from singleton to sample to statistics, with clear separation among them, is important.

Whenever a technical term from the IPPM Framework document is first used in this memo, it will be tagged with a trailing asterisk. For example, "term*" indicates that "term" is defined in the Framework.

1.1. Motivation

Understanding one-way packet loss of Type-P* packets from a source host* to a destination host is useful for several reasons:

- + Some applications do not perform well (or at all) if end-to-end loss between hosts is large relative to some threshold value.

- + Excessive packet loss may make it difficult to support certain real-time applications (where the precise threshold of "excessive" depends on the application).
- + The larger the value of packet loss, the more difficult it is for transport-layer protocols to sustain high bandwidths.
- + The sensitivity of real-time applications and of transport-layer protocols to loss become especially important when very large delay-bandwidth products must be supported.

The measurement of one-way loss instead of round-trip loss is motivated by the following factors:

- + In today's Internet, the path from a source to a destination may be different than the path from the destination back to the source ("asymmetric paths"), such that different sequences of routers are used for the forward and reverse paths. Therefore round-trip measurements actually measure the performance of two distinct paths together. Measuring each path independently highlights the performance difference between the two paths which may traverse different Internet service providers, and even radically different types of networks (for example, research versus commodity networks, or ATM versus packet-over-SONET).
- + Even when the two paths are symmetric, they may have radically different performance characteristics due to asymmetric queueing.
- + Performance of an application may depend mostly on the performance in one direction. For example, a file transfer using TCP may depend more on the performance in the direction that data flows, rather than the direction in which acknowledgements travel.
- + In quality-of-service (QoS) enabled networks, provisioning in one direction may be radically different than provisioning in the reverse direction, and thus the QoS guarantees differ. Measuring the paths independently allows the verification of both guarantees.

It is outside the scope of this document to say precisely how loss metrics would be applied to specific problems.

1.2. General Issues Regarding Time

{Comment: the terminology below differs from that defined by ITU-T documents (e.g., G.810, "Definitions and terminology for synchronization networks" and I.356, "B-ISDN ATM layer cell transfer performance"), but is consistent with the IPPM Framework document. In general, these differences derive from the different backgrounds;

the ITU-T documents historically have a telephony origin, while the authors of this document (and the Framework) have a computer systems background. Although the terms defined below have no direct equivalent in the ITU-T definitions, after our definitions we will provide a rough mapping. However, note one potential confusion: our definition of "clock" is the computer operating systems definition denoting a time-of-day clock, while the ITU-T definition of clock denotes a frequency reference.}

Whenever a time (i.e., a moment in history) is mentioned here, it is understood to be measured in seconds (and fractions) relative to UTC.

As described more fully in the Framework document, there are four distinct, but related notions of clock uncertainty:

synchronization*

measures the extent to which two clocks agree on what time it is. For example, the clock on one host might be 5.4 msec ahead of the clock on a second host. {Comment: A rough ITU-T equivalent is "time error".}

accuracy*

measures the extent to which a given clock agrees with UTC. For example, the clock on a host might be 27.1 msec behind UTC. {Comment: A rough ITU-T equivalent is "time error from UTC".}

resolution*

measures the precision of a given clock. For example, the clock on an old Unix host might tick only once every 10 msec, and thus have a resolution of only 10 msec. {Comment: A very rough ITU-T equivalent is "sampling period".}

skew*

measures the change of accuracy, or of synchronization, with time. For example, the clock on a given host might gain 1.3 msec per hour and thus be 27.1 msec behind UTC at one time and only 25.8 msec an hour later. In this case, we say that the clock of the given host has a skew of 1.3 msec per hour relative to UTC, which threatens accuracy. We might also speak of the skew of one clock relative to another clock, which threatens synchronization. {Comment: A rough ITU-T equivalent is "time drift".}

2. A Singleton Definition for One-way Packet Loss

2.1. Metric Name:

Type-P-One-way-Packet-Loss

2.2. Metric Parameters:

- + Src, the IP address of a host
- + Dst, the IP address of a host
- + T, a time

2.3. Metric Units:

The value of a Type-P-One-way-Packet-Loss is either a zero (signifying successful transmission of the packet) or a one (signifying loss).

2.4. Definition:

>>The *Type-P-One-way-Packet-Loss* from Src to Dst at T is 0<< means that Src sent the first bit of a Type-P packet to Dst at wire-time* T and that Dst received that packet.

>>The *Type-P-One-way-Packet-Loss* from Src to Dst at T is 1<< means that Src sent the first bit of a type-P packet to Dst at wire-time T and that Dst did not receive that packet.

2.5. Discussion:

Thus, Type-P-One-way-Packet-Loss is 0 exactly when Type-P-One-way-Delay is a finite value, and it is 1 exactly when Type-P-One-way-Delay is undefined.

The following issues are likely to come up in practice:

+ A given methodology will have to include a way to distinguish between a packet loss and a very large (but finite) delay. As noted by Mahdavi and Paxson [3], simple upper bounds (such as the 255 seconds theoretical upper bound on the lifetimes of IP packets [4]) could be used, but good engineering, including an understanding of packet lifetimes, will be needed in practice. {Comment: Note that, for many applications of these metrics, there may be no harm in treating a large delay as packet loss. An audio playback packet, for example, that arrives only after the playback point may as well have been lost.}

+ If the packet arrives, but is corrupted, then it is counted as lost. {Comment: one is tempted to count the packet as received since corruption and packet loss are related but distinct phenomena. If the IP header is corrupted, however, one cannot be sure about the source or destination IP addresses and is thus on shaky grounds about knowing that the corrupted received packet corresponds to a given sent test packet. Similarly, if other parts of the packet needed by the methodology to know that the corrupted received packet corresponds to a given sent test packet, then such a packet would have to be counted as lost. Counting these packets as lost but packet with corruption in other parts of the packet as not lost would be inconsistent.}

+ If the packet is duplicated along the path (or paths) so that multiple non-corrupt copies arrive at the destination, then the packet is counted as received.

+ If the packet is fragmented and if, for whatever reason, reassembly does not occur, then the packet will be deemed lost.

2.6. Methodologies:

As with other Type-P-* metrics, the detailed methodology will depend on the Type-P (e.g., protocol number, UDP/TCP port number, size, precedence).

Generally, for a given Type-P, one possible methodology would proceed as follows:

+ Arrange that Src and Dst have clocks that are synchronized with each other. The degree of synchronization is a parameter of the methodology, and depends on the threshold used to determine loss (see below).

+ At the Src host, select Src and Dst IP addresses, and form a test packet of Type-P with these addresses.

+ At the Dst host, arrange to receive the packet.

+ At the Src host, place a timestamp in the prepared Type-P packet, and send it towards Dst.

+ If the packet arrives within a reasonable period of time, the one-way packet-loss is taken to be zero.

+ If the packet fails to arrive within a reasonable period of time, the one-way packet-loss is taken to be one. Note that the threshold of "reasonable" here is a parameter of the methodology.

{Comment: The definition of reasonable is intentionally vague, and is intended to indicate a value "Th" so large that any value in the closed interval [Th-delta, Th+delta] is an equivalent threshold for loss. Here, delta encompasses all error in clock synchronization along the measured path. If there is a single value after which the packet must be counted as lost, then we reintroduce the need for a degree of clock synchronization similar to that needed for one-way delay. Therefore, if a measure of packet loss parameterized by a specific non-huge "reasonable" time-out value is needed, one can always measure one-way delay and see what percentage of packets from a given stream exceed a given time-out value. This point is examined in detail in [RFC6703], including analysis preferences to assign undefined delay to packets that fail to arrive with the difficulties emerging from the informal "infinite delay" assignment, and an estimation of an upper bound on waiting time for packets in transit. Further, enforcing a specific constant waiting time on stored singletons of one-way delay is compliant with this specification and may allow the results to serve more than one reporting audience.}

Issues such as the packet format, the means by which Dst knows when to expect the test packet, and the means by which Src and Dst are synchronized are outside the scope of this document. {Comment: We plan to document elsewhere our own work in describing such more detailed implementation techniques and we encourage others to as well.}

2.7. Errors and Uncertainties:

The description of any specific measurement method should include an accounting and analysis of various sources of error or uncertainty. The Framework document provides general guidance on this point.

For loss, there are three sources of error:

- + Synchronization between clocks on Src and Dst.
- + The packet-loss threshold (which is related to the synchronization between clocks).
- + Resource limits in the network interface or software on the receiving instrument.

The first two sources are interrelated and could result in a test packet with finite delay being reported as lost. Type-P-One-way-Packet-Loss is 1 if the test packet does not arrive, or if it does arrive and the difference between Src timestamp and Dst timestamp is greater than the "reasonable period of time", or loss threshold. If the clocks are not sufficiently synchronized, the loss threshold may

not be "reasonable" - the packet may take much less time to arrive than its Src timestamp indicates. Similarly, if the loss threshold is set too low, then many packets may be counted as lost. The loss threshold must be high enough, and the clocks synchronized well enough so that a packet that arrives is rarely counted as lost. (See the discussions in the previous two sections.)

Since the sensitivity of packet loss measurement to lack of clock synchronization is less than for delay, we refer the reader to the treatment of synchronization errors in the One-way Delay metric [2] for more details.

The last source of error, resource limits, cause the packet to be dropped by the measurement instrument, and counted as lost when in fact the network delivered the packet in reasonable time.

The measurement instruments should be calibrated such that the loss threshold is reasonable for application of the metrics and the clocks are synchronized enough so the loss threshold remains reasonable.

In addition, the instruments should be checked to ensure the that the possibility a packet arrives at the network interface, but is lost due to congestion on the interface or to other resource exhaustion (e.g., buffers) on the instrument is low.

2.8. Reporting the metric:

The calibration and context in which the metric is measured MUST be carefully considered, and SHOULD always be reported along with metric results. We now present four items to consider: Type-P of the test packets, the loss threshold, instrument calibration, and the path traversed by the test packets. This list is not exhaustive; any additional information that could be useful in interpreting applications of the metrics should also be reported (see [RFC6703] for extensive discussion of reporting considerations for different audiences).

2.8.1. Type-P

As noted in the Framework document [1], the value of the metric may depend on the type of IP packets used to make the measurement, or "Type-P". The value of Type-P-One-way-Delay could change if the protocol (UDP or TCP), port number, size, or arrangement for special treatment (e.g., IP precedence or RSVP) changes. The exact Type-P used to make the measurements MUST be accurately reported.

2.8.2. Loss Threshold

The threshold (or methodology to distinguish) between a large finite delay and loss MUST be reported.

2.8.3. Calibration Results

The degree of synchronization between the Src and Dst clocks MUST be reported. If possible, possibility that a test packet that arrives at the Dst network interface is reported as lost due to resource exhaustion on Dst SHOULD be reported.

2.8.4. Path

Finally, the path traversed by the packet SHOULD be reported, if possible. In general it is impractical to know the precise path a given packet takes through the network. The precise path may be known for certain Type-P on short or stable paths. If Type-P includes the record route (or loose-source route) option in the IP header, and the path is short enough, and all routers* on the path support record (or loose-source) route, then the path will be precisely recorded. This is impractical because the route must be short enough, many routers do not support (or are not configured for) record route, and use of this feature would often artificially worsen the performance observed by removing the packet from common-case processing. However, partial information is still valuable context. For example, if a host can choose between two links* (and hence two separate routes from Src to Dst), then the initial link used is valuable context. {Comment: For example, with Merit's NetNow setup, a Src on one NAP can reach a Dst on another NAP by either of several different backbone networks.}

3. A Definition for Samples of One-way Packet Loss

Given the singleton metric Type-P-One-way-Packet-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 T. The means for defining the values of T is to select a beginning time T_0 , a final time T_f , and an average rate λ , then define a pseudo-random Poisson process of rate λ , whose values fall between T_0 and T_f . The time interval between successive values of T will then average $1/\lambda$.

{Comment: Note that Poisson sampling is only one way of defining a sample. Poisson has the advantage of limiting bias, but other methods of sampling might be appropriate for different situations.

We encourage others who find such appropriate cases to use this general framework and submit their sampling method for standardization.}

>>> Editor proposal: Add ref to RFC 3432 Periodic sampling above.

3.1. Metric Name:

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

3.2. 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

3.3. Metric Units:

A sequence of pairs; the elements of each pair are:

- + T, a time, and
- + L, either a zero or a one

The values of T in the sequence are monotonic increasing. Note that T would be a valid parameter to Type-P-One-way-Packet-Loss, and that L would be a valid value of Type-P-One-way-Packet-Loss.

3.4. Definition:

Given T0, Tf, and lambda, we compute a pseudo-random Poisson process beginning at or before T0, with average arrival rate lambda, and ending at or after Tf. Those time values greater than or equal to T0 and less than or equal to Tf are then selected. At each of the times in this process, we obtain the value of Type-P-One-way-Packet-Loss at this time. The value of the sample is the sequence made up of the resulting <time, loss> pairs. If there are no such pairs, the sequence is of length zero and the sample is said to be empty.

3.5. Discussion:

The reader should be familiar with the in-depth discussion of Poisson sampling in the Framework document [1], which includes methods to compute and verify the pseudo-random Poisson process.

We specifically do not constrain the value of λ , except to note the extremes. If the rate is too large, then the measurement traffic will perturb the network, and itself cause congestion. If the rate is too small, then you might not capture interesting network behavior. {Comment: We expect to document our experiences with, and suggestions for, λ elsewhere, culminating in a "best current practices" document.}

Since a pseudo-random number sequence is employed, the sequence of times, and hence the value of the sample, is not fully specified. Pseudo-random number generators of good quality will be needed to achieve the desired qualities.

The sample is defined in terms of a Poisson process both to avoid the effects of self-synchronization and also capture a sample that is statistically as unbiased as possible. The Poisson process is used to schedule the delay measurements. The test packets will generally not arrive at Dst according to a Poisson distribution, since they are influenced by the network.

{Comment: there is, of course, no claim that real Internet traffic arrives according to a Poisson arrival process.}

It is important to note that, in contrast to this metric, loss rates observed by transport connections do not reflect unbiased samples. For example, TCP transmissions both (1) occur in bursts, which can induce loss due to the burst volume that would not otherwise have been observed, and (2) adapt their transmission rate in an attempt to minimize the loss rate observed by the connection.}

All the singleton Type-P-One-way-Packet-Loss metrics in the sequence will have the same values of Src, Dst, and Type-P.

Note also that, given one sample that runs from T_0 to T_f , and given new time values T_0' and T_f' such that $T_0 \leq T_0' \leq T_f' \leq T_f$, the subsequence of the given sample whose time values fall between T_0' and T_f' are also a valid Type-P-One-way-Packet-Loss-Poisson-Stream sample.

3.6. Methodologies:

The methodologies follow directly from:

- + the selection of specific times, using the specified Poisson arrival process, and

- + the methodologies discussion already given for the singleton Type-P-One-way-Packet-Loss metric.

Care must be given to correctly handle out-of-order arrival of test packets; it is possible that the Src could send one test packet at TS[i], then send a second one (later) at TS[i+1], while the Dst could receive the second test packet at TR[i+1], and then receive the first one (later) at TR[i].

>>> Editor proposal: Add ref to RFC 4737 Reordering metric above.

3.7. Errors and Uncertainties:

In addition to sources of errors and uncertainties associated with methods employed to measure the singleton values that make up the sample, care must be given to analyze the accuracy of the Poisson arrival process of the wire-times of the sending of the test packets. Problems with this process could be caused by several things, including problems with the pseudo-random number techniques used to generate the Poisson arrival process. The Framework document shows how to use the Anderson-Darling test verify the accuracy of the Poisson process over small time frames. {Comment: The goal is to ensure that the test packets are sent "close enough" to a Poisson schedule, and avoid periodic behavior.}

3.8. Reporting the metric:

The calibration and context for the underlying singletons MUST be reported along with the stream. (See "Reporting the metric" for Type-P-One-way-Packet-Loss.)

4. Some Statistics Definitions for One-way Packet Loss

Given the sample metric Type-P-One-way-Packet-Loss-Poisson-Stream, we now offer several statistics of that sample. These statistics are offered mostly to be illustrative of what could be done. See [RFC6703] for additional discussion of statistics that are relevant to different audiences.

4.1. Type-P-One-way-Packet Loss-Average

Given a Type-P-One-way-Packet-Loss-Poisson-Stream, the average of all the L values in the Stream. In addition, the Type-P-One-way-Packet-Loss-Average is undefined if the sample is empty.

Example: suppose we take a sample and the results are:

```
Stream1 = <  
  <T1, 0>  
  <T2, 0>  
  <T3, 1>  
  <T4, 0>  
  <T5, 0>  
>
```

Then the average would be 0.2.

Note that, since healthy Internet paths should be operating at loss rates below 1% (particularly if high delay-bandwidth products are to be sustained), the sample sizes needed might be larger than one would like. Thus, for example, if one wants to discriminate between various fractions of 1% over one-minute periods, then several hundred samples per minute might be needed. This would result in larger values of lambda than one would ordinarily want.

Note that although the loss threshold should be set such that any errors in loss are not significant, if the possibility that a packet which arrived is counted as lost due to resource exhaustion is significant compared to the loss rate of interest, Type-P-One-way-Packet-Loss-Average will be meaningless.

5. 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 will 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 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. Unlike passive measurements, there can be no release of existing user data.

6. Acknowledgements

Thanks are due to Matt Mathis for encouraging this work and for calling attention on so many occasions to the significance of packet loss.

Thanks are due also to Vern Paxson for his valuable comments on early drafts, and to Garry Couch and Will Leland for several useful suggestions.

7. RFC 2680 bis

The following text constitutes RFC 2680 bis proposed for advancement on the IETF Standards Track.

[I-D.ietf-ippm-testplan-rfc2680] (now approved) provides the test plan and results supporting [RFC2680] advancement along the standards track, according to the process in [RFC6576]. The conclusions of [I-D.ietf-ippm-testplan-rfc2680] list four minor modifications for inclusion:

1. Section 6.2.3 of [I-D.ietf-ippm-testplan-rfc2680] asserts that the assumption of post-processing to enforce a constant waiting time threshold is compliant, and that the text of the RFC should be revised slightly to include this point (see the last list item of section 2.6, above).
2. Section 6.5 of [I-D.ietf-ippm-testplan-rfc2680] indicates that Type-P-One-way-Packet-Loss-Average statistic is more commonly called Packet Loss Ratio, so it is re-named in RFC2680bis (this small discrepancy does not affect candidacy for advancement) (see section 4.1, above).
3. The IETF has reached consensus on guidance for reporting metrics in [RFC6703], and this memo should be referenced in RFC2680bis to incorporate recent experience where appropriate (see the last list item of section 2.6, section 2.8, and section 4 above).
4. There are currently two errata with status "Verified" and "Held for document update" for [RFC2680], and it appears these minor revisions should be incorporated in RFC2680bis (see section 1 and section 2.7).

A small number of updates to the [RFC2680] text have been proposed (by the current Editor) in the text, principally to reference key IPPM RFCs that were approved after [RFC2680] (see sections 3 and 3.6, above).

8. IANA Considerations

This memo makes no requests of IANA.

9. Acknowledgements

Special thanks are due to Vern Paxson of Lawrence Berkeley Labs for his helpful comments on issues of clock uncertainty and statistics. Thanks also to Garry Couch, Will Leland, Andy Scherrer, Sean Shapira, and Roland Wittig for several useful suggestions.

10. References (temporary)

- [1] Paxson, V., Almes, G., Mahdavi, J. and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.
- [2] Almes, G., Kalidindi, S. and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, September 1999.

- [3] Mahdavi, J. and V. Paxson, "IPPM Metrics for Measuring Connectivity", RFC 2678, September 1999.
- [4] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [5] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [6] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

11. References

11.1. Normative References

- [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.
- [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.

- [RFC5835] Morton, A. and S. Van den Berghe, "Framework for Metric Composition", RFC 5835, April 2010.
- [RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, January 2011.
- [RFC6576] Geib, R., Morton, A., Fardid, R., and A. Steinmitz, "IP Performance Metrics (IPPM) Standard Advancement Testing", BCP 176, RFC 6576, March 2012.
- [RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View", RFC 6703, August 2012.

11.2. 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.
- [I-D.ietf-ippm-testplan-rfc2680] Ciavattone, L., Geib, R., Morton, A., and M. Wieser, "Test Plan and Results for Advancing RFC 2680 on the Standards Track", draft-ietf-ippm-testplan-rfc2680-01 (work in progress), January 2013.
- [RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.

Authors' Addresses

Guy Almes
Texas A&M

Phone:
Fax:
Email:
URI:

Sunil Kalidindi
Ixia

Phone:
Fax:
Email:
URI:

Matt Zekauskas
Internet2

Phone:
Fax:
Email: matt@internet2.edu
URI:

Al Morton (editor)
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
Updates: 5357 (if approved)
Intended status: Standards Track
Expires: March 7, 2013

A. Morton
L. Ciavattone
AT&T Labs
September 3, 2012

TWAMP Burst Rate Measurement Features
draft-morton-ippm-twamp-rate-02

Abstract

This memo describes two rate-measurement features for the core specification of TWAMP - the Two-Way Active Measurement Protocol: an optional capability where the reflector host responds with a controlled burst of test-session packets (instead of a single packet), and an optional test mode that requires the responder to measure a burst of test packets and communicate the results in truncated packet(s). Both features add the ability to control packet size in the tested direction, enabling asymmetrical packet size testing. There is an open question on using TCP transport instead of UDP.

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 March 7, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	4
1.1. Question on Transport Protocol Selection	4
2. Purpose and Scope	5
3. TWAMP Control Extensions	6
3.1. Connection Setup with New Features	6
3.2. Burst Generation: Request-TW-Session Packet Format	6
3.3. Burst Measurement: Request-TW-Session Packet Format	8
3.4. Burst Gen and Meas: Accept Session Packet Format	9
3.5. Burst Gen and Meas: Stopping Test Sessions	9
3.6. Additional considerations	9
4. Burst Generation in TWAMP Test	10
4.1. Sender Behavior	10
4.1.1. Packet Timings	10
4.1.2. Packet Formats and Contents	10
4.2. Reflector Behavior	11
4.2.1. Session-Reflector Burst Packet Format and Contents	11
5. Burst Measurement in TWAMP Test	13
5.1. Sender Behavior	13
5.1.1. Packet Timings	13
5.1.2. Packet Formats and Contents	13
5.2. Reflector Behavior	14
5.2.1. Session-Reflector Burst Measurement Response Packet Format and Contents	15
6. Special Case of One-packet Bursts	17
7. Security Considerations	17
8. IANA Considerations	17
8.1. Registry Specification	17
8.2. Registry Contents	18
9. Acknowledgements	18
10. References	18
10.1. Normative References	18
10.2. Informative References	19

Authors' Addresses

19

1. Introduction

TWAMP - the Two-Way Active Measurement Protocol [RFC5357] is an extension of the One-way Active Measurement Protocol, OWAMP [RFC4656]. The TWAMP specification gathered wide review as it was deployed, resulting in recommendations for new features.

This memo describes two closely-related features for TWAMP. When measuring packet delivery rate to end-systems, unique control and measurement capabilities become useful, especially when the path tested includes asymmetrical link speeds (as are often deployed in consumer Internet access services).

One feature is the OPTIONAL capability for the responder host to return a controlled burst of test-session packets (instead of a single packet).

Another is an optional sender packet format that requires the responder to measure a burst of test packets and communicate the results in a single packet.

Both features add the ability to control packet size in each direction, enabling asymmetrical packet size testing. Although TWAMP [RFC5357] recommends padding/truncation to achieve symmetrical sizes (to compensate for the Session-Reflector's larger test packet header), these features configure test packet sizes when the test session is requested using the TWAMP-Control protocol.

We note that [draft-baillargeon-ippm-twamp-value-added-octets-01.txt] addresses a similar measurement problem, but places different requirements on the reflector host and does not include the asymmetrical size aspect.

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

Throughout this memo, the bits marked MBZ (Must Be Zero) MUST be set to zero by senders and MUST be ignored by receivers. Also, the HMAC (Hashed Message Authentication Code) MUST be calculated as defined in Section 3.2 of [RFC4656].

1.1. Question on Transport Protocol Selection

An open question in the IPPM problem statement draft is whether testing with TCP transport protocol is a needed capability. The current TWAMP test protocol capability is limited to UDP transport.

What are the implications of specifying a TWAMP-Test capability with TCP transport headers, with or without TCP flow control?

This is clearly a topic where coordination is required between the testing sender and receiver devices. It could be specified as an independent TWAMP feature, although it is clearly related to the features described here.

2. Purpose and Scope

The purpose of this memo is to define two OPTIONAL closely-related features for TWAMP [RFC5357]. The features enhance the TWAMP responder's capabilities to perform a simple operations on test packets, and the capability to demand asymmetrical size TWAMP-Test packets.

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

- o Burst Generation: the capability of the Session-Reflector to generate a burst of packets for return to the Session-Sender, and the corresponding TWAMP-Control messages to activate the capability between compliant hosts.
- o Burst Measurement: the capability of the Session-Reflector to measure a burst of packets from the Session-Sender, report the key information (receive timestamps) in the response packet(s), and the corresponding TWAMP-Control messages to activate the capability between compliant hosts.
- o Asymmetrical Size: the capability to ensure that TWAMP-Test protocol uses a specific packet size in each direction. This feature is combined with the Burst features, and essentially adds a third simple capability when the Burst size = 1.

This memo extends the modes of operation through assignment of two new values 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 two new values correspond to the two features defined in this memo.

When the Server and Control-Client have agreed to use the Burst Generation 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.

When the Server and Control-Client have agreed to use the Burst Measurement 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.

3. 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. The following sections describe two such extensions.

3.1. Connection Setup with New Features

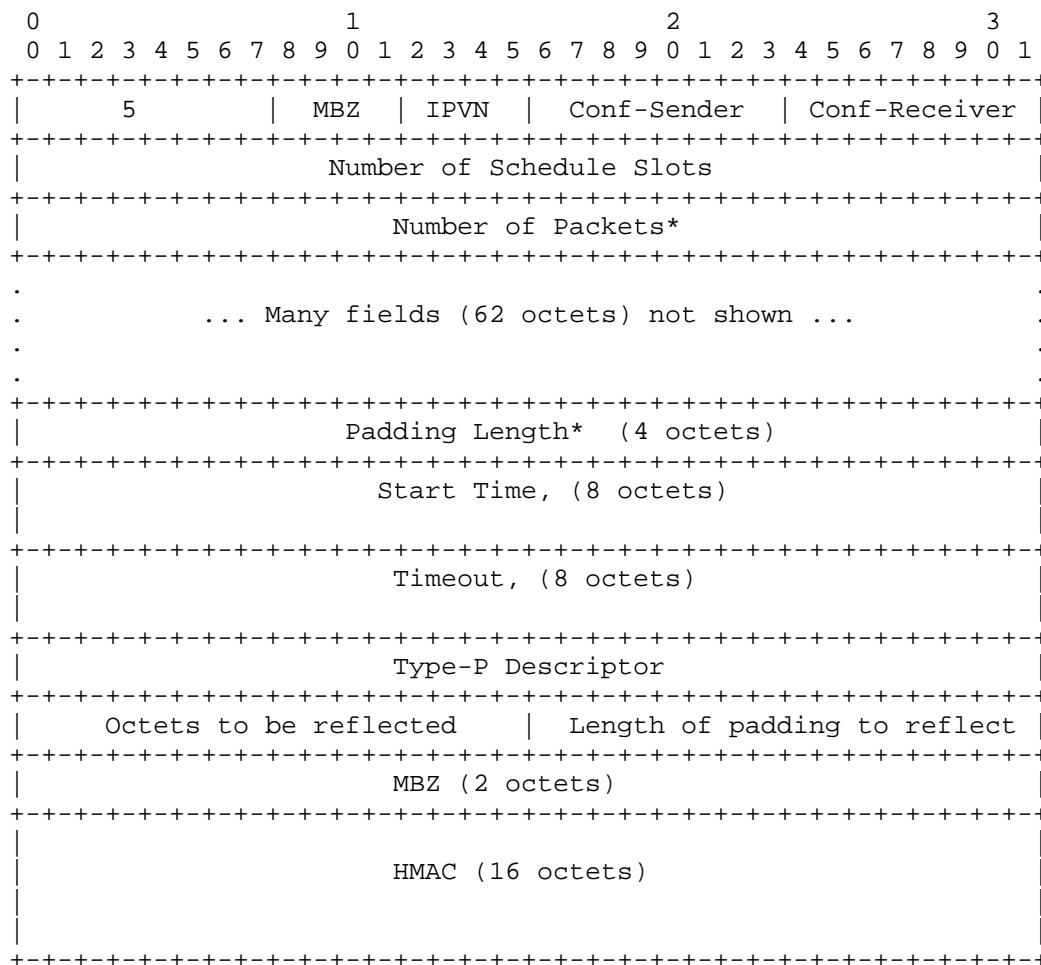
TWAMP connection establishment follows the procedure defined in section 3.1 of [RFC4656] and section 3.1 of [RFC5357]. The new features require two new bit positions (and values). See the IANA section for details on the assigned values and bit positions.

The Server sets one or both of the new bit positions in the Modes Field of the Server Greeting message to indicate its capabilities and willingness to operate in either of these modes if desired.

If the Control-Client intends to operate all test sessions invoked with this control connection using one of the new modes, it MUST set the Mode Field bit corresponding to each function in the Setup Response message. With this and other extensions, the Control-Client MAY set multiple Mode Field bits in the Setup Response message, but these new features are mutually exclusive, and MUST NOT be used together.

3.2. Burst Generation: Request-TW-Session Packet Format

The bits designated for the Burst Generation feature in the Request-TW-Session command are as shown in the packet format below.



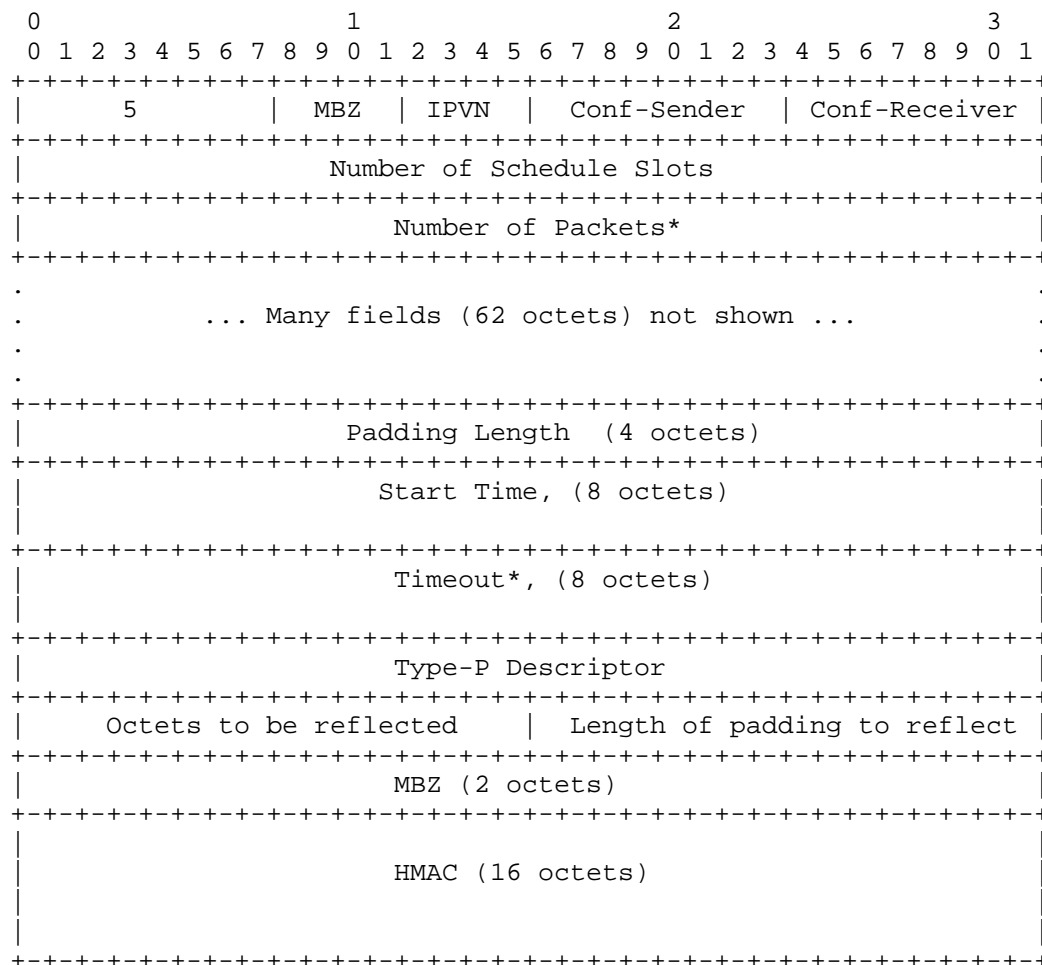
* = re-interpreted field

Two re-interpreted fields appear in the Request-TW-Session command when using Burst Generation mode:

1. Number of Packets: In this mode, re-interpreted as the number of packets that the Session-Reflector MUST generate in each Burst.
2. Packet Padding Length: In the mode, re-interpreted as the number of octets the Session-Reflector MUST append to the Test packet header of each packet it generates as part of the burst. The Session-Reflector MUST NOT assume that the Session-Sender will use any packet padding, and MUST be prepared to generate the padding itself.

3.3. Burst Measurement: Request-TW-Session Packet Format

The bits designated for the Burst Generation feature in the Request-TW-Session command are as shown in the packet format below.



* = re-interpreted field

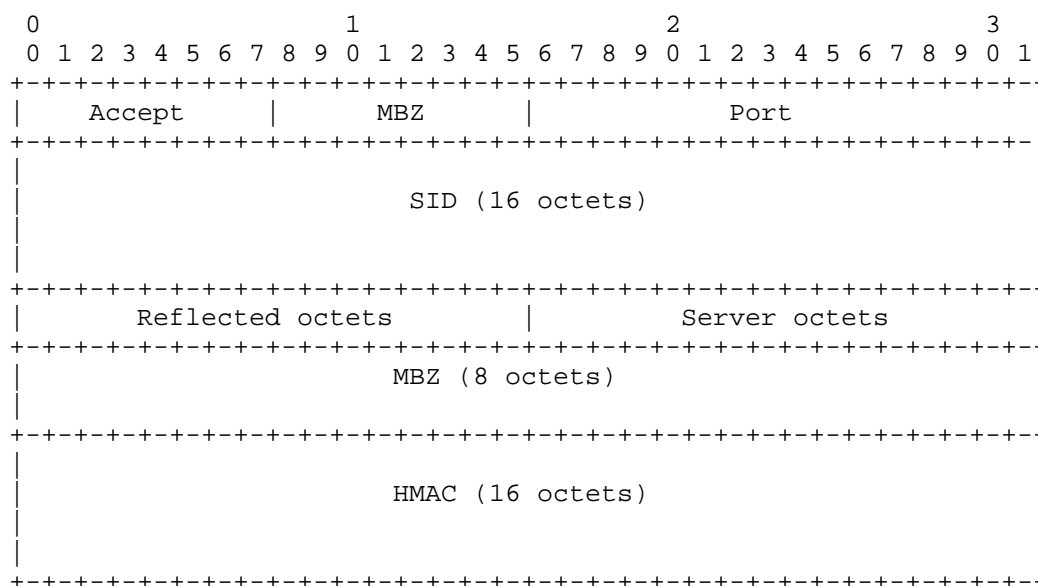
Two re-interpreted fields appear in the Request-TW-Session command when using Burst Measurement mode:

1. Number of Packets: In this mode, re-interpreted as the number of packets that the Session-Reflector MUST expect to measure as part of each Burst.

2. Timeout: In this mode, re-interpreted as the time to wait for all packets in a burst to arrive, expressed in the existing timestamp format used in TWAMP and OWAMP. In the case of lost packets, the Session-Reflector is commanded to wait through this time-out for packets in a burst to arrive.

3.4. Burst Gen and Meas: Accept Session Packet Format

The Accept Session command for the Burst feature is as shown in the packet format below (assuming the Reflect Octets feature is also in use).



3.5. Burst Gen and Meas: Stopping Test Sessions

The Control-Client SHALL stop in-progress test sessions using any standardized methods, including section 3.8 of [RFC5357] or the optional capability of [RFC5938].

3.6. Additional considerations

The value of the Modes Field sent by the Server in the Server Greeting message is the bit-wise OR of the mode values that it is willing to support during this session.

We note that Burst Generation and Measurement features are incompatible with each other, and with the Symmetrical Size feature

described in [RFC6038], and MUST NOT be used in combination with those features.

With the publication of this memo as an RFC, the last 9 bit positions of the Modes 32-bit Field are used. A Control-Client conforming to this extension of [RFC5357] MAY ignore the values in the higher bits of the Modes Field, or it MAY support other features that are communicated in those bit positions. The other bits are available for future protocol extensions.

4. Burst Generation in TWAMP Test

The TWAMP test protocol is similar to the OWAMP [RFC4656] test protocol with the exception that the Session-Reflector transmits test packets to the Session-Sender in response to each test packet it receives. The Burst Generation feature modifies the behavior of TWAMP section 4[RFC5357]. This mode requires the Session-Sender to send a Burst-Initiation packet, and the Session-Reflector generates test session packets according to the configuration agreed using the TWAMP-Control protocol.

4.1. Sender Behavior

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

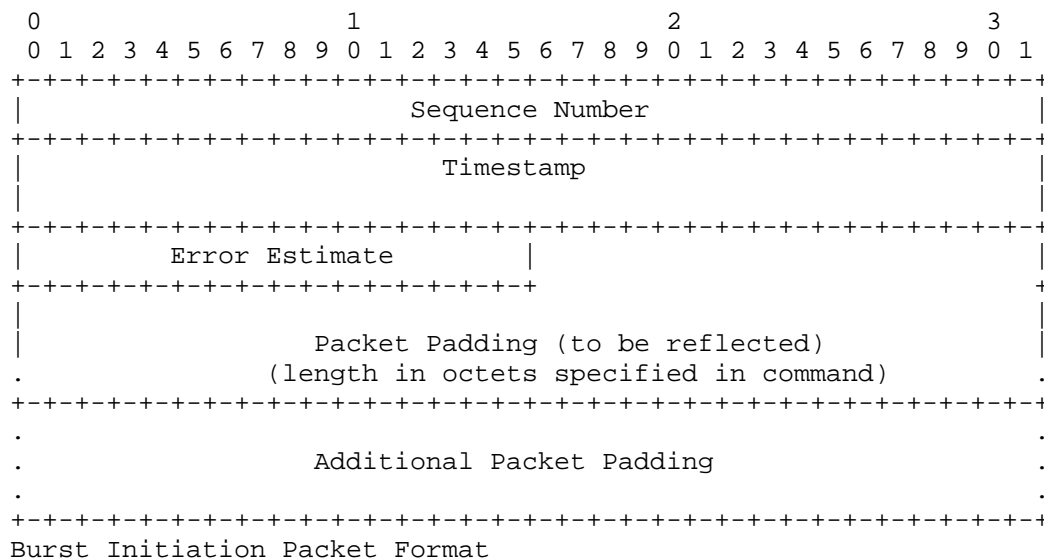
4.1.1. Packet Timings

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

4.1.2. Packet Formats and Contents

The Session-Sender packet format and content follow the same procedure and guidelines as defined in section 4.1.2 of [RFC4656] (as indicated in section 4.1.2 of TWAMP [RFC5357]).

This mode uses the original TWAMP-Test Packet Padding Field (see section 4.1.2 of [RFC4656]), or can be used with Reflect Octets feature as shown below for unauthenticated mode:



The Sequence Number, Timestamp, and Error Estimate fields are the same as specified in section 4.1.2 of [RFC4656] in OWAMP.

We note that the format of the Burst Initiation packet has not been changed from the usual Session-Sender test packet format, to simplify adoption.

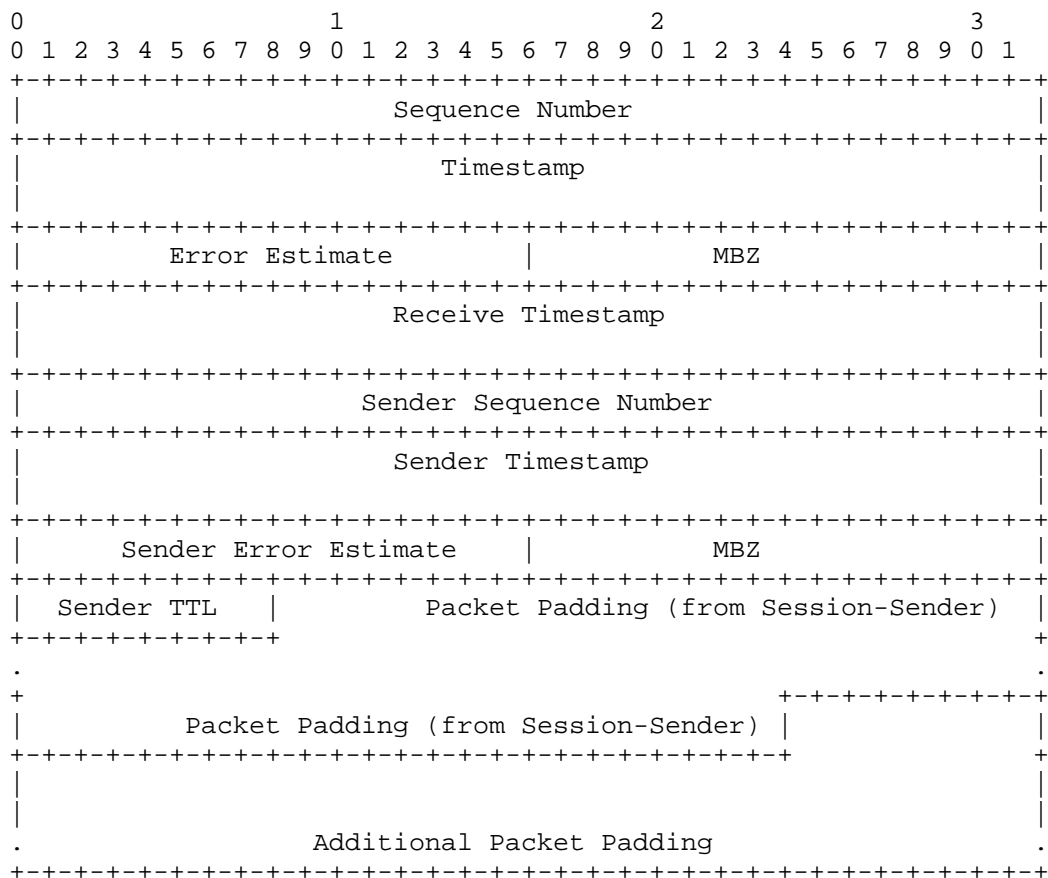
4.2. Reflector Behavior

The TWAMP Reflector differs significantly from the procedures and guidelines in section 4.2 of [RFC5357]. The following new functions MUST be performed:

- o Recognition of the function of the Burst Initiation Packet used in this mode.
- o Generation of the required burst of test session packets, according to the configuration agreed in Request-TW-Session command, with the agreed number of packets in each burst and size of each packet in the burst.

4.2.1. Session-Reflector Burst Packet Format and Contents

The Burst Generation feature retains the usual Reflector packet fields, as shown below. When the Burst Generation mode is selected, the Session-Reflector SHALL use the following TWAMP-Test Packet Format in Unauthenticated mode (shown with Reflect Octets feature activated):



Section 4.2.1 of [RFC5357] describes the above fields as used in TWAMP, with one exception.

The Sequence Number field SHALL indicate the sequence number of each packet sent throughout the test session. The Sequence Number SHALL be increased by 1 for each packet. The initial Sequence Number SHALL be 0.

When one burst is complete, the Sequence Numbers SHALL continue to increment by 1 in the packets generated in response to the next burst.

The total Packet Padding octets SHALL have the length specified in the TWAMP-Control request for the appropriate test session. The Session-Reflector MAY need to generate its own packet padding, if the Burst Request packet does not include this field (or contains insufficient padding).

In any case, the Session-Reflector MAY re-use the Sender's Packet Padding (since the requirements for padding generation are the same for each) when possible.

The Session-Reflector SHALL send a series of TWAMP-Test Packets in response to reception of the Burst Initiation Packet, according to the configuration agreed in the Request-TW-Session command (number of packets and padding), and as immediately as possible. The Session-Reflector SHALL send all packets in a burst as close to back-to-back as possible (recognizing that lower layers may have spacing requirements that take precedence).

5. Burst Measurement in TWAMP Test

The Burst Measurement feature modifies the behavior of TWAMP section 4[RFC5357]. This mode requires the Session-Sender to send a Burst of test packets, and the Session-Reflector measures the burst of packets and reports the results in the Burst Response packet format(s), as described below.

5.1. Sender Behavior

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

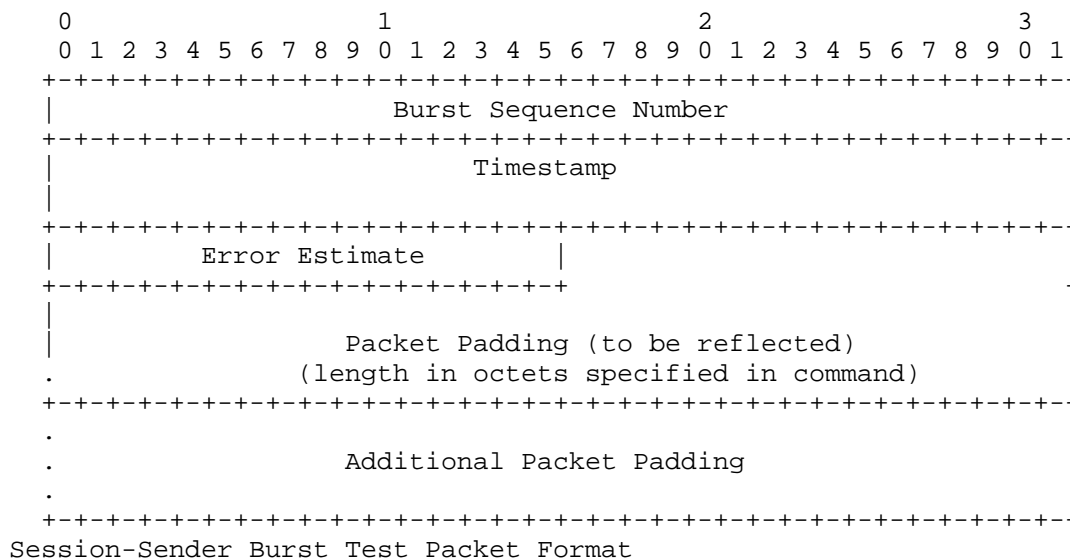
5.1.1. Packet Timings

The Session-Sender SHALL send all packets in a burst as close to back-to-back as possible (recognizing that lower layers may have spacing requirements that take precedence).

5.1.2. Packet Formats and Contents

The Session-Sender packet format and content SHALL comply with that defined in section 4.1.2 of [RFC4656] (as indicated in section 4.1.2 of TWAMP [RFC5357]).

This mode uses the original TWAMP-Test Packet Padding Field (see section 4.1.2 of [RFC4656]), or can be used with Reflect Octets feature as shown below for unauthenticated mode:



The Burst Sequence Number field SHALL indicate the number of each burst. The Burst Sequence Number SHALL be increased by 1 for each burst, and remain the same for each packet in a burst. The initial number SHALL be 0.

When one burst is complete, the Burst Sequence Number used in the all packets of the next burst SHALL be increased by 1.

5.2. Reflector Behavior

The TWAMP Reflector differs slightly from the procedures and guidelines in section 4.2 of [RFC5357]. The following new functions MUST be performed:

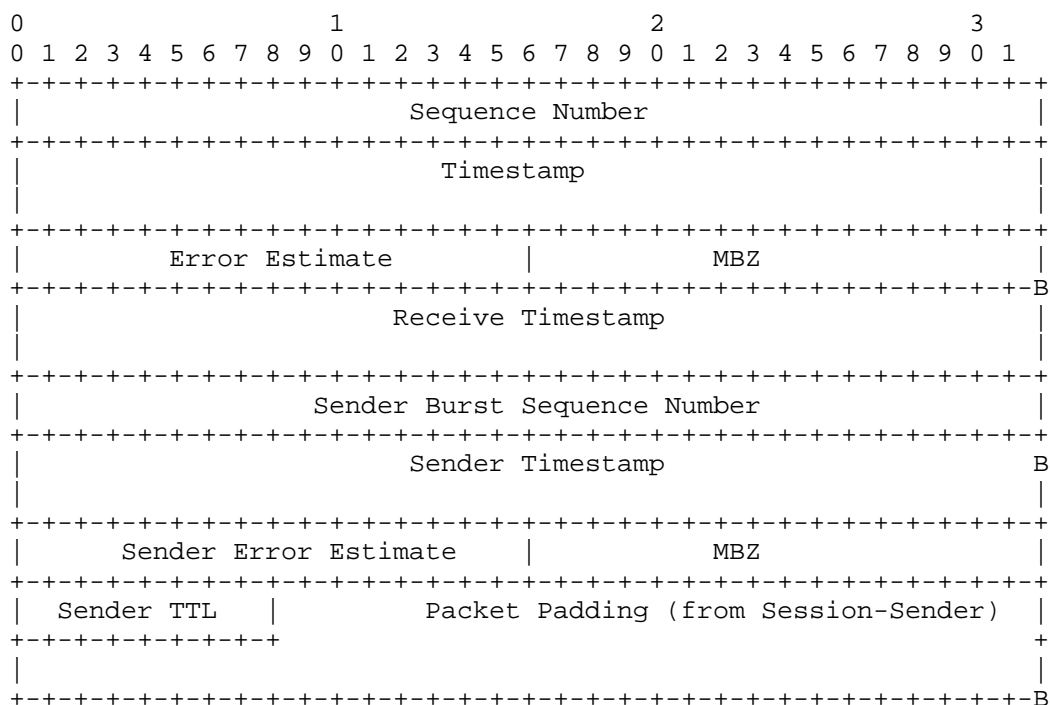
- o Recognition of the function of the Session-Sender Burst Test Packet Format used in this mode.
- o Processing the required bursts of test session packets, according to the configuration agreed in Request-TW-Session command, with the agreed length of the burst in packets and size of each packet in the burst, and the agreed Burst Time-out.
- o Response with an abbreviated Session-Reflector test packet as described below. For discussion, we will call this the 1-to-1 response.
- o OR - Response with the new Burst Measurement Response packet described below. For discussion, we will call this the

accumulated response.

We seek feedback from the IPPM working group on which of these two alternatives is preferable.

5.2.1. Session-Reflector Burst Measurement Response Packet Format and Contents

The Burst Measurement feature specifies a standard Session-Reflector packet to communicate the results, as shown below. When the Burst measurement mode is selected, the Session-Sender SHALL use the following Burst Measurement Response packet Format in Unauthenticated mode (shown with Reflect Octets feature also in use):



Session-Reflector Measurement Packet (1-to-1 response)

Section 4.2.1 of [RFC5357] describes the fields in the 1-to-1 response packet above; they are the same as used in TWAMP. The main difference is that Packet Padding SHALL be truncated on a 16 octet-word boundary, returning the minimum information to the Session-Sender.

All Timestamps SHALL be formatted according to the precedent set in section 4.1.2 of [RFC4656], which is to use [RFC1305] (and updated

version), as follows:

"The first 32 bits represent the unsigned integer number of seconds elapsed since 0h on 1 January 1900; the next 32 bits represent the fractional part of a second that has elapsed since then."

The Session-Reflector MUST truncate the Sender's Packet Padding, unless the Reflect Octets feature is also active in which case the Session Reflector MAY re-use the Sender's Packet Padding (since the requirements for padding generation are the same for each) to reach a word boundary.

The Sender Timestamp field SHALL have the sender's timestamp from each packet received in the burst.

In 1-to-1 response mode, the Session-Reflector SHALL send a Session-Reflector Measurement Packet in response to every Session-Sender packet received, and as immediately as possible.

=====

In the accumulated response alternative, the Session-Reflector creates and holds all packet headers described above in a buffer, and sends them all at once in a single Session-Reflector test packet. The length of the burst and the path MTU MUST be coordinated to avoid fragmentation.

The first Session-Sender packet to arrive with a previously unseen Burst Sequence Number SHALL be designated as the "First" packet in that burst, and its timestamp is used in processing below.

As subsequent packets arrive, Session-Reflector SHALL:

- o Maintain a count of packets with the same Burst Sequence Number (one burst).
- o Time stamp each packet as it arrives and store the time stamp in a response packet structure with all fields complete, as in the 1-to-1 alternative.

When

- o The count of packets with the same Burst Sequence Number equals the agreed Burst Length, OR
- o The agreed Timeout expires (computed by a the time to the "First" Packet Timestamp), OR

- o The Burst Sequence Number increases from previous packets (indicating a new Burst is in progress),

then the current burst is determined to be complete.

When the Burst is complete, the Session-Reflector SHALL terminate the current burst processing as described above and send the Burst Measurement Response Packet to the Session-Sender as immediately as possible.

In Accumulated Response, the Burst Measurement Response Packet is a single packet with the concatenation of all previously-generated response packet formats in the information field.

6. Special Case of One-packet Bursts

When the Number of Packets field in the Request-TW-Session command equals 1, then the Burst Generation and Measurement modes are reduced to test sessions with controlled, asymmetrical packet sizes. A minimal size packet travels in one direction, and the measured direction uses a packet with all Packet Padding specified in the Request-TW-Session command.

7. Security Considerations

These extended modes of operation do not appear to permit any new attacks on hosts communicating with core TWAMP [RFC5357].

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 two modes 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 Set-up 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

TWAMP Modes Registry is recommended to be augmented as follows:

Value	Description	Semantics Definition
xxx	Burst Generation Capability	this memo, section 3.1 new bit position (X)
yyy	Burst Measurement	this memo, section 3.1 new bit position (Y)

>>>IANA: change xxx, yyy, X, Y, and RFC???? to the assigned values

The suggested values are

X=7, xxx=128

Y=8, yyy=256 <<<<

9. Acknowledgements

The authors thank folks for review and comment.

10. References

10.1. Normative References

- [RFC1305] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation", RFC 1305, March 1992.
- [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.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J.

Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, October 2008.

[RFC5618] Morton, A. and K. Hedayat, "Mixed Security Mode for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5618, August 2009.

[RFC5938] Morton, A. and M. Chiba, "Individual Session Control Feature for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5938, August 2010.

[RFC6038] Morton, A. and L. Ciavattone, "Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features", RFC 6038, October 2010.

10.2. Informative References

Authors' Addresses

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/>

Len Ciavattone
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 1239
Fax:
Email: lencia@att.com
URI:

