Network Working Group
Internet-Draft
Obsoletes: 2679 (if approved)
Intended status: Standards Track
Expires: April 26, 2015

G. Almes
Texas A&M
S. Kalidindi
Ixia
M. Zekauskas
Internet2
A. Morton, Ed.
AT&T Labs
October 23, 2014

A One-Way Delay Metric for IPPM
draft-ietf-ippm-2679-bis-00

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

Table of Contents

1. RFC 2679 bis

The following text constitutes RFC 2769 bis proposed for advancement on the IETF Standards Track. This section tracks the changes from [RFC2679].

[RFC6808] provides the test plan and results supporting [RFC2679] advancement along the standards track, according to the process in [RFC6576]. The conclusions of [RFC6808] list four minor modifications:

1. Section 6.2.3 of [RFC6808] 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 [RFC6808] 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 number of updates to the [RFC2679] text have been implemented in the text below, to reference key IPPM RFCs that were approved after

[RFC2679], and to address comments on the IPPM mailing list describing current conditions and experience.

1. Near the end of section 2.1, update of a network example using ATM and clarification of TCP's affect on queue occupation and importance of one-way delay measurement.

2. Explicit inclusion of the maximum waiting time input parameter in section 3.2 and 4.2, reflecting recognition of this parameter in more recent RFCs and ITU-T Recommendation Y.1540.

3. Addition of reference to RFC6703 in the discussion of packet life time and application timeouts in section 3.5.

4. Addition of reference to the default requirement (that packets be standard-formed) from RFC2330 as a new list item in section 3.5.

5. GPS-based NTP experience replaces "to be tested" in section 3.5.

6. Added parenthetical guidance on minimizing interval between timestamp placement to send time in section 3.6.

7. Added text recognizing the impending deployment of transport layer encryption in section 3.6.

8. Section 3.7.2 notes that some current systems perform host time stamping on the network interface hardware.

9. "instrument" replaced by the defined term "host" in sections 3.7.3 and 3.8.3.

10. Added reference to RFC 3432 Periodic sampling alongside Poisson sampling in section 4, and also noting that a truncated Poisson distribution may be needed with modern networks as described in the IPPM Framework update, RFC7312.

11. Add reference to RFC 4737 Reordering metric in the related discussion of section 4.6, Methodologies.

12. Clarifying the conclusions on two related points on harm to measurements (recognition of measurement traffic for unexpected priority treatment and attacker traffic which emulates measurement) in section 6, Security Considerations.

Section 5.4.4 of [RFC6390] suggests a common template for performance metrics partially derived from previous IPPM and BMWG RFCs, but also contains some new items. All of the [RFC6390] Normative points are

covered, but not quite in the same section names or orientation. Several of the Informative points are covered. Maintaining the familiar outline of IPPM literature has both value and minimizes unnecessary differences between this revised RFC and current/future IPPM RFCs.

The publication of RFC 6921 suggested an area where this memo might need updating. Packet transfer on Faster-Than-Light (FTL) networks could result in negative delays and packet reordering, however both are covered as possibilities in the current text and no revisions are deemed necessary (we also note that this is an April 1st RFC).

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, [RFC2330]; 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") [RFC2680].

Although [RFC2119] 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 sent 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 networks with asymmetric link capacities, or wireless vs. wireline access).

+ 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 TCP-based communication may experience reduced throughput if congestion occurs in one direction of its communication. Trouble shooting may be simplified if the congested direction of TCP transmission can be identified.

+ 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

+ Tmax, a loss threshold waiting 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, >>the *Type-P-One-way-Delay* from Src to Dst at T is dT<< 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.

>>The *Type-P-One-way-Delay* from Src to Dst at T is undefined (informally, infinite)<< 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 (within the loss threshold waiting time, Tmax).

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. This was tested in [RFC6808], where a GPS measurement system's results compared well with a GPS-based NTP synchronized system for the same intercontinental path.

+ 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 [RFC2678], simple upper bounds (such as the 255 seconds theoretical upper bound on the lifetimes of IP packets [RFC0791]) 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. See section 4.1.1 of [RFC6703] for examination of unusual packet delays and application performance estimation.}

+ 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.

+ The packet is standard-formed, the default criteria for all metric definitions defined in Section 15 of [RFC2330], otherwise 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. Note that use of transport layer encryption will counteract the deployment of network-based analysis and may reduce the adoption of payload optimizations like compression.

+ 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 (ideally minimizing time before sending).

+ 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, Tmax, the one-way delay is taken to be undefined (informally, infinite). Note that the threshold of 'reasonable' is a parameter of the metric. 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 Tsource, and the observed time when the packet was received

by the destination clock as Tdest. 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 Tsynch if the source clock is Tsynch ahead of the destination clock. Thus, if we know the value of Tsynch exactly, we could correct for clock synchronization by adding Tsynch to the uncorrected value of Tdest-Tsource.

+ 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 Rsource and Rdest, respectively.

+ The skew of a clock is not so much an additional issue as it is a realization of the fact that Tsynch is itself a function of time. Thus, if we attempt to measure or to bound Tsynch, 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 Tsynch is made smaller, but remains a source of uncertainty that must be accounted for. We use the function Esynch(t) to denote an upper bound on the uncertainty in synchronization. Thus, |Tsynch(t)| <= Esynch(t).

Taking these items together, we note that naive computation Tdest-Tsource will be off by Tsynch(t) +/- (Rsource + Rdest). Using the notion of Esynch(t), we note that these clock-related problems introduce a total uncertainty of Esynch(t)+ Rsource + Rdest. 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".

We note that some systems perform host time stamping on the network interface hardware, in an attempt to minimize the difference from wire 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:

measured value = true value + systematic error + random error

If the systematic error (the constant bias in measured values) can be determined, it can be compensated for in the reported results.

reported value = measured value - systematic error

therefore

reported value = true value + random error

The goal of calibration is to determine the systematic and random error generated by the hosts 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 host 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

Esynch(t) + Rsource + Rdest + Hsource + Hdest.

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 hosts 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 Esynch(t) + Rsource + Rdest is small, and is also bounded for the duration of the measurement because of the global time source.

The host-related uncertainties, Hsource + Hdest, could be bounded by connecting two hosts 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 measurement hosts. 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 host, 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 hosts will see in the field.

We wish to reiterate that this statistical treatment refers to the calibration of the host; it is used to "calibrate the meter stick" and say how well the meter stick reflects reality.

In addition to calibrating the hosts 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 host.

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 [RFC2330], 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 host 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 T0, a final time Tf, and an average rate lambda, then define a pseudo-random Poisson process of rate lambda, whose values fall between T0 and Tf.

The time interval between successive values of T will then average 1/lambda.

Note that Poisson sampling is only one way of defining a sample. Poisson has the advantage of limiting bias, but other methods of sampling will be appropriate for different situations. For example, a truncated Poisson distribution may be needed to avoid reactive network state changes during intervals of inactivity, see section 4.6 of [RFC7321]. Sometimes, the goal is sampling with a known bias, and [RFC3432] describes a method for periodic sampling with random start times.

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

+ T0, a time

+ Tf, a time

+ Tmax, a loss threshold waiting time

+ lambda, a rate in reciprocal seconds (or parameters for another distribution)

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 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-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 [RFC2330], 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 T0 to Tf, and given new time values T0' and Tf' such that T0 <= T0' <= Tf' <= Tf, the subsequence of the given sample whose time values fall between T0' and Tf' 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]. Metrics for reordering may be found in [RFC4737].

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 [RFC2330] 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. Therefore, the

measurement methodologies SHOULD include appropriate techniques to reduce the probability measurement traffic can be distinguished from "normal" traffic.

If an attacker injects packets emulating traffic that are accepted as legitimate, the loss ratio or other measured values could be corrupted. 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

For [RFC2679], 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.

For RFC 2679 bis, thanks to Joachim Fabini, Ruediger Geib, Nalini Elkins, and Barry Constantine for sharing their measurement experience as part of their careful reviews.

9. References

9.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[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.

[RFC2678] Mahdavi, J. and V. Paxson, "IPPM Metrics for Measuring Connectivity", RFC 2678, September 1999.

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

[RFC2680] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, September 1999.

[RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, November 2002.

[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.

[RFC7321] McGrew, D. and P. Hoffman, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 7321, August 2014.

9.2. Informative References

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, November 2006.

[RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, October 2011.

[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.

Authors' Addresses

Guy Almes
Texas A&M

Email: galmes@tamu.edu

Sunil Kalidindi
Ixia

Email: skalidindi@ixiacom.com

Matt Zekauskas
Internet2

Email: matt@internet2.edu

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/