Network Working Group
Internet-Draft
Updates: 2330 (if approved)
Intended status: Standards Track
Expires: April 29, 2018

J. Alvarez-Hamelin
Universidad de Buenos Aires
A. Morton
AT&T Labs
J. Fabini
TU Wien
October 26, 2017

Advanced Unidirectional Route Assessment
draft-amf-ippm-route-01

Abstract

This memo introduces an advanced unidirectional route assessment metric and associated measurement methodology, based on the IP Performance Metrics (IPPM) Framework RFC 2330. This memo updates RFC 2330 in the areas of path-related terminology and path description, primarily to include the possibility of parallel subpaths between a given Source and Destination pair, owing to the presence of multi-path technologies.

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 https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on April 29, 2018.

Table of Contents

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. It has been updated in the area of metric composition [RFC5835], and in several areas related to active stream measurement of modern networks with reactive properties [RFC7312].

The [RFC2330] framework motivated the development of "performance and reliability metrics for paths through the Internet," and Section 5 of [RFC2330] defines terms that support description of a path under test. However, metrics for assessment of path components and related performance aspects had not been attempted in IPPM when the [RFC2330] framework was written.

This memo takes-up the route measurement challenge and specifies a new route metric, two practical frameworks for methods of measurement (using either active or hybrid active-passive methods [RFC7799]), and round-trip delay and link information discovery using the results of measurements.

1.1. Issues with Earlier Work to define Route

Section 7 of [RFC2330] presented a simple example of a "route" metric along with several other examples. The example is reproduced below (where the reference is to Section 5 of [RFC2330]):

"route: The path, as defined in Section 5, from A to B at a given time."

This example provides a starting point to develop a more complete definition of route. Areas needing clarification include:

Time: In practice, the route will be assessed over a time interval, because active path detection methods like [PT] rely on TTL limits for their operation and cannot accomplish discovery of all hosts using a single packet.

Type-P: The legacy route definition lacks the option to cater for packet-dependent routing. In this memo, we assess the route for a specific packet of Type-P, and reflect this in the metric definition. The methods of measurement determine the specific Type-P used.

Parallel Paths: This a reality of Internet paths and a strength of advanced route assessment methods, so the metric must acknowledge this possibility. Use of Equal Cost Multi-Path (ECMP) and Unequal Cost Multi-Path (UCMP) technologies are common sources of parallel subpaths.

Cloud Subpath: May contain hosts that do not decrement TTL or Hop Limit, but may have two or more exchange links connecting

"discoverable" hosts or routers. Parallel subpaths contained within clouds cannot be discovered. The assessment methods only discover hosts or routers on the path that decrement TTL or Hop Count, or cooperate with interrogation protocols. The presence of tunnels and nested tunnels further complicate assessment by hiding hops.

Hop: Although the [RFC2330] definition was a link-host pair, only hosts are discoverable or have the capability to cooperate with interrogation protocols where link information may be exposed.

The refined definition of Route metrics begins in the sections that follow.

2. Scope

The purpose of this memo is to add new route metrics and methods of measurement to the existing set of IPPM metrics.

The scope is to define route metrics that can identify the path taken by a packet or a flow traversing the Internet between any two hosts.

<@@@@ or only hosts communicating at the IP layer? We would have to re-define the Src and Dst Parameters and Host Identity if we generalize beyond IP. Should we include MPLS and the capabilities of [RFC8029], with explicit multipath identification (section 6.2.6)? >

Also, to specify a framework for active methods of measurement which use the techniques described in [PT] at a minimum, and a framework for hybrid active-passive methods of measurement, such as the Hybrid Type I method [RFC7799] described in [I-D.ietf-ippm-ioam-data](intended only for single administrative domains), which do not rely on ICMP and provide a protocol for explicit interrogation of nodes on a path. Combinations of active methods and hybrid active-passive methods are also in-scope.

Further, this memo provides additional analysis of the round-trip delay measurements made possible by the methods, in an effort to discover more details about the path, such as the link technology in use.

This memo updates Section 5 of [RFC2330] in the areas of path-related terminology and path description, primarily to include the possibility of parallel subpaths between a given Source and Destination address pair (possibly resulting from Equal Cost Multi-Path (ECMP) and Unequal Cost Multi-Path (UCMP) technologies).

There are several simple non-goals of this memo. There is no attempt to assess the reverse path from any host on the path to the host attempting the path measurement. The reverse path contribution to delay will be that experienced by ICMP packets (in active methods), and may be different from UDP or TCP packets. Also, the round trip delay will include an unknown contribution of processing time at the host that generates the ICMP response. Therefore, the ICMP-based active methods are not supposed to yield accurate, reproducible estimations of the round-trip delay that UDP or TCP packets will experience.

3. Route Metric Terms and Definitions

This section sets requirements for the following components to support the Route Metric:

Note: the definitions concentrate on the IP-layer, but can be extended to other layers, and follow agreements on the scope.

Host Identity For hosts communicating at the IP-layer, the globally routable IP address(es) which the host uses when communicating with other hosts under normal or error conditions. The Host Identity revealed (and its connection to a Host Name through reverse DNS) determines whether interfaces to parallel links can be associated with a single host, or appear to be unique hosts.

Discoverable Host For hosts communicating at the IP-layer, compliance with Section 3.2.2.4 of [RFC1122] when discarding a packet due to TTL or Hop Limit Exceeded condition, MUST result in sending the corresponding Time Exceeded message (containing a form of host identity) to the source. This requirement is also consistent with section 5.3.1 of [RFC1812] for routers.

Cooperating Host Hosts MUST respond to direct queries for their host identity as part of a previously agreed and established interrogation protocol. Hosts SHOULD also provide information such as arrival/departure interface identification, arrival timestamp, and any relevant information about the host or specific link which delivered the query to the host.

Hop A Hop MUST contain a Host Identity, and MAY contain arrival and/or departure interface identification.

3.1. Formal Name

Type-P-Route-Ensemble-Method-Variant, abbreviated as Route Ensemble.

Note that Type-P depends heavily on the chosen method and variant.

3.2. Parameters

This section lists the REQUIRED input factors to specify a Route metric.

- Src, the IP address of a host
- Dst, the IP address of a host
- i, the TTL or Hop Limit of a packet sent from the host at Src to the host at Dst.
- MaxHops, the maximum value of i used, (i=1,2,3,...MaxHops).
- T0, a time (start of measurement interval)
- Tf, a time (end of measurement interval)
- T, the host time of a packet as measured at MP(Src), meaning Measurement Point at the Source.
- Ta, the host time of a reply packet's *arrival* as measured at MP(Src), assigned to packets that arrive within a "reasonable" time (see parameter below).
- Tmax, a maximum waiting time for reply packets to return to the source, set sufficiently long to disambiguate packets with long delays from packets that are discarded (lost), thus the distribution of delay is not truncated.
- F, the number of different flows simulated by the method and variant.
- flow, the stream of packets with the same n-tuple of designated header fields that (when held constant) results in identical treatment in a multi-path decision (such as that taken in load balancing).
- Type-P, the complete description of the packets for which this assessment applies (including the flow-defining fields).

3.3. Metric Definitions

This section defines the REQUIRED measurement components of the Route metrics (unless otherwise indicated):

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

N, the smallest value of i needed for a packet to be received at Dst (sent between T0 and Tf).

Nmax, the largest value of i needed for a packet to be received at Dst (sent between T0 and Tf). Nmax may be equal to N.

Next, define a *singleton* definition for a Hop on the path, with sufficient indexes to identify all Hops identified in a measurement interval.

A Hop, designated h(i,j), the IP address and/or identity of one of j Discoverable Hosts (or Cooperating Hosts) that are i hops away from the host with IP address = Src during the measurement interval, T0 to Tf. As defined above, a Hop singleton measurement MUST contain a Host Identity, hid(i,j), and MAY contain one or more of the following attributes:

- a(i,j) Arrival Interface ID
- d(i,j) Departure Interface ID
- t(i,j) Arrival Timestamp (where t(i,j) is ideally supplied by the hop, or approximated from the sending time of the packet that revealed the hop)
- Measurements of Round Trip Delay (for each packet that reveals the same Host Identity and attributes, but not timestamp of course, see next section)

Now that Host Identities and related information can be positioned according to their distance from the host with address Src in hops, we introduce two forms of Routes:

A Route Ensemble is defined as the combination of all routes traversed by different flows from the host at Src address to the host at Dst address. The route traversed by each flow (with addresses Src and Dst, and other fields which constitute flow criteria) is a member of the ensemble and called a Member Route.

Using h(i,j) and components and parameters, further define:

A Member Route is an ordered graph {h(1,j), ... h(Nj, j)} in the context of a single flow, where h(i-1, j) and h(i, j) are by 1 hop away from each other and Nj=Dst is the minimum TTL value needed by the packet on Member Route j to reach Dst. Member Routes must be unique. This uniqueness requires that any two Member routes j and k that are part of the same Route Ensemble differ either in terms of minimum hop count Nj and Nk to reach the destination Dst, or, in the

case of identical hop count Nj=Nk, they have at least one distinct hop: h(i,j) != h(i, k) for at least one i (i=1..Nj).

The Route Ensemble from Src to Dst, during the measurement interval T0 to Tf, is the aggregate of all m distinct Member Routes discovered between the two hosts with Src and Dst addresses. More formally, with the host having address Src omitted:

```
Route Ensemble = {
{h(1,1), h(2,1), h(3,1), ... h(N1,1)=Dst},
{h(1,2), h(2,2), h(3,2),..., h(N2,2)=Dst},
...
{h(1,m), h(2,m), h(3,m), ....h(Nm,m)=Dst}
}
```

where the following conditions apply: i <= Nj <= Nmax (j=1..m)

Note that some h(i,j) may be empty (null) in the case that systems do not reply (not discoverable, or not cooperating).

h(i-1,j) and h(i,j) are the Hops on the same Member Route one hop away from each other.

Hop h(i,j) may be identical with h(k,l) for i!=k and j!=l ; which means there may be portions shared among different Member Routes (parts of various routes may overlap).

3.4. Related Round-Trip Delay and Loss Definitions

RTD(i,j,T) is defined as a singleton of the [RFC2681] Round-trip Delay between the host with IP address = Src and the host at Hop h(i,j) at time T.

RTL(i,j,T) is defined as a singleton of the [RFC6673] Round-trip Loss between the host with IP address = Src and the host at Hop h(i,j) at time T.

3.5. Discussion

Depending on the way that Host Identity is revealed, it may be difficult to determine parallel subpaths between the same pair of hosts (i.e. multiple parallel links). It is easier to detect parallel subpaths involving different hosts.

- If a pair of discovered hosts identify two different IP addresses, then they will appear to be different hosts.

- If a pair of discovered hosts identify two different IP addresses, and the IP addresses resolve to the same host name (in the DNS), then they will appear to be the same hosts.

- If a discovered host always replies using the same IP address, regardless of the interface a packet arrives on, then multiple parallel links cannot be detected at the IP layer.

- If parallel links between routers are aggregated below the IP layer, In other words, all links share the same pair of IP addresses, then the existence of these parallel links can't be detected at IP layer.

Section 9.2 of [RFC2330] describes Temporal Composition of metrics, and introduces the possibility of a relationship between earlier measurement results and the results for measurement at the current time (for a given metric). If this topic is investigated further, there may be some value in establishing a Temporal Composition relationship for Route Metrics. However, this relationship does not represent a forecast of future route conditions in any way.

When a route assessment employs packets at the IP layer (for example), the reality of flow assignment to parallel subpaths involves layers above IP. Thus, the measured Route Ensemble is applicable to IP and higher layers (as described in the methodology's packet of Type-P and flow parameters).

@@@@ Editor's Note: There is an opportunity to investigate and discuss the RFC 2330 notion of equal treatment for a class of packets, "...very useful to know if a given Internet component treats equally a class C of different types of packets", as it applies to Route measurements. Knowledge of "class C" parameters on a path potentially reduces the number of flows required for a given method.

3.6. Reporting the Metric

@@@@ to be provided

4. Route Assessment Methodologies

There are two classes of methods described in this section, active methods relying on the reaction to TTL or Hop Limit Exceeded condition to discover hosts on a path, and Hybrid active-passive methods that involve direct interrogation of cooperating hosts (usually within a single domain). Description of these methods follow.

@@@@ Editor's Note: We need to incorporate description of Type-P packets (with the flow parameters) used in each method below.

4.1. Active Methodologies

We have chosen to describe the method based on that employed in current open source tools, thereby providing a practical framework for further advanced techniques to be included as method variants. This method is applicable to use across multiple administrative domains.

Paris-traceroute [PT] provides some measure of protection from path variation generated by ECMP load balancing, and it ensures traceroute packets will follow the same path in 98% of cases according to [SCAMPER]. If it is necessary to find every path possible between two hosts, Paris-traceroute provides "exhaustive" mode while scamper provides "tracelb" (stands for traceroute load balance).

The Type-P of packets used could be ICMP (as ones in the original traceroute), UDP and TCP. The later are used when a particular characteristic is needed to verify, such as filtering or traffic shaping on specific ports (i.e., services).

The advanced route assessment methods used in Paris-traceroute [PT] keep the critical fields constant for every packet to maintain the appearance of the same flow. Since route assessment can be conducted using TCP, UDP or ICMP packets, this method REQUIRES the Diffserv field, the protocol number, IP source and destination addresses, and the port settings for TCP or UDP kept constant. For ICMP probes, the method additionally REQUIRES the type, code, and ICMP checksum constant; which take the same position in the header of an IP packet, e.g., bytes 20 to 23 when the header IP has no options.

Maintaining a constant checksum in ICMP is most challenging because the ICMP Sequence Number is part of the calculation. The advanced traceroute method requires calculations using the IP Sequence Number Field and the Identifier Field, yielding a constant ICMP checksum in successive packets. For an example of calculations to maintain a constant checksum, see Appendix A of [RFC7820], where revision of a timestamp field is complemented by modifying the 2 octet checksum complement field (these fields take the roles of the ICMP Sequence Number Identifier Fields, respectively).

For TCP and UDP packets, the checksum must also be kept constant. Therefore, the first four bytes of UDP (or TCP) data field are modified to compensate for fields that change from packet to packet.

Note: other variants of advanced traceroute are planned be described.

Finally, the return path is also important to check. Taking into account that it is an ICMP time exceeded (during transit) packet, the source and destination IP are constant for every reply. Then, we should consider the fields in the first 32 bits of the protocol on the top of IP: the type and code of ICMP packet, and its checksum. Again, to maintain the ICMP checksum constant for the returning packets, we need to consider the whole ICMP message. It contains the IP header of the discarded packet plus the first 8 bytes of the IP payload; that is some of the fields of TCP header, the UDP header plus four data bytes, the ICMP header plus four bytes. Therefore, for UDP case the data field is used to maintain the ICMP checksum constant in the returning packet. For the ICMP case, the identifier and sequence fields of the sent ICMP probe are manipulated to be constant. The TCP case presents no problem because its first eight bytes will be the same for every packet probe.

Formally, to maintain the same flow in the measurements to a certain hop, the Type-P-Route-Ensemble-Method-Variant packets should be[PT]:

- TCP case: Fields Src, Dst, port-Src, port_Dst, and Diffserv Field should be the same.

- UDP case: Fields Src, Dst, port-Src, port-Dst, and Diffserv Field should be the same, the UDP-checksum should change to maintain constant the IP checksum of the ICMP time exceeded reply. Then, the data length should be fixed, and the data field is used to fixing it (consider that ICMP checksum uses its data field, which contains the original IP header plus 8 bytes of UDP, where TTL, IP identification, IP checksum, and UDP checksum changes).

- ICMP case: The Data field should compensate variations on TTL, IP identification, and IP checksum for every packet.

Then, the way to identify different hops and attempts of the same flow is:

- TCP case: The IP identification field.

- UDP case: The IP identification field.

- ICMP case: The IP identification field, and ICMP Sequence number.

4.2. Hybrid Methodologies

The Hybrid Type I methods provide an alternative method for Route Member assessment. As mentioned in the Scope section, [I-D.ietf-ippm-ioam-data] provides a possible set of data fields that would support route identification.

In general, nodes in the measured domain would be equipped with specific abilities:

1. The ingress node adds one or more fields to the measurement packets, and identifies to other nodes in the domain that a route assessment will be conducted using one or more specific packets. The packets typically originate from a host outside the domain, and constitute normal traffic on the domain.

2. Each node visited by the specific packet within in the domain identifies itself in a data field of the packet (the field has been added for this purpose).

3. When a measurement packet reaches the edge node of the domain, the edge node adds its identity to the list, removes all the identities from the packet, forwards the packet onward, and communicates the ordered list of node identities to the intended receiver.

In addition to node identity, nodes may also identify the ingress and egress interfaces utilized by the tracing packet, the time of day when the packet was processed, and other generic data (as described in section 4 of [I-D.ietf-ippm-ioam-data]).

4.3. Combining Different Methods

In principle, there are advantages if the entity conducting Route measurements can utilize both forms of advanced methods (active and hybrid), and combine the results. For example, if there are hosts involved in the path that qualify as Cooperating Hosts, but not as Discoverable Hosts, then a more complete view of hops on the path is possible when a hybrid method (or interrogation protocol) is applied and the results are combined with the active method results collected across all other domains.

In order to combine the results of active and hybrid/interrogation methods, the network hosts that are part of a domain supporting an interrogation protocol have the following attributes:

1. Hosts at the ingress to the domain SHOULD be both Discoverable and Cooperating, and SHOULD reveal the same Host Identity in response to both active and hybrid methods.

2. Any Hosts within the domain that are both Discoverable and Cooperating SHOULD reveal the same Host Identity in response to both active and hybrid methods.

3. Hosts at the egress to the domain SHOULD be both Discoverable and Cooperating, and SHOULD reveal the same Host Identity in response to both active and hybrid methods.

When Hosts follow these requirements, it becomes a simple matter to match single domain measurements with the overlapping results from a multidomain measurement.

In practice, Internet users do not typically have the ability to utilize the OAM capabilities of networks that their packets traverse, so the results from a remote domain supporting an interrogation protocol would not normally be accessible. However, a network operator could combine interrogation results from their access domain with other measurements revealing the path outside their domain.

5. Background on Round-Trip Delay Measurement Goals

The aim of this method is to use packet probes to unveil the paths between any two end-hosts of the network. Moreover, information derived from RTD measurements might be meaningful to identify:

1. Intercontinental submarine links

2. Satellite communications

3. Congestion

4. Inter-domain paths

This categorization is widely accepted in the literature and among operators alike, and it can be trusted with empirical data and several sources as ground of truth (e.g., [RTTSub] [bdrmap][IDCong]).

The first two categories correspond to the physical distance dependency on Round Trip Delay (RTD) while the last one binds RTD with queueing delay on routers. Due to the significant contribution of propagation delay in long distance hops, RTD will be at least 100ms on transatlantic hops, depending on the geolocation of the vantage points. Moreover, RTD is typically greater than 480ms when two hops are connected using geostationary satellite technology (i.e., their orbit is at 36000km). Detecting congestion with latency implies deeper mathematical understanding since network traffic load is not stationary. Nonetheless, as the first approach, a link seems to be congested if after sending several traceroute probes, it is possible to detect congestion observing different statistics parameters (e.g., see [IDCong]).

6. Tools to Measure Delays in the Internet

Internet routing is complex because it depends on the policies of thousands Autonomous Systems (AS). While most of the routers perform load balancing on flows using Equal Cost Multiple Path (ECMP), a few still divide the workload through packet-based techniques. The former scenario is defined according to [RFC2991] while the latter generates a round-robin scheme to deliver every new outgoing packet. ECMP keeps flow state in the router to ensure every packet of a flow is delivered by the same path, and this avoids increasing the packet delay variation and possibly producing overwhelming packet reordering in TCP flows.

Taking into account that Internet protocol was designed under the "end-to-end" principle, the IP payload and its header do not provide any information about the routes or path necessary to reach some destination. For this reason, the well-known tool traceroute was developed to gather the IP addresses of each hop along a path using the ICMP protocol [RFC0792]. Besides, traceroute adds the measured RTD from each hop. However, the growing complexity of the Internet makes it more challenging to develop accurate traceroute implementation. For instance, the early traceroute tools would be inaccurate in the current network, mainly because they were not designed to retain flow state. However, evolved traceroute tools, such as Paris-traceroute [PT] [MLB] and Scamper [SCAMPER], expect to encounter ECMP and achieve more accurate results when they do.

Paris-traceroute-like tools operate in the following way: every packet should follow the same path because the sensitive fields of the header are controlled to appear as the same flow. This means that source and destination IP addresses, source and destination port numbers are the same in every packet. Additionally, Differentiated Services Code Point (DSCP), checksum and ICMP code should remain constant since they may affect the path selection.

Today's traceroute tools can send either UDP, TCP or ICMP packet probes. Since ICMP header does not include transport layer information, there are no fields for source and destination port numbers. For this reason, these tools keep constant ICMP type, code, and checksum fields to generate a kind of flow. However, the checksum may vary in every packet, therefore when probes use ICMP packets, ICMP Identifier and Sequence Number are manipulated to maintain constant checksum in every packet. On the other hand, when UDP probes are generated, the expected variation in the checksum of each packet is again compensated by manipulating the payload.

Paris-traceroute allows its users to measure RTD in every hop of the path for a particular flow. Furthermore, either Paris-traceroute or

Scamper is capable of unveiling the many available paths between a source and destination (which are visible to this method). This task is accomplished by repeating complete traceroute measurements with different flow parameters for each measurement. The Framework for IP Performance Metrics (IPPM) ([RFC2330] updated by[RFC7312]) has the flexibility to require that the round-trip delay measurement [RFC2681] uses packets with the constraints to assure that all packets in a single measurement appear as the same flow. This flexibility covers ICMP, UDP, and TCP. The accompanying methodology of [RFC2681] needs to be expanded to report the sequential hop identifiers along with RTD measurements, but no new metric definition is needed.

7. RTD Measurements Statistics

Several articles have shown that network traffic presents a self-similar nature [SSNT] [MLRM] which is accountable for filling the queues of the routers. Moreover, router queues are designed to handle traffic bursts, which is one of the most remarkable features of self-similarity. Naturally, while queue length increases, the delay to traverse the queue increases as well and leads to an increase on RTD. Due to traffic bursts generate short-term overflow on buffers (spiky patterns), every RTD only depicts the queueing status on the instant when that packet probe was in transit. For this reason, several RTD measurements during a time window could begin to describe the random behavior of latency. Loss must also be accounted for in the methodology.

To understand the ongoing process, examining the quartiles provides a non-parametric way of analysis. Quartiles are defined by five values: minimum RTD (m), RTD value of the 25% of the Empirical Cumulative Distribution Function (ECDF) (Q1), the median value (Q2), the RTD value of the 75% of the ECDF (Q3) and the maximum RTD (M). Congestion can be inferred when RTD measurements are spread apart, and consequently, the Inter-Quartile Range (IQR), the distance between Q3 and Q1, increases its value.

This procedure requires to compute quartile values "on the fly" using the algorithm presented in [P2].

This procedure allow us to update the quartiles value whenever a new measurement arrives, which is radically different from classic methods of computing quartiles because they need to use the whole dataset to compute the values. This way of calculus provides savings in memory and computing time.

To sum up, the proposed measurement procedure consists in performing traceroutes several times to obtain samples of the RTD in every hop

from a path, during a time window (W) and compute the quantiles for every hop. This could be done for a single path flow or for every detected path flow.

Even though a particular hop may be understood as the amount of hops away from the source, a more detailed classification could be used. For example, a possible classification may be identify ICMP Time Exceeded packets coming from the same routers to those who have the same hop distance, IP address of the router which is replying and TTL value of the received ICMP packet.

Thus, the proposed methodology is based on this algorithm:

```
 ================================================================
  1  input:   W (window time of the measurement)
  2           i_t (time between two measurements)
  3           E (True: exhaustive, False: a single path)
  4           Dst (destination IP address)
  5  output:  Qs (quartiles for every hop and alt in the path(s) to Dst)
 ----------------------------------------------------------------
  6  T <? start_timer(W)
  7  while T is not finished do:
  8  |        start_timer(i_t)
  9  |        RTD(hop,alt) = advanced-traceroute(Dst,E)
 10  |        for each hop and alt in RTD do:
 11  |        |     Qs[Dst,hop,alt] <? ComputeQs(RTD(hop,alt))
 12  |        done
 13  |        wait until i_t timer is expired
 14  done
 15  return (Qs)
 ================================================================
```

In line 9 the advance-traceroute could be either Paris-traceroute or Scamper, which will use "exhaustive" mode or "tracelb" option if E is set True, respectively. The procedure returns a list of tuples (m,Q1,Q2,Q3,M) for each intermediate hop in the path towards the Dst. Additionally, it could also return path variations using "alt" variable.

8. Conclusions

Combining the method proposed in Section 4 and statistics in Section 7, we can measure the performance of paths interconnecting two endpoints in Internet, and attempt the categorization of link types and congestion presence based on RTD.

9. Security Considerations

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

The active measurement process of "changing several fields to keep the checksum of different packets identical" does not require special security considerations because it is part of synthetic traffic generation, and is designed to have minimal to zero impact on network processing (to process the packets for ECMP).

@@@@ add reference to security considerations from [I-D.ietf-ippm-ioam-data].

When considering privacy of those involved in measurement or those whose traffic is measured, the sensitive information available to potential observers is greatly reduced when using active techniques which are within this scope of work. Passive observations of user traffic for measurement purposes raise many privacy issues. We refer the reader to the privacy considerations described in the Large Scale Measurement of Broadband Performance (LMAP) Framework [RFC7594], which covers active and passive techniques.

10. IANA Considerations

This memo makes no requests of IANA.

11. Acknowledgements

The authors acknowledge Ruediger Geib, for his penetrating comments on the initial draft. Carlos Pignataro challenged the authors to consider a wider scope, and applied his substantial expertise with many technologies and their measurement features in his extensive comments. Frank Brockners also shared useful comments. We thank them all!

12. References

12.1. Normative References

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-00 (work in progress), September 2017.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

[RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", RFC 1812, DOI 10.17487/RFC1812, June 1995, <https://www.rfc-editor.org/info/rfc1812>.

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

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <https://www.rfc-editor.org/info/rfc2330>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <https://www.rfc-editor.org/info/rfc2460>.

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, DOI 10.17487/RFC2675, August 1999, <https://www.rfc-editor.org/info/rfc2675>.

[RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <https://www.rfc-editor.org/info/rfc2681>.

[RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, DOI 10.17487/RFC2991, November 2000, <https://www.rfc-editor.org/info/rfc2991>.

[RFC4494] Song, JH., Poovendran, R., and J. Lee, "The AES-CMAC-96 Algorithm and Its Use with IPsec", RFC 4494, DOI 10.17487/RFC4494, June 2006, <https://www.rfc-editor.org/info/rfc4494>.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5644] Stephan, E., Liang, L., and A. Morton, "IP Performance Metrics (IPPM): Spatial and Multicast", RFC 5644, DOI 10.17487/RFC5644, October 2009, <https://www.rfc-editor.org/info/rfc5644>.

[RFC5835] Morton, A., Ed. and S. Van den Berghe, Ed., "Framework for Metric Composition", RFC 5835, DOI 10.17487/RFC5835, April 2010, <https://www.rfc-editor.org/info/rfc5835>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <https://www.rfc-editor.org/info/rfc6282>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <https://www.rfc-editor.org/info/rfc6437>.

[RFC6564] Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., and M. Bhatia, "A Uniform Format for IPv6 Extension Headers", RFC 6564, DOI 10.17487/RFC6564, April 2012, <https://www.rfc-editor.org/info/rfc6564>.

[RFC6673] Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, DOI 10.17487/RFC6673, August 2012, <https://www.rfc-editor.org/info/rfc6673>.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <https://www.rfc-editor.org/info/rfc7045>.

[RFC7312] Fabini, J. and A. Morton, "Advanced Stream and Sampling Framework for IP Performance Metrics (IPPM)", RFC 7312, DOI 10.17487/RFC7312, August 2014, <https://www.rfc-editor.org/info/rfc7312>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <https://www.rfc-editor.org/info/rfc7799>.

[RFC7820] Mizrahi, T., "UDP Checksum Complement in the One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP)", RFC 7820, DOI 10.17487/RFC7820, March 2016, <https://www.rfc-editor.org/info/rfc7820>.

[RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <https://www.rfc-editor.org/info/rfc8029>.

12.2. Informative References

[bdrmap] Luckie, M., Dhamdhere, A., Huffaker, B., Clark, D., and KC. Claffy, "bdrmap: Inference of Borders Between IP Networks", In Proceedings of the 2016 ACM on Internet Measurement Conference, pp. 381-396. ACM, 2016.

[I-D.brockners-inband-oam-data] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-brockners-inband-oam-data-07 (work in progress), July 2017.

[IDCong] Luckie, M., Dhamdhere, A., Clark, D., and B. Huffaker, "Challenges in inferring Internet interdomain congestion", In Proceedings of the 2014 Conference on Internet Measurement Conference, pp. 15-22. ACM, 2014.

[MLB] Augustin, B., Friedman, T., and R. Teixeira, "Measuring load-balanced paths in the Internet", Proceedings of the 7th ACM SIGCOMM conference on Internet measurement, pp. 149-160. ACM, 2007., 2007.

[MLRM] Fontugne, R., Mazel, J., and K. Fukuda, "An empirical mixture model for large-scale RTT measurements", 2015 IEEE Conference on Computer Communications (INFOCOM), pp. 2470-2478. IEEE, 2015., 2015.

[P2] Jain, R. and I. Chlamtac, "The P 2 algorithm for dynamic calculation of quantiles and histograms without storing observations", Communications of the ACM 28.10 (1985): 1076-1085, 2015.

[PT] Augustin, B., Cuvellier, X., Orgogozo, B., Viger, F., Friedman, T., Latapy, M., Magnien, C., and R. Teixeira, "Avoiding traceroute anomalies with Paris traceroute", Proceedings of the 6th ACM SIGCOMM conference on Internet measurement, pp. 153-158. ACM, 2006., 2006.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/info/rfc7594>.

[RTTSub] Bischof, Z., Rula, J., and F. Bustamante, "In and out of Cuba: Characterizing Cuba's connectivity", In Proceedings of the 2015 ACM Conference on Internet Measurement Conference, pp. 487-493. ACM, 2015.

[SCAMPER] Matthew Luckie, M., "Scamper: a scalable and extensible packet prober for active measurement of the Internet", Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, pp. 239-245. ACM, 2010., 2010.

[SSNT] Park, K. and W. Willinger, "Self-Similar Network Traffic and Performance Evaluation (1st ed.)", John Wiley & Sons, Inc., New York, NY, USA, 2000.

Authors' Addresses

Jose Ignacio Alvarez-Hamelin
Universidad de Buenos Aires
Av. Paseo Colon 850
Buenos Aires C1063ACV
Argentine

Phone: +54 11 5285-0716
Email: ihameli@cnet.fi.uba.ar
URI: http://cnet.fi.uba.ar/ignacio.alvarez-hamelin/

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/

Joachim Fabini
TU Wien
Gusshausstrasse 25/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/

IPPM Working Group
Internet-Draft
Intended status: Experimental
Expires: December 31, 2018

G. Fioccola, Ed.
M. Cociglio
Telecom Italia
A. Sapio
R. Sisto
Politecnico di Torino
June 29, 2018

Multipoint Alternate Marking method for passive and hybrid performance monitoring
draft-fioccola-ippm-multipoint-alt-mark-04

Abstract

The Alternate Marking method, as presented in RFC 8321 [RFC8321], can be applied only to point-to-point flows because it assumes that all the packets of the flow measured on one node are measured again by a single second node. This document aims to generalize and expand this methodology to measure any kind of unicast flows, whose packets can follow several different paths in the network, in wider terms a multipoint-to-multipoint network. For this reason the technique here described is called Multipoint Alternate Marking. Some definitions here introduced extend the scope of RFC 5644 [RFC5644] in the context of alternate marking schema.

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 https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on December 31, 2018.

Table of Contents

1. Introduction

The alternate marking method, as presented until now, is applicable to a point-to-point path; so the extension proposed in this document explains the most general case of multipoint-to-multipoint path and

enables flexible and adaptive performance measurements in a managed network.

The Alternate Marking methodology described in RFC 8321 [RFC8321] has the property to synchronize measurements in different points maintaining the coherence of the counters. So it is possible to show what is happening in every marking period for each monitored flow. The monitoring parameters are the packet counter and timestamps of a flow for each marking period.

There are some applications of the alternate marking method where there are a lot of monitored flows and nodes. Multipoint Alternate Marking aims to reduce these values and makes the performance monitoring more flexible in case a detailed analysis is not needed. For instance, by considering n measurement points and m monitored flows,the order of magnitude of the packet counters for each time interval is n*m*2 (1 per color). If both n and m are high values the packet counters increase a lot and Multipoint Alternate Marking offers a tool to control these parameters.

The approach presented in this document is applied only to unicast flows and not to multicast. BUM (Boradcast Unkown Unicast Multicast) traffic is not considered here, because traffic replication is not covered by the Multipoint Alternate Marking method.

Alternate Marking method works by definition for multipoint to multipoint paths but the network clustering approach presented in this document is the formalization of how to implement this property and it allows a flexible and optimized performance measurement support.

Without network clustering, it is possible to apply alternate marking only for all the network or per single flow. Instead, with network clustering, it is possible to use the network clusters partition at different levels to perform the needed degree of detail. In some circumstances it is possible to monitor a Multipoint Network by analyzing the Network Clustering, without examining in depth. In case of problems (packet loss is measured or the delay is too high) the filtering criteria could be specified more in order to perform a detailed analysis by using a different combination of clusters up to a per-flow measurement as described in RFC 8321 [RFC8321].

An application could be the Software Defined Network (SDN) paradigm where the SDN Controllers are the brains of the network and can manage flow control to the switches and routers and, in the same way, can calibrate the performance measurements depending on the necessity. An SDN Controller Application can orchestrate how deep the network performance monitoring is setup.

2. Correlation with RFC5644

RFC 5644 [RFC5644] is limited to active measurements using a single source packet or stream, and observations of corresponding packets along the path (spatial), at one or more destinations (one-to-group), or both. Instead, the scope of this memo is to define multiparty metrics for passive and hybrid measurements in a group-to-group topology with multiple sources and destinations.

RFC 5644 [RFC5644] introduces metric names that can be reused also here but have to be extended and rephrased to be applied to the alternate marking schema:

a. the multiparty metrics are not only one-to-group metrics but can be also group-to-group metrics;

b. the spatial metrics, used for measuring the performance of segments of a source to destination path, are applied here to group-to-group segments (called Clusters).

3. Flow classification

An unicast flow is identified by all the packets having a set of common characteristics. This definition is inspired by RFC 7011 [RFC7011].

As an example, by considering a flow as all the packets sharing the same source IP address or the same destination IP address, it is easy to understand that the resulting pattern will not be a point-to-point connection, but a point-to-multipoint or multipoint-to-point connection.

In general a flow can be defined by a set of selection rules used to match a subset of the packets processed by the network device. These rules specify a set of headers fields (Identification Fields) and the relative values that must be found in matching packets.

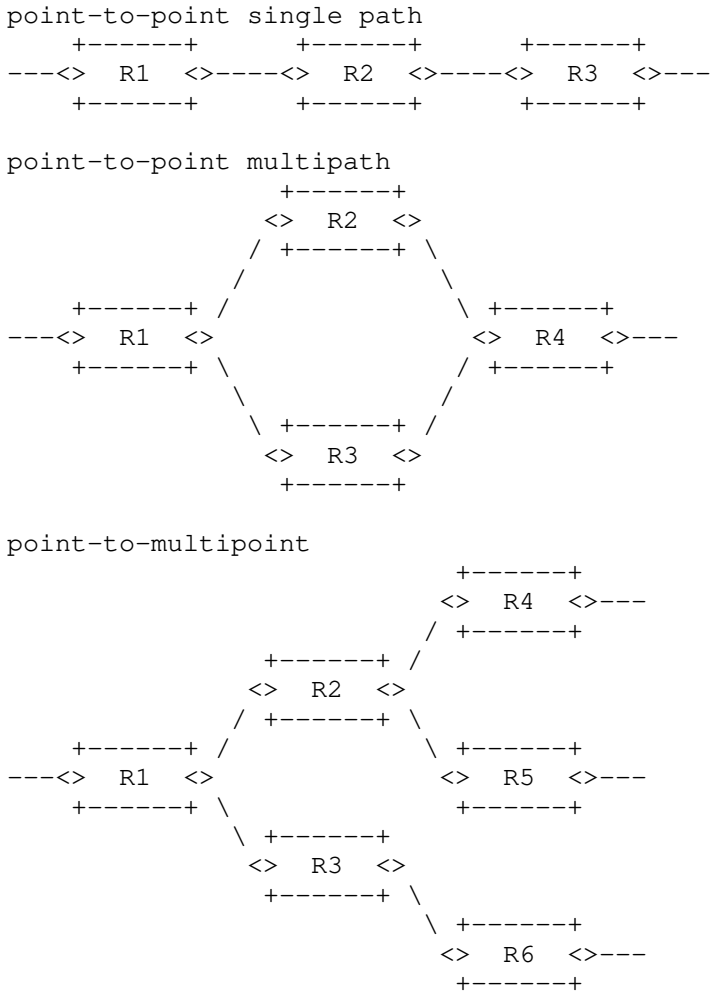
The choice of the identification fields directly affects the type of paths that the flow would follow in the network. In fact, it is possible to relate a set of identification fields with the pattern of the resulting graphs, as listed in Figure 1.

A TCP 5-tuple usually identifies flows following either a single path or a point-to-point multipath (in case of load balancing). On the contrary, a single source address selects flows following a point-to-multipoint, while a multipoint-to-point can be the result of a matching on a single destination address. In case a selection rule and its reverse are used for bidirectional measurements, they can

correspond to a point-to-multipoint in one direction and a multipoint-to-point in the opposite direction.

In this way the flows to be monitored are selected into the monitoring points using packet selection rules, that can also change the pattern of the monitored network.

The alternate marking method is applicable only to a single path (and partially to a one-to-one multipath), so the extension proposed in this document is suitable also for the most general case of multipoint-to-multipoint, which embraces all the other patterns of Figure 1.



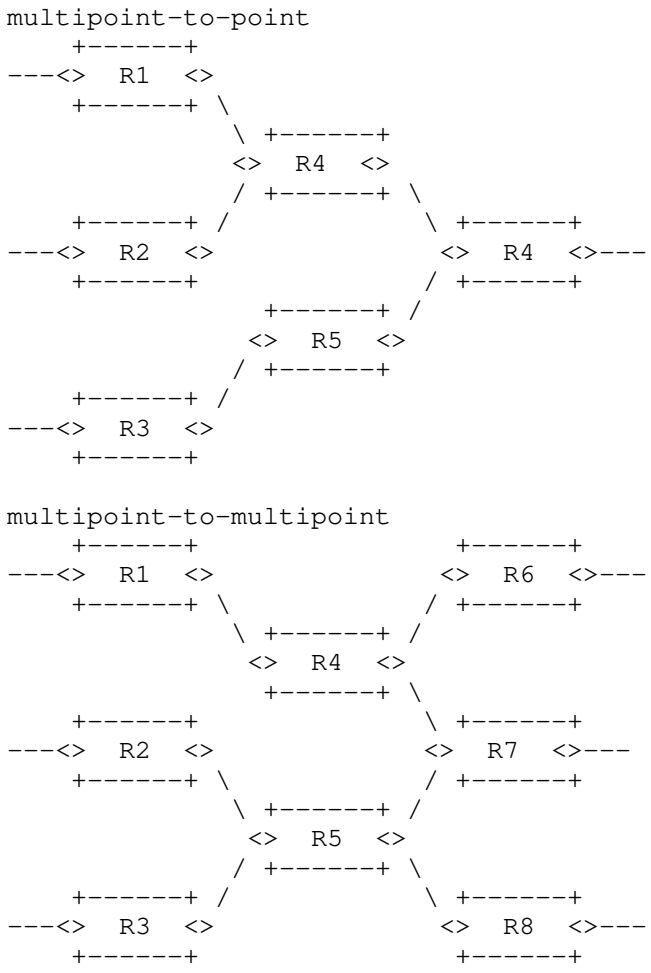


Figure 1: Flow classification

4. Multipoint Performance Measurement

By Using the "traditional" alternate marking method only point-to-point paths can be monitored. To have an IP (TCP/UDP) flow that follows a point-to-point path we have to define, with a specific value, 5 identification fields (IP Source, IP Destination, Transport Protocol, Source Port, Destination Port).

Multipoint Alternate Marking enables the performance measurement for multipoint flows selected by identification fields without any

constraints (even the entire network production traffic). It is also possible to use multiple marking points for the same monitored flow.

4.1. Monitoring Network

The Monitoring Network is deduced from the Production Network, by identifying the nodes of the graph that are the measurement points, and the links that are the connections between measurement points.

There are some techniques that can help with the building of the monitoring network (as an example it is possible to mention [I-D.amf-ippm-route]). In general there are different options: the monitoring network can be obtained by considering all the possible paths for the traffic or also by checking the traffic sometimes and update the graph consequently.

So a graph model of the monitoring network can be built according to the alternate marking method: the monitored interfaces and links are identified. Only the measurement points and links where the traffic has flowed have to be represented in the graph.

The following figure shows a simple example of a Monitoring Network graph:

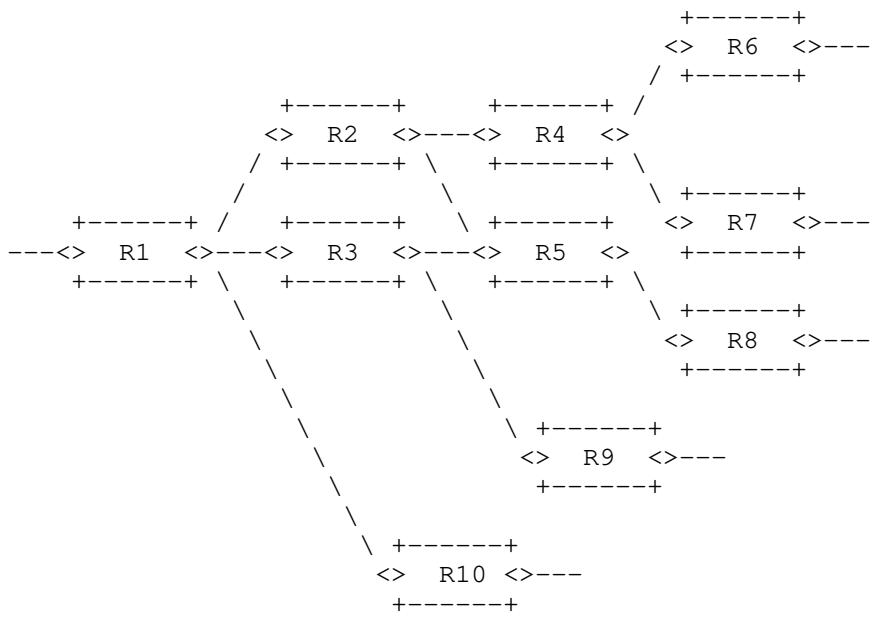


Figure 2: Monitoring Network Graph

Each monitoring point is characterized by the packet counter that refers only to a marking period of the monitored flow.

The same is applicable also for the delay but it will be described in the following sections.

5. Multipoint Packet Loss

Since all the packets of the considered flow leaving the network have previously entered the network, the number of packets counted by all the input nodes is always greater or equal than the number of packets counted by all the output nodes.

And in case of no packet loss occurring in the marking period, if all the input and output points of the network domain to be monitored are measurement points, the sum of the number of packets on all the ingress interfaces and on all the egress interfaces is the same. In this circumstance, if no packet loss occurs, the intermediate measurement points have only the task to split the measurement.

It is possible to define the Network Packet Loss (for 1 flow, for 1 period): <<In a packet network, the number of lost packets is the

number of packets counted by the input nodes minus the number of packets counted by the output nodes>>. This is true for every packet flow in each marking period.

The Monitored Network Packet Loss with n input nodes and m output nodes is given by:

PL = (PI1 + PI2 +...+ PIn) - (PO1 + PO2 +...+ POm)

where:

PL is the Network Packet Loss (number of lost packets)

PIi is the Number of packets flowed through the i-th Input node in this period

POj is the Number of packets flowed through the j-th Output node in this period

The equation is applied on a per-time-interval basis.

6. Network Clustering

The previous Equation can determine the number of packets lost globally in the monitored network, exploiting only the data provided by the counters in the input and output nodes.

In addition it is also possible to leverage the data provided by the other counters in the network to converge on the smallest identifiable subnetworks where the losses occur. These subnetworks are named Clusters.

A Cluster graph is a subnetwork of the entire Monitoring Network graph that still satisfies the packet loss equation where PL in this case is the number of packets lost in the Cluster.

For this reason a Cluster should contain all the arcs emanating from its input nodes and all the arcs terminating at its output nodes. This ensures that we can count all the packets (and only those) exiting an input node again at the output node, whatever path they follow.

In a completely monitored network (a network where every network interface is monitored), each network device corresponds to a Cluster and each physical link corresponds to two Clusters (one for each direction).

Clusters can have different sizes depending on flow filtering criteria adopted.

Moreover, sometimes Clusters can be optionally simplified. For example when two monitored interfaces are divided by a single router (one is the input interface and the other is the output interface and the router has only these two interfaces), instead of counting exactly twice, upon entering and leaving, it is possible to consider a single measurement point (in this case we do not care of the internal packet loss of the router).

6.1. Algorithm for Cluster partition

A simple algorithm can be applied in order to split our monitoring network into Clusters. It is a two-step algorithm:

- Group the links where there is the same starting node;
- Join the grouped links with at least one ending node in common.

In our monitoring network graph example it is possible to identify the Clusters partition by applying this two-step algorithm.

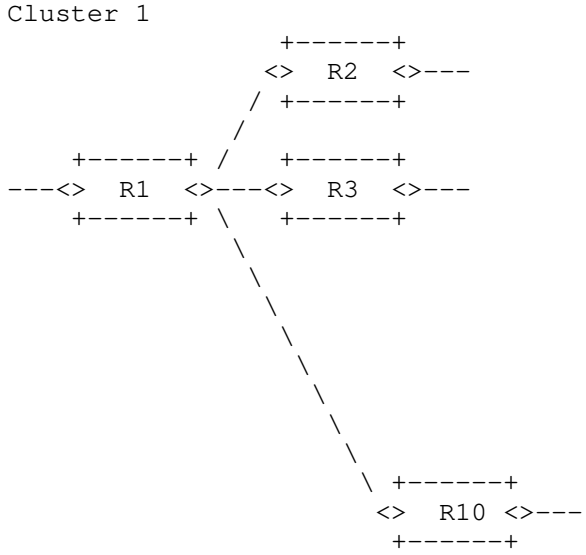
The first step identifies the following groups:

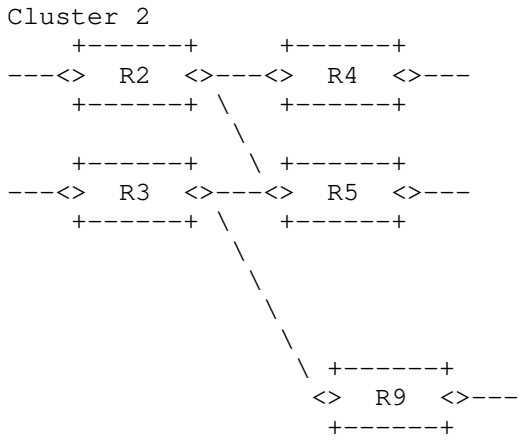
1. Group 1: (R1-R2), (R1-R3), (R1-R10)
2. Group 2: (R2-R4), (R2-R5)
3. Group 3: (R3-R5), (R3-R9)
4. Group 4: (R4-R6), (R4-R7)
5. Group 5: (R5-R8)

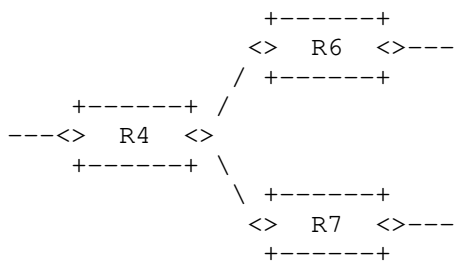
And then, the second step builds the Clusters partition (in particular we can underline that Group 2 and Group 3 connect together, since R5 is in common):

1. Cluster 1: (R1-R2), (R1-R3), (R1-R10)
2. Cluster 2: (R2-R4), (R2-R5), (R3-R5), (R3-R9)
3. Cluster 3: (R4-R6), (R4-R7)
4. Cluster 4: (R5-R8)

In the end the following 4 Clusters are obtained:


Cluster 1
R2
R1
R3
R10


Cluster 2
R2
R4
R3
R5
R9


Cluster 3
R6
R4
R7

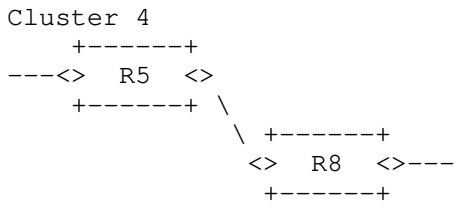


Figure 3: Clusters example

There are Clusters with more than 2 nodes and two-nodes Clusters. In the two-nodes Clusters the loss is on the link (Cluster 4). In more-than-2-nodes Clusters the loss is on the Cluster but we cannot know in which link (Cluster 1, 2, 3).

In this way the calculation of packet loss can be made on Cluster basis. Note that CIR(Committed Information Rate) and EIR(Excess Information Rate) can also be deduced on Cluster basis.

Obviously, by combining some Clusters in a new connected subnetwork (called Super Cluster) the Packet Loss Rule is still true.

In this way in a very large network there is no need to configure detailed filter criteria to inspect the traffic. You can check multipoint network and only in case of problems you can go deep with a step-by-step cluster analysis, but only for the cluster or combination of clusters where the problem happens.

7. Timing Aspects

The mark switching approach based on a fixed timer is considered in this document.

So, if we analyze a multipoint-to-multipoint path with more than one marking node, it is important to recognize the reference measurement interval. In general the measurement interval for describing the results is the interval of the marking node that is more aligned with the start of the measurement, as reported in the following figure.

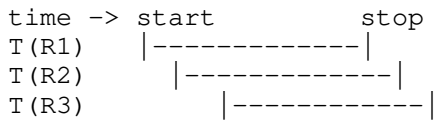


Figure 4: Measurement Interval

T(R1) is the measurement interval and this is essential in order to be compatible and make comparison with other active/passive/hybrid Packet Loss metrics.

That is why, when we expand to multipoint-to-multipoint flows, we have to consider that all source nodes mark the traffic.

Regarding the timing aspects of the methodology, RFC 8321 [RFC8321] already describes two contributions that are taken into account: the clock error between network devices and the network delay between measurement points.

But we should now consider an additional contribution. Since all source nodes mark the traffic, the source measurement intervals can be of different lengths and with different offsets and this mismatch m can be added to d, as shown in figure.

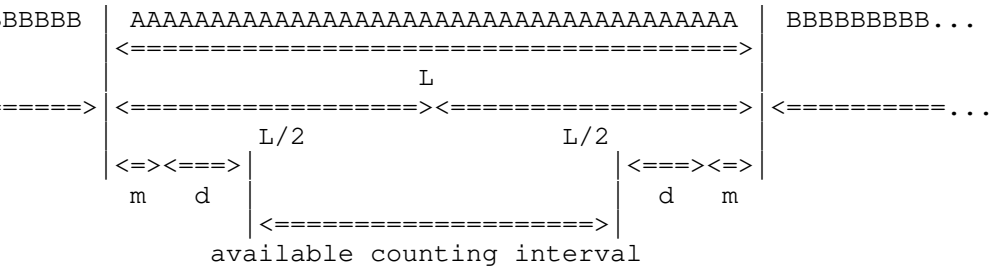


Figure 5: Timing Aspects for Multipoint paths

So the misalignment between the marking source routers gives an additional constraint and the value of m is added to d (that already includes clock error and network delay).

In the end, the condition that must be satisfied to enable the method to function properly is that the available counting interval must be > 0, and that means: L - 2m - 2d > 0 for each measurement point on the multipoint path. Therefore, the mismatch between measurement intervals must satisfy this condition.

8. Multipoint Delay and Delay Variation

The same line of reasoning can be applied to Delay and Delay Variation. It is important to highlight that both delay and delay variation measurements make sense in a multipoint path. The Delay Variation is calculated by considering the same packets selected for measuring the Delay.

In general, it is possible to perform delay and delay variation measurements on multipoint paths basis or on single packets basis:

- Delay measurements on multipoint paths basis means that the delay value is representative of an entire multipoint path (e.g. whole multipoint network, a cluster or a combination of clusters).

- Delay measurements on single packets basis means that you can use multipoint path just to easily couple packets between inputs and output nodes of a multipoint path, as it is described in the following sections.

8.1. Delay measurements on multipoint paths basis

8.1.1. Single Marking measurement

Mean delay and mean delay variation measurements can also be generalized to the case of multipoint flows. It is possible to compute the average one-way delay of packets, in one block, in a cluster or in the entire monitored network.

The average latency can be measured as the difference between the weighted averages of the mean timestamps of the sets of output and input nodes.

8.2. Delay measurements on single packets basis

8.2.1. Single and Double Marking measurement

Delay and delay variation measurements relative to only one picked packet per period (both single and double marked) can be performed in the Multipoint scenario with some limitations:

Single marking based on the first/last packet of the interval would not work, because it would not be possible to agree on the first packet of the interval.

Double marking or multiplexed marking would work, but each measurement would only give information about the delay of a single path. However, by repeating the measurement multiple

times, it is possible to get information about all the paths in the multipoint flow. This can be done in case of point-to-multipoint path but it is more difficult to achieve in case of multipoint-to-multipoint path because of the multiple source routers.

if we would perform a delay measurement for more than one picked packet in the same marking period and, especially, if we want to get delay mesurements on multipoint-to-multipoint basis, both single and double marking method are not useful in the Multipoint scenario, since they would not be representative of the entire flow. The packets can follow different paths with various delays and in general it can be very difficult to recognize marked packets in a multipoint-to-multipoint path especially in case they are more than one per period.

A desirable option is to monitor simultaneously all the paths of a multipoint path in the same marking period and, for this purpose, hashing can be used as reported in the next Section.

8.2.2. Hashing selection method

RFC 5474 [RFC5474] and RFC 5475 [RFC5475] introduce sampling and filtering techniques for IP Packet Selection.

The hash-based selection methodologies for delay measurement can work in a multipoint-to-multipoint path and can be used both coupled to mean delay or stand alone.

[I-D.mizrahi-ippm-compact-alternate-marking] introduces how to use the Hash method combined with alternate marking method for point-to-point flows. It is also called Mixed Hashed Marking: the coupling of marking method and hashing technique is very useful because the marking batches anchor the samples selected with hashing and this simplifies the correlation of the hashing packets along the path.

It is possible to use a basic hash or a dynamic hash method. One of the challenges of the basic approach is that the frequency of the sampled packets may vary considerably. For this reason the dynamic approach has been introduced for point-to-point flow in order to have the desired and almost fixed number of samples for each measurement period. In the hash-based sampling, alternate marking is used to create periods, so that hash-based samples are divided into batches, allowing to anchor the selected samples to their period. Moreover in the dynamic hash-based sampling, by dynamically adapting the length of the hash value, the number of samples is bounded in each marking period. This can be realized by choosing the maximum number of samples (NMAX) to be catched in a marking period. The algorithm

starts with only few hash bits, that permit to select a greater percentage of packets (e.g. with 0 bit of hash all the packets are sampled, with 1 bit of hash half of the packets are sampled, and so on). When the number of selected packets reaches NMAX, a hashing bit is added. As a consequence, the sampling proceeds at half of the original rate and also the packets already selected that don't match the new hash are discarded. This step can be repeated iteratively. It is assumed that each sample includes the timestamp (used for delay measurement) and the hash value, allowing the management system to match the samples received from the two measurement points. The dynamic process statistically converges at the end of a marking period and the final number of selected samples is between NMAX/2 and NMAX. Therefore, the dynamic approach paces the sampling rate, allowing to bound the number of sampled packets per sampling period.

In a multipoint environment the behaviour is similar to point-to point flow. In particular, in the context of multipoint-to-multipoint flow, the dynamic hash could be the solution to perform delay measurements on specific packets and to overcome the single and double marking limitations.

The management system receives the samples including the timestamps and the hash value from all the MPs, and this happens both for point-to-point and for multipoint-to-multipoint flow. Then the longest hash used by MPs is deduced and it is applied to couple timestamps of same packets of 2 MPs of a point-to-point path or of input and output MPs of a Cluster (or a Super Cluster or the entire network). But some considerations are needed: if there isn't packet loss the set of input samples is always equal to the set of output samples. In case of packet loss the set of output samples can be a subset of input samples but the method still works because, at the end, it is easy to couple the input and output timestamps of each catched packet using the hash (in particular the "unused part of the hash" that should be different for each packet).

In summary, the basic hash is logically similar to the double marking method, and in case of point-to-point path double marking and basic hash selection are equivalent. The dynamic approach scales the number of measurements per interval, and it would seem that double marking would also work well if we reduced the interval length, but this can be done only for point-to-point path and not for multipoint path, where we cannot couple the picked packets in a multipoint paths. So, in general, if we want to get delay mesurements on multipoint-to-multipoint path basis and want to select more than one packet per period, double marking cannot be used because we could not be able to couple the picked packets between input and output nodes. On the other hand we can do that by using hashing selection.

9. An SDN enabled Performance Management

The Multipoint Alternate Marking framework that is introduced in this document adds flexibility to PM because it can reduce the order of magnitude of the packet counters. This allows an SDN Orchestrator to supervise, control and manage PM in large networks.

The monitoring network can be considered as a whole or can be split in Clusters, that are the smallest subnetworks (group-to-group segments), maintaining the packet loss property for each subnetwork. They can also be combined in new connected subnetworks at different levels depending on the detail we want to achieve.

An SDN Controller can calibrate Performance Measurements. It can start without examining in depth. In case of necessity (packet loss is measured or the delay is too high), the filtering criteria could be immediately specified more in order to perform a partition of the network by using Clusters and/or different combinations of Clusters. In this way the problem can be localized in a specific Cluster or in a single combination of Clusters and a more detailed analysis can be performed step-by-step by successive approximation up to a point-to-point flow detailed analysis.

In addition an SDN Controller could also collect the measurement history.

10. Examples of application

There are three application fields where it may be useful to take into consideration the Multipoint Alternate Marking:

- VPN: The IP traffic is selected on IP source basis in both directions. At the end point WAN interface all the output traffic is counted in a single flow. The input traffic is composed by all the other flows aggregated for source address. So, by considering n end-points, the monitored flows are n (each flow with 1 ingress point and (n-1) egress points) instead of n*(n-1) flows (each flow, with 1 ingress point and 1 egress point);

- Mobile Backhaul: LTE traffic is selected, in the Up direction, by the EnodeB source address and, in Down direction, by the EnodeB destination address because the packets are sent from the Mobile Packet Core to the EnodeB. So the monitored flow is only one per EnodeB in both directions;

- OTT(Over The Top) services: The traffic is selected, in the Down direction by the source addresses of the packets sent by OTT Servers. In the opposite direction (Up) by the destination IP

addresses of the same Servers. So the monitoring is based on a single flow per OTT Servers in both directions.

11. Security Considerations

This document specifies a method to perform measurements that does not directly affect Internet security nor applications that run on the Internet. However, implementation of this method must be mindful of security and privacy concerns, as explained in RFC 8321 [RFC8321].

12. Acknowledgements

The authors would like to thank Al Morton, Tal Mizrahi, Rachel Huang for the precious contribution.

13. IANA Considerations

tbc

14. References

14.1. Normative References

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

[RFC5644] Stephan, E., Liang, L., and A. Morton, "IP Performance Metrics (IPPM): Spatial and Multicast", RFC 5644, DOI 10.17487/RFC5644, October 2009, <https://www.rfc-editor.org/info/rfc5644>.

[RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <https://www.rfc-editor.org/info/rfc8321>.

14.2. Informative References

[I-D.amf-ippm-route] Alvarez-Hamelin, J., Morton, A., and J. Fabini, "Advanced Unidirectional Route Assessment", draft-amf-ippm-route-01 (work in progress), October 2017.

[I-D.mizrahi-ippm-compact-alternate-marking]
Mizrahi, T., Arad, C., Fioccola, G., Cociglio, M., Chen, M., Zheng, L., and G. Mirsky, "Compact Alternate Marking Methods for Passive and Hybrid Performance Monitoring", draft-mizrahi-ippm-compact-alternate-marking-01 (work in progress), March 2018.

[RFC5474] Duffield, N., Ed., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, DOI 10.17487/RFC5474, March 2009, <https://www.rfc-editor.org/info/rfc5474>.

[RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, DOI 10.17487/RFC5475, March 2009, <https://www.rfc-editor.org/info/rfc5475>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

Authors' Addresses

Giuseppe Fioccola (editor)
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: giuseppe.fioccola@telecomitalia.it

Mauro Cociglio
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: mauro.cociglio@telecomitalia.it

Amedeo Sapio
Politecnico di Torino
Corso Duca degli Abruzzi, 24
Torino 10129
Italy

Email: amedeo.sapio@polito.it

Riccardo Sisto
Politecnico di Torino
Corso Duca degli Abruzzi, 24
Torino 10129
Italy

Email: riccardo.sisto@polito.it

Network Working Group
Internet-Draft
Updates: 2330 (if approved)
Intended status: Informational
Expires: January 1, 2019

A. Morton
AT&T Labs
J. Fabini
TU Wien
N. Elkins
Inside Products, Inc.
M. Ackermann
Blue Cross Blue Shield of Michigan
V. Hegde
Consultant
June 30, 2018

IPv6, IPv4 and Coexistence Updates for IPPM's Active Metric Framework
draft-ietf-ippm-2330-ipv6-06

Abstract

This memo updates the IP Performance Metrics (IPPM) Framework RFC 2330 with new considerations for measurement methodology and testing. It updates the definition of standard-formed packets in RFC 2330 to include IPv6 packets, deprecates the definition of minimal IP packet, and augments distinguishing aspects of packets, referred to as Type-P for test packets in RFC 2330. This memo identifies that IPv4-IPv6 co-existence can challenge measurements within the scope of the IPPM Framework. Example use cases include, but are not limited to IPv4-IPv6 translation, NAT, or protocol encapsulation. IPv6 header compression and use of IPv6 over Low-Power Wireless Area Networks (6LoWPAN) are considered and excluded from the standard-formed packet evaluation.

Requirements Language

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

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 https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

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. It has been updated in the area of metric composition [RFC5835], and in several areas related to active stream measurement of modern networks with reactive properties [RFC7312].

The IPPM framework [RFC2330] recognized (in section 13) that many aspects of IP packets can influence its processing during transfer across the network.

In Section 15 of [RFC2330], the notion of a "standard-formed" packet is defined. However, the definition was never updated to include IPv6, as the original authors originally desired to do.

In particular, IPv6 Extension Headers and protocols which use IPv6 header compression are growing in use. This memo seeks to provide the needed updates.

2. Scope

The purpose of this memo is to expand the coverage of IPPM metrics to include IPv6, and to highlight additional aspects of test packets and make them part of the IPPM performance metric framework.

The scope is to update key sections of [RFC2330], adding considerations that will aid the development of new measurement methodologies intended for today's IP networks. Specifically, this memo expands the Type-P examples in section 13 of [RFC2330] and expands the definition (in section 15 of [RFC2330]) of a standard-formed packet to include IPv6 header aspects and other features.

Other topics in [RFC2330] which might be updated or augmented are deferred to future work. This includes the topics of passive and various forms of hybrid active/passive measurements.

3. Packets of Type-P

A fundamental property of many Internet metrics is that the measured value of the metric depends on characteristics of the IP packet(s) used to make the measurement. Potential influencing factors include IP header fields and their values, but also higher-layer protocol headers and their values. Consider an IP-connectivity metric: one obtains different results depending on whether one is interested in connectivity for packets destined for well-known TCP ports or unreserved UDP ports, or those with invalid IPv4 checksums, or those

with TTL or Hop Limit of 16, for example. In some circumstances these distinctions will result in special treatment of packets in intermediate nodes and end systems (for example, if Diffserv [RFC2474], ECN [RFC3168], Router Alert [RFC6398], Hop-by-hop extensions [RFC7045], or Flow Labels [RFC6437] are used, or in the presence of firewalls or RSVP reservations).

Because of this distinction, we introduce the generic notion of a "packet of Type-P", where in some contexts P will be explicitly defined (i.e., exactly what type of packet we mean), partially defined (e.g., "with a payload of B octets"), or left generic. Thus we may talk about generic IP-Type-P-connectivity or more specific IP-port-HTTP-connectivity. Some metrics and methodologies may be fruitfully defined using generic Type-P definitions which are then made specific when performing actual measurements.

Whenever a metric's value depends on the type of the packets involved in the metric, the metric's name will include either a specific type or a phrase such as "Type-P". Thus we will not define an "IP-connectivity" metric but instead an "IP-Type-P-connectivity" metric and/or perhaps an "IP-port-HTTP-connectivity" metric. This naming convention serves as an important reminder that one must be conscious of the exact type of traffic being measured.

If the information constituting Type-P at the Source is found to have changed at the Destination (or at a measurement point between the Source and Destination, as in [RFC5644]), then the modified values MUST be noted and reported with the results. Some modifications occur according to the conditions encountered in transit (such as congestion notification) or due to the requirements of segments of the Source to Destination path. For example, the packet length will change if IP headers are converted to the alternate version/address family, or if optional Extension Headers are added or removed. Even header fields like TTL/Hop Limit that typically change in transit may be relevant to specific tests. For example Neighbor Discovery Protocol (NDP) [RFC4861] packets are transmitted with Hop Limit value set to 255, and the validity test specifies that the Hop Limit MUST have a value of 255 at the receiver, too. So, while other tests may intentionally exclude the TTL/Hop Limit value from their Type-P definition, for this particular test the correct Hop Limit value is of high relevance and MUST be part of the Type-P definition.

Local policies in intermediate nodes based on examination of IPv6 Extension Headers may affect measurement repeatability. If intermediate nodes follow the recommendations of [RFC7045], repeatability may be improved to some degree.

A closely related note: it would be very useful to know if a given Internet component (like host, link, or path) treats equally a class C of different types of packets. If so, then any one of those types of packets can be used for subsequent measurement of the component. This suggests we devise a metric or suite of metrics that attempt to determine class C (a designation which has no relationship to address assignments, of course).

Load balancing over parallel paths is one particular example where such a class C would be more complex to determine in IPPM measurements. Load balancers and routers often use flow identifiers, computed as hashes of (specific parts of) the packet header, for deciding among the available parallel paths a packet will traverse. Packets with identical hashes are assigned to the same flow and forwarded to the same resource in the load balancer's (or router's) pool. The presence of a load balancer on the measurement path, as well as the specific headers and fields that are used for the forwarding decision, are not known when measuring the path as a black-box. Potential assessment scenarios include the measurement of one of the parallel paths, and the measurement of all available parallel paths that the load balancer can use. Knowledge of a load balancer's flow definition (alternatively: its class C specific treatment in terms of header fields in scope of hash operations) is therefore a prerequisite for repeatable measurements. A path may have more than one stage of load balancing, adding to class C definition complexity.

4. Standard-Formed Packets

Unless otherwise stated, all metric definitions that concern IP packets include an implicit assumption that the packet is *standard-formed*. A packet is standard-formed if it meets all of the following REQUIRED criteria:

+ It includes a valid IP header: see below for version-specific criteria.

+ It is not an IP fragment.

+ The Source and Destination addresses correspond to the intended Source and Destination, including Multicast Destination addresses.

+ If a transport header is present, it contains a valid checksum and other valid fields.

For an IPv4 ([RFC0791] and updates) packet to be standard-formed, the following additional criteria are REQUIRED:

- The version field is 4
- The Internet Header Length (IHL) value is >= 5; the checksum is correct.
- Its total length as given in the IPv4 header corresponds to the size of the IPv4 header plus the size of the payload.
- Either the packet possesses sufficient TTL to travel from the Source to the Destination if the TTL is decremented by one at each hop, or it possesses the maximum TTL of 255.
- It does not contain IP options unless explicitly noted.

For an IPv6 ([RFC8200] and updates) packet to be standard-formed, the following criteria are REQUIRED:

- The version field is 6.
- Its total length corresponds to the size of the IPv6 header (40 octets) plus the length of the payload as given in the IPv6 header.
- The payload length value for this packet (including Extension Headers) conforms to the IPv6 specifications.
- Either the packet possesses sufficient Hop Limit to travel from the Source to the Destination if the Hop Limit is decremented by one at each hop, or it possesses the maximum Hop Limit of 255.
- Either the packet does not contain IP Extension Headers, or it contains the correct number and type of headers as specified in the packet, and the headers appear in the standard-conforming order (Next Header).
- All parameters used in the header and Extension Headers are found in the IANA Registry of Internet Protocol Version 6 (IPv6) Parameters, specified in [IANA-6P].

Two mechanisms require some discussion in the context of standard-formed packets, namely IPv6 over Low-Power Wireless Area Networks (6LowPAN, [RFC4944]) and Robust Header Compression (ROHC, [RFC3095]). IPv6 over Low-Power Wireless Area Networks (6LowPAN), as defined in [RFC4944] and updated by [RFC6282] with header compression and [RFC6775] with neighbor discovery optimizations, proposes solutions for using IPv6 in resource-constrained environments. An adaptation layer enables the transfer of IPv6 packets over networks having a MTU smaller than the minimum IPv6 MTU. Fragmentation and re-assembly of

IPv6 packets, as well as the resulting state that would be stored in intermediate nodes, poses substantial challenges to measurements. Likewise, ROHC operates statefully in compressing headers on subpaths, storing state in intermediate hosts. The modification of measurement packets' Type-P by ROHC and 6LowPAN, as well as requirements with respect to the concept of standard-formed packets for these two protocols requires substantial work. Because of these reasons we consider ROHC and 6LowPAN packets to be out of the scope for the standard-formed packet evaluation.

The topic of IPv6 Extension Headers brings current controversies into focus as noted by [RFC6564] and [RFC7045]. However, measurement use cases in the context of the IPPM framework like in-situ OAM [I-D.ietf-ippm-ioam-data] in enterprise environments can benefit from inspection, modification, addition or deletion of IPv6 extension headers in hosts along the measurement path.

[RFC8250] endorses the use of IPv6 Destination Option for measurement purposes, consistent with other approved IETF specifications.

The following additional considerations apply when IPv6 Extension Headers are present:

- Extension Header inspection: Some intermediate nodes may inspect Extension Headers or the entire IPv6 packet while in transit. In exceptional cases, they may drop the packet or route via a sub-optimal path, and measurements may be unreliable or unrepeatable. The packet (if it arrives) may be standard-formed, with a corresponding Type-P.

- Extension Header modification: In Hop-by-Hop headers, some TLV encoded options may be permitted to change at intermediate nodes while in transit. The resulting packet may be standard-formed, with a corresponding Type-P.

- Extension Header insertion or deletion: Although such behavior is not endorsed by current standards, it is possible that Extension Headers could be added to, or removed from the header chain. The resulting packet may be standard-formed, with a corresponding Type-P. This point simply encourages measurement system designers to be prepared for the unexpected, and to notify users when such events occur. There are issues with Extension Header insertion and deletion of course, such as exceeding the path MTU due to insertion, etc.

- A change in packet length (from the corresponding packet observed at the Source) or header modification is a significant factor in

Internet measurement, and REQUIRES a new Type-P to be reported with the test results.

It is further REQUIRED that if a packet is described as having a "length of B octets", then 0 <= B <= 65535; and if B is the payload length in octets, then B <= (65535-IP header size in octets, including any Extension Headers). The jumbograms defined in [RFC2675] are not covered by the above length analysis, but if the IPv6 Jumbogram Payload Hop-by-Hop Option Header is present, then a packet with corresponding length MUST be considered standard-formed. In practice, the path MTU will restrict the length of standard-formed packets that can successfully traverse the path. Path MTU Discovery for IP version 6 (PMTUD, [RFC8201]) or Packetization Layer Path MTU Discovery (PLPMTUD, [RFC4821]) is recommended to prevent fragmentation.

So, for example, one might imagine defining an IP connectivity metric as "IP-type-P-connectivity for standard-formed packets with the IP Diffserv field set to 0", or, more succinctly, "IP-type-P-connectivity with the IP Diffserv Field set to 0", since standard-formed is already implied by convention. Changing the contents of a field, such as the Diffserv Code Point, ECN bits, or Flow Label may have a profound affect on packet handling during transit, but does not affect a packet's status as standard-formed. Likewise, the addition, modification, or deletion of extension headers may change the handling of packets in transit hosts.

[RFC2330] defines the "minimal IP packet from A to B" as a particular type of standard-formed packet often useful to consider. When defining IP metrics no packet smaller or simpler than this can be transmitted over a correctly operating IP network. However, the concept of the minimal IP packet has not been employed (since typical active measurement systems employ a transport layer and a payload) and its practical use is limited. Therefore, this memo deprecates the concept of the "minimal IP packet from A to B".

5. NAT, IPv4-IPv6 Transition and Compression Techniques

This memo adds the key considerations for utilizing IPv6 in two critical conventions of the IPPM Framework, namely packets of Type-P and standard-formed packets. The need for co-existence of IPv4 and IPv6 has originated transitioning standards like the Framework for IPv4/IPv6 Translation in [RFC6144] or IP/ICMP Translation Algorithms in [RFC7915] and [RFC7757].

The definition and execution of measurements within the context of the IPPM Framework is challenged whenever such translation mechanisms are present along the measurement path. In particular use cases like

IPv4-IPv6 translation, NAT, protocol encapsulation, or IPv6 header compression may result in modification of the measurement packet's Type-P along the path. All these changes MUST be reported. Example consequences include, but are not limited to:

- Modification or addition of headers or header field values in intermediate nodes. IPv4-IPv6 transitioning or IPv6 header compression mechanisms may result in changes of the measurement packets' Type-P, too. Consequently, hosts along the measurement path may treat packets differently because of the Type-P modification. Measurements at observation points along the path may also need extra context to uniquely identify a packet.

- Network Address Translators (NAT) on the path can have unpredictable impact on latency measurement (in terms of the amount of additional time added), and possibly other types of measurements. It is not usually possible to control this impact (as testers may not have any control of the underlying network or middleboxes). There is a possibility that stateful NAT will lead to unstable performance for a flow with specific Type-P, since state needs to be created for the first packet of a flow, and state may be lost later if the NAT runs out of resources. However, this scenario does not invalidate the Type-P for testing - for example the purpose of a test might be exactly to quantify the NAT's impact on delay variation. The presence of NAT may mean that the measured performance of Type-P will change between the source and the destination. This can cause an issue when attempting to correlate measurements conducted on segments of the path that include or exclude the NAT. Thus, it is a factor to be aware of when conducting measurements.

- Variable delay due to internal state. One side effect of changes due to IPv4-IPv6 transitioning mechanisms is the variable delay that intermediate nodes spend for header modifications. Similar to NAT the allocation of internal state and establishment of context within intermediate nodes may cause variable delays, depending on the measurement stream pattern and position of a packet within the stream. For example the first packet in a stream will typically trigger allocation of internal state in an intermediate IPv4-IPv6 transition host. Subsequent packets can benefit from lower processing delay due to the existing internal state. However, large inter-packet delays in the measurement stream may result in the intermediate host deleting the associated state and needing to re-establish it on arrival of another stream packet. It is worth noting that this variable delay due to internal state allocation in intermediate nodes can be an explicit use case for measurements.

- o Variable delay due to packet length. IPv4-IPv6 transitioning or header compression mechanisms modify the length of measurement packets. The modification of the packet size may or may not change the way how the measurement path treats the packets.

6. Security Considerations

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

When considering privacy of those involved in measurement or those whose traffic is measured, the sensitive information available to potential observers is greatly reduced when using active techniques which are within this scope of work. Passive observations of user traffic for measurement purposes raise many privacy issues. We refer the reader to the privacy considerations described in the Large Scale Measurement of Broadband Performance (LMAP) Framework [RFC7594], which covers active and passive techniques.

7. IANA Considerations

This memo makes no requests of IANA.

8. Acknowledgements

The authors thank Brian Carpenter for identifying the lack of IPv6 coverage in IPPM's Framework, and for listing additional distinguishing factors for packets of Type-P. Both Brian and Fred Baker discussed many of the interesting aspects of IPv6 with the co-authors, leading to a more solid first draft: thank you both. Thanks to Bill Jouris for an editorial pass through the pre-00 text. As we completed our journey, Nevil Brownlee, Mike Heard, Spencer Dawkins, Warren Kumari, and Suresh Krishnan all contributed useful suggestions.

9. References

9.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/info/rfc791>.

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

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <https://www.rfc-editor.org/info/rfc2330>.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <https://www.rfc-editor.org/info/rfc2474>.

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, DOI 10.17487/RFC2675, August 1999, <https://www.rfc-editor.org/info/rfc2675>.

[RFC3095] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, DOI 10.17487/RFC3095, July 2001, <https://www.rfc-editor.org/info/rfc3095>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <https://www.rfc-editor.org/info/rfc3168>.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <https://www.rfc-editor.org/info/rfc4821>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <https://www.rfc-editor.org/info/rfc4944>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5644] Stephan, E., Liang, L., and A. Morton, "IP Performance Metrics (IPPM): Spatial and Multicast", RFC 5644, DOI 10.17487/RFC5644, October 2009, <https://www.rfc-editor.org/info/rfc5644>.

[RFC5835] Morton, A., Ed. and S. Van den Berghe, Ed., "Framework for Metric Composition", RFC 5835, DOI 10.17487/RFC5835, April 2010, <https://www.rfc-editor.org/info/rfc5835>.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, DOI 10.17487/RFC6144, April 2011, <https://www.rfc-editor.org/info/rfc6144>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <https://www.rfc-editor.org/info/rfc6282>.

[RFC6398] Le Faucheur, F., Ed., "IP Router Alert Considerations and Usage", BCP 168, RFC 6398, DOI 10.17487/RFC6398, October 2011, <https://www.rfc-editor.org/info/rfc6398>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <https://www.rfc-editor.org/info/rfc6437>.

[RFC6564] Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., and M. Bhatia, "A Uniform Format for IPv6 Extension Headers", RFC 6564, DOI 10.17487/RFC6564, April 2012, <https://www.rfc-editor.org/info/rfc6564>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <https://www.rfc-editor.org/info/rfc7045>.

[RFC7312] Fabini, J. and A. Morton, "Advanced Stream and Sampling Framework for IP Performance Metrics (IPPM)", RFC 7312, DOI 10.17487/RFC7312, August 2014, <https://www.rfc-editor.org/info/rfc7312>.

[RFC7757] Anderson, T. and A. Leiva Popper, "Explicit Address Mappings for Stateless IP/ICMP Translation", RFC 7757, DOI 10.17487/RFC7757, February 2016, <https://www.rfc-editor.org/info/rfc7757>.

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

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

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

[RFC8250] Elkins, N., Hamilton, R., and M. Ackermann, "IPv6 Performance and Diagnostic Metrics (PDM) Destination Option", RFC 8250, DOI 10.17487/RFC8250, September 2017, <https://www.rfc-editor.org/info/rfc8250>.

9.2. Informative References

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., daniel.bernier@bell.ca, d., and J. Lemon, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-03 (work in progress), June 2018.

[IANA-6P] IANA, "IANA Internet Protocol Version 6 (IPv6) Parameters", Internet Assigned Numbers Authority https://www.iana.org/assignments/ipv6-parameters, January 2018.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/info/rfc7594>.

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/

Joachim Fabini
TU Wien
Gusshausstrasse 25/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/

Nalini Elkins
Inside Products, Inc.
Carmel Valley, CA 93924
USA

Email: nalini.elkins@insidethestack.com

Michael S. Ackermann
Blue Cross Blue Shield of Michigan

Email: mackermann@bcbsm.com

Vinayak Hegde
Consultant
Brahma Sun City, Wadgaon-Sheri
Pune, Maharashtra 411014
INDIA

Phone: +91 9449834401
Email: vinayakh@gmail.com
URI: http://www.vinayakhegde.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 10, 2019

A. Morton
AT&T Labs
M. Bagnulo
UC3M
P. Eardley
BT
K. D'Souza
AT&T Labs

December 7, 2018

Initial Performance Metric Registry Entries
draft-ietf-ippm-initial-registry-09

Abstract

This memo defines the Initial Entries for the Performance Metrics Registry. This version includes:

* removed sections which only contained examples, or a blank outine for new metric entries.

* removed remaining comments (did not require action).

Requirements Language

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

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 https://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 June 10, 2019.

Table of Contents

1. Introduction

Note: Efforts to synchronize structure and terminology with [I-D.ietf-ippm-metric-registry] will likely be incomplete until both drafts are stable.

This memo proposes an initial set of entries for the Performance Metric Registry. It uses terms and definitions from the IPPM literature, primarily [RFC2330].

Although there are several standard templates for organizing specifications of performance metrics (see [RFC2679] for an example of the traditional IPPM template, based to large extent on the

Benchmarking Methodology Working Group's traditional template in [RFC1242], and see [RFC6390] for a similar template), none of these templates were intended to become the basis for the columns of an IETF-wide registry of metrics. While examining aspects of metric specifications which need to be registered, it became clear that none of the existing metric templates fully satisfies the particular needs of a registry.

Therefore, [I-D.ietf-ippm-metric-registry] defines the overall format for a Performance Metric Registry. Section 5 of [I-D.ietf-ippm-metric-registry] also gives guidelines for those requesting registration of a Metric, that is the creation of entry(s) in the Performance Metric Registry: "In essence, there needs to be evidence that a candidate Registered Performance Metric has significant industry interest, or has seen deployment, and there is agreement that the candidate Registered Performance Metric serves its intended purpose." The process in [I-D.ietf-ippm-metric-registry] also requires that new entries are administered by IANA through Expert Review, which will ensure that the metrics are tightly defined.

2. Scope

This document defines the initial set of Performance Metrics Registry entries, for which IETF approval (following development in the IP Performance Metrics (IPPM) Working Group) will satisfy the requirement for Expert Review. Most are Active Performance Metrics, which are based on RFCs prepared in the IPPM working group of the IETF, according to their framework [RFC2330] and its updates.

3. Registry Categories and Columns

This section provides the categories and columns of the registry, for easy reference. An entry (row) therefore gives a complete description of a Registered Metric.

Registry Categories and Columns, shown as

	Category	
	Column	Column

Summary

Identifier	Name	URIs	Desc.	Reference	Change Controller	Ver

Metric Definition

Reference Definition	Fixed Parameters

Method of Measurement

Reference Method	Packet Stream Generation	Traffic Filter	Sampling Distribution	Run-time Parameters	Role

Output

Type	Reference Definition	Units	Calibration

Administrative Information

Status	Request	Rev	Rev.Date

Comments and Remarks

4. UDP Round-trip Latency and Loss Registry Entries

This section specifies an initial registry entry for the UDP Round-trip Latency, and another entry for UDP Round-trip Loss Ratio.

Note: Each Registry entry only produces a "raw" output or a statistical summary. To describe both "raw" and one or more statistics efficiently, the Identifier, Name, and Output Categories can be split and a single section can specify two or more closely-related metrics. This section specifies two Registry entries with many common columns. See Section 7 for an example specifying multiple Registry entries with many common columns.

All column entries beside the ID, Name, Description, and Output Reference Method categories are the same, thus this section proposes two closely-related registry entries. As a result, IANA is also asked to assign corresponding URNs and URLs to each Named Metric.

4.1. Summary

This category includes multiple indexes to the registry entry: the element ID and metric name.

4.1.1. ID (Identifier)

IANA is asked to assign different numeric identifiers to each of the two Named Metrics.

4.1.2. Name

RTDelay_Active_IP-UDP-Periodic_RFCXXXXsecY_Seconds_95Percentile

RTLoss_Active_IP-UDP-Periodic_RFCXXXXsecY_Percent_LossRatio

4.1.3. URIs

URN: Prefix urn:ietf:metrics:perf:<name>

URL: http://<TBD by IANA>/<name>

4.1.4. Description

RTDelay: This metric assesses the delay of a stream of packets exchanged between two hosts (which are the two measurement points), and the Output is the Round-trip delay for all successfully exchanged packets expressed as the 95th percentile of their conditional delay distribution.

RTLoss: This metric assesses the loss ratio of a stream of packets exchanged between two hosts (which are the two measurement points), and the Output is the Round-trip loss ratio for all successfully exchanged packets expressed as a percentage.

4.1.5. Change Controller

IETF

4.1.6. Version (of Registry Format)

1.0

4.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

4.2.1. Reference Definition

Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.

[RFC2681]

Section 2.4 of [RFC2681] provides the reference definition of the singleton (single value) Round-trip delay metric. Section 3.4 of [RFC2681] provides the reference definition expanded to cover a multi-singleton sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

Note that although the [RFC2681] definition of "Round-trip-Delay between Src and Dst" is directionally ambiguous in the text, this metric tightens the definition further to recognize that the host in the "Src" role will send the first packet to "Dst", and ultimately receive the corresponding return packet from "Dst" (when neither are lost).

Finally, note that the variable "dT" is used in [RFC2681] to refer to the value of Round-trip delay in metric definitions and methods. The variable "dT" has been re-used in other IPPM literature to refer to different quantities, and cannot be used as a global variable name.

Morton, A., "Round-trip Packet Loss Metrics", RFC 6673, August 2012.

[RFC6673]

Both delay and loss metrics employ a maximum waiting time for received packets, so the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 6.1 of [RFC6673].

4.2.2. Fixed Parameters

Type-P as defined in Section 13 of [RFC2330]:

- IPv4 header values:
  - DSCP: set to 0
  - TTL: set to 255
  - Protocol: Set to 17 (UDP)
- IPv6 header values:

    * DSCP: set to 0

    * Hop Count: set to 255

    * Protocol: Set to 17 (UDP)

  o UDP header values:

    * Checksum: the checksum MUST be calculated and included in the header

  o UDP Payload

    * total of 100 bytes

  Other measurement parameters:

  o Tmax: a loss threshold waiting time

    * 3.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

4.3. Method of Measurement

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

4.3.1. Reference Method

The methodology for this metric is defined as Type-P-Round-trip-Delay-Poisson-Stream in section 2.6 of RFC 2681 [RFC2681] and section 3.6 of RFC 2681 [RFC2681] using the Type-P and Tmax defined under Fixed Parameters. However, the Periodic stream will be generated according to [RFC3432].

The reference method distinguishes between long-delayed packets and lost packets by implementing a maximum waiting time for packet arrival. Tmax is the waiting time used as the threshold to declare a packet lost. Lost packets SHALL be designated as having undefined delay, and counted for the RTLoss metric.

The calculations on the delay (RTT) SHALL be performed on the conditional distribution, conditioned on successful packet arrival within Tmax. Also, when all packet delays are stored, the process

which calculates the RTT value MAY enforce the Tmax threshold on stored values before calculations. See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The reference method requires some way to distinguish between different packets in a stream to establish correspondence between sending times and receiving times for each successfully-arriving packet. Sequence numbers or other send-order identification MUST be retained at the Src or included with each packet to disambiguate packet reordering if it occurs.

If a standard measurement protocol is employed, then the measurement process will determine the sequence numbers or timestamps applied to test packets after the Fixed and Runtime parameters are passed to that process. The chosen measurement protocol will dictate the format of sequence numbers and time-stamps, if they are conveyed in the packet payload.

Refer to Section 4.4 of [RFC6673] for expanded discussion of the instruction to "send a Type-P packet back to the Src as quickly as possible" in Section 2.6 of RFC 2681 [RFC2681]. Section 8 of [RFC6673] presents additional requirements which MUST be included in the method of measurement for this metric.

4.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

Section 3 of [RFC3432] prescribes the method for generating Periodic streams using associated parameters.

incT the nominal duration of inter-packet interval, first bit to first bit, with value 0.0200, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms).

dT the duration of the interval for allowed sample start times, with value 1.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms).

T0 the actual start time of the periodic stream, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]).

NOTE: an initiation process with a number of control exchanges resulting in unpredictable start times (within a time interval) may be sufficient to avoid synchronization of periodic streams, and therefore a valid replacement for selecting a start time at random from a fixed interval.

The T0 parameter will be reported as a measured parameter. Parameters incT and dT are Fixed Parameters.

4.3.3. Traffic Filtering (observation) Details

The measured results based on a filtered version of the packets observed, and this section provides the filter details (when present).

NA

4.3.4. Sampling Distribution

NA

4.3.5. Run-time Parameters and Data Format

Run-time Parameters are input factors that must be determined, configured into the measurement system, and reported with the results for the context to be complete.

Src the IP address of the host in the Src Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the Dst Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a start time is unspecified and Tf is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Tf a time, the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of

[RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a end time date is ignored and Tf is interpreted as the Duration of the measurement interval.

4.3.6. Roles

Src launches each packet and waits for return transmissions from Dst.

Dst waits for each packet from Src and sends a return packet to Src.

4.4. Output

This category specifies all details of the Output of measurements using the metric.

4.4.1. Type

Percentile -- for the conditional distribution of all packets with a valid value of Round-trip delay (undefined delays are excluded), a single value corresponding to the 95th percentile, as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The percentile = 95, meaning that the reported delay, "95Percentile", is the smallest value of Round-trip delay for which the Empirical Distribution Function (EDF), F(95Percentile) >= 95% of the singleton Round-trip delay values in the conditional distribution. See section 11.3 of [RFC2330] for the definition of the percentile statistic using the EDF.

LossRatio -- the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 6.1 of [RFC6673].

4.4.2. Reference Definition

For all outputs ---

T0 the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

Tf the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of

[RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

TotalPkts the count of packets sent by the Src to Dst during the measurement interval.

For

RTDelay_Active_IP-UDP-Periodic_RFCXXXXsecY_Seconds_95Percentile:

95Percentile The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as

For

RTLoss_Active_IP-UDP-Periodic_RFCXXXXsecY_Percent_LossRatio:

Percentile The numeric value of the result is expressed in units of lost packets to total packets times 100%, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.0000000001.

4.4.3. Metric Units

The 95th Percentile of Round-trip Delay is expressed in seconds.

The Round-trip Loss Ratio is expressed as a percentage of lost packets to total packets sent.

4.4.4. Calibration

Section 3.7.3 of [RFC7679] provides a means to quantify the systematic and random errors of a time measurement. In-situ calibration could be enabled with an internal loopback at the Source host that includes as much of the measurement system as possible, performs address manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface contention. Some portion of the random and systematic error can be characterized this way.

When a measurement controller requests a calibration measurement, the loopback is applied and the result is output in the same format as a normal measurement with additional indication that it is a calibration result.

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

4.5. Administrative items

4.5.1. Status

Current

4.5.2. Requestor

This RFC numner

4.5.3. Revision

1.0

4.5.4. Revision Date

YYYY-MM-DD

4.6. Comments and Remarks

None.

5. Packet Delay Variation Registry Entry

This section gives an initial registry entry for a Packet Delay Variation metric.

Note: If each Registry entry should only produce a "raw" output or a statistical summary, then the "Output" Category can be split and this section can become two closely-related metrics.

5.1. Summary

This category includes multiple indexes to the registry entries, the element ID and metric name.

5.1.1. ID (Identifier)

<insert numeric identifier, an integer>

5.1.2. Name

OWPDV_Active_IP-UDP-Periodic_RFCXXXXsecY_Seconds_95Percentile

5.1.3. URIs

URI: Prefix urn:ietf:metrics:perf:<name>

URL: http://<TBD by IANA>/<name>

5.1.4. Description

An assessment of packet delay variation with respect to the minimum delay observed on the periodic stream, and the Output is expressed as the 95th percentile of the packet delay variation distribution.

5.1.5. Change Controller

IETF

5.1.6. Version (of Registry Format)

1.0

5.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

5.2.1. Reference Definition

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

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

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

Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.[RFC5905]

See sections 2.4 and 3.4 of [RFC3393]. Singleton delay differences measured are referred to by the variable name "ddT" (applicable to

all forms of delay variation). However, this metric entry specifies the PDV form defined in section 4.2 of [RFC5481], where the singleton PDV for packet i is referred to by the variable name "PDV(i)".

5.2.2. Fixed Parameters

- IPv4 header values:
  - DSCP: set to 0
  - TTL: set to 255
  - Protocol: Set to 17 (UDP)
- IPv6 header values:
  - DSCP: set to 0
  - Hop Count: set to 255
  - Protocol: Set to 17 (UDP)
- UDP header values:
  - Checksum: the checksum MUST be calculated and included in the header
- UDP Payload
  - total of 200 bytes

Other measurement parameters:

Tmax: a loss threshold waiting time with value 3.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

F a selection function unambiguously defining the packets from the stream selected for the metric. See section 4.2 of [RFC5481] for the PDV form.

See the Packet Stream generation category for two additional Fixed Parameters.

5.3. Method of Measurement

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

5.3.1. Reference Method

See section 2.6 and 3.6 of [RFC3393] for general singleton element calculations. This metric entry requires implementation of the PDV form defined in section 4.2 of [RFC5481]. Also see measurement considerations in section 8 of [RFC5481].

The reference method distinguishes between long-delayed packets and lost packets by implementing a maximum waiting time for packet arrival. Tmax is the waiting time used as the threshold to declare a packet lost. Lost packets SHALL be designated as having undefined delay.

The calculations on the one-way delay SHALL be performed on the conditional distribution, conditioned on successful packet arrival within Tmax. Also, when all packet delays are stored, the process which calculates the one-way delay value MAY enforce the Tmax threshold on stored values before calculations. See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The reference method requires some way to distinguish between different packets in a stream to establish correspondence between sending times and receiving times for each successfully-arriving packet. Sequence numbers or other send-order identification MUST be retained at the Src or included with each packet to disambiguate packet reordering if it occurs.

If a standard measurement protocol is employed, then the measurement process will determine the sequence numbers or timestamps applied to test packets after the Fixed and Runtime parameters are passed to that process. The chosen measurement protocol will dictate the format of sequence numbers and time-stamps, if they are conveyed in the packet payload.

5.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

Section 3 of [RFC3432] prescribes the method for generating Periodic streams using associated parameters.

incT the nominal duration of inter-packet interval, first bit to first bit, with value 0.0200, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms).

dT the duration of the interval for allowed sample start times, with value 1.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms).

T0 the actual start time of the periodic stream, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]).

NOTE: an initiation process with a number of control exchanges resulting in unpredictable start times (within a time interval) may be sufficient to avoid synchronization of periodic streams, and therefore a valid replacement for selecting a start time at random from a fixed interval.

The T0 parameter will be reported as a measured parameter. Parameters incT and dT are Fixed Parameters.

5.3.3. Traffic Filtering (observation) Details

NA

5.3.4. Sampling Distribution

NA

5.3.5. Run-time Parameters and Data Format

Src the IP address of the host in the Src Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the Dst Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of

[RFC2330]. When T0 is "all-zeros", a start time is unspecified and Tf is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Tf a time, the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a end time date is ignored and Tf is interpreted as the Duration of the measurement interval.

5.3.6. Roles

5.4. Output

This category specifies all details of the Output of measurements using the metric.

5.4.1. Type

Percentile -- for the conditional distribution of all packets with a valid value of one-way delay (undefined delays are excluded), a single value corresponding to the 95th percentile, as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The percentile = 95, meaning that the reported delay, "95Percentile", is the smallest value of one-way PDV for which the Empirical Distribution Function (EDF), F(95Percentile) >= 95% of the singleton one-way PDV values in the conditional distribution. See section 11.3 of [RFC2330] for the definition of the percentile statistic using the EDF.

5.4.2. Reference Definition

T0 the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

Tf the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

95Percentile The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction

digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

5.4.3. Metric Units

The 95th Percentile of one-way PDV is expressed in seconds.

5.4.4. Calibration

Section 3.7.3 of [RFC7679] provides a means to quantify the systematic and random errors of a time measurement. In-situ calibration could be enabled with an internal loopback that includes as much of the measurement system as possible, performs address manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface contention. Some portion of the random and systematic error can be characterized this way.

For one-way delay measurements, the error calibration must include an assessment of the internal clock synchronization with its external reference (this internal clock is supplying timestamps for measurement). In practice, the time offsets of clocks at both the source and destination are needed to estimate the systematic error due to imperfect clock synchronization (the time offsets are smoothed, thus the random variation is not usually represented in the results).

time_offset The time value of the result is expressed in units of seconds, as a signed value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

When a measurement controller requests a calibration measurement, the loopback is applied and the result is output in the same format as a normal measurement with additional indication that it is a calibration result. In any measurement, the measurement function SHOULD report its current estimate of time offset as an indicator of the degree of synchronization.

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

5.5. Administrative items

5.5.1. Status

Current

5.5.2. Requestor

This RFC number

5.5.3. Revision

1.0

5.5.4. Revision Date

YYYY-MM-DD

5.6. Comments and Remarks

Lost packets represent a challenge for delay variation metrics. See section 4.1 of [RFC3393] and the delay variation applicability statement[RFC5481] for extensive analysis and comparison of PDV and an alternate metric, IPDV.

6. DNS Response Latency and Loss Registry Entries

This section gives initial registry entries for DNS Response Latency and Loss from a network user's perspective, for a specific named resource. The metric can be measured repeatedly using different names. RFC 2681 [RFC2681] defines a Round-trip delay metric. We build on that metric by specifying several of the input parameters to precisely define two metrics for measuring DNS latency and loss.

Note to IANA: Each Registry "Name" below specifies a single registry entry, whose output format varies in accordance with the name.

All column entries beside the ID, Name, Description, and Output Reference Method categories are the same, thus this section proposes two closely-related registry entries. As a result, IANA is also asked to assign corresponding URNs and URLs to each Named Metric.

6.1. Summary

This category includes multiple indexes to the registry entries, the element ID and metric name.

6.1.1. ID (Identifier)

<insert numeric identifier, an integer>

IANA is asked to assign different numeric identifiers to each of the two Named Metrics.

6.1.2. Name

RTDNS_Active_IP-UDP-Poisson_RFCXXXXsecY_Seconds_Raw

RLDNS_Active_IP-UDP-Poisson_RFCXXXXsecY_Logical_Raw

6.1.3. URI

URI: Prefix urn:ietf:metrics:perf:<name>

URL: http://<TBD by IANA>/<name>

6.1.4. Description

This is a metric for DNS Response performance from a network user's perspective, for a specific named resource. The metric can be measured repeatedly using different resource names.

RTDNS: This metric assesses the response time, the interval from the query transmission to the response.

RLDNS: This metric indicates that the response was deemed lost. In other words, the response time exceeded the maximum waiting time.

6.1.5. Change Controller

IETF

6.1.6. Version (of Registry Format)

1.0

6.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

6.2.1. Reference Definition

Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987. (and updates)

[RFC1035]

Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.

[RFC2681]

Section 2.4 of [RFC2681] provides the reference definition of the singleton (single value) Round-trip delay metric. Section 3.4 of [RFC2681] provides the reference definition expanded to cover a multi-singleton sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

For DNS Response Latency, the entities in [RFC1035] must be mapped to [RFC2681]. The Local Host with its User Program and Resolver take the role of "Src", and the Foreign Name Server takes the role of "Dst".

Note that although the [RFC2681] definition of "Round-trip-Delay between Src and Dst at T" is directionally ambiguous in the text, this metric tightens the definition further to recognize that the host in the "Src" role will send the first packet to "Dst", and ultimately receive the corresponding return packet from "Dst" (when neither are lost).

Morton, A., "Round-trip Packet Loss Metrics", RFC 6673, August 2012.

[RFC6673]

Both response time and loss metrics employ a maximum waiting time for received responses, so the count of lost packets to total packets sent is the basis for the loss determination as per Section 4.3 of [RFC6673].

6.2.2. Fixed Parameters

Type-P as defined in Section 13 of [RFC2330]:

- IPv4 header values:
  - DSCP: set to 0
  - TTL set to 255

      * Protocol: Set to 17 (UDP)

   o IPv6 header values:

      * DSCP: set to 0

      * Hop Count: set to 255

      * Protocol: Set to 17 (UDP)

   o UDP header values:

      * Source port: 53

      * Destination port: 53

      * Checksum: the checksum must be calculated and included in the header

   o Payload: The payload contains a DNS message as defined in RFC 1035 [RFC1035] with the following values:

      * The DNS header section contains:

         + Identification (see the Run-time column)

         + 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 Fully Qualified Domain Name (FQDN) provided as input for the test, see the Run-time column
  + QTYPE: the query type provided as input for the test, see the Run-time column
  + QCLASS: set to 1 for IN
* The other sections do not contain any Resource Records.

Other measurement parameters:

- Tmax: a loss threshold waiting time (and to help disambiguate queries)
  * 5.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

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

6.3. Method of Measurement

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

6.3.1. Reference Method

The methodology for this metric is defined as Type-P-Round-trip-Delay-Poisson-Stream in section 2.6 of RFC 2681 [RFC2681] and section 3.6 of RFC 2681 [RFC2681] using the Type-P and Timeout defined under Fixed Parameters.

The reference method distinguishes between long-delayed packets and lost packets by implementing a maximum waiting time for packet arrival. Tmax is the waiting time used as the threshold to declare a response packet lost. Lost packets SHALL be designated as having undefined delay and counted for the RLDNS metric.

The calculations on the delay (RTT) SHALL be performed on the conditional distribution, conditioned on successful packet arrival within Tmax. Also, when all packet delays are stored, the process

which calculates the RTT value MAY enforce the Tmax threshold on stored values before calculations. See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The reference method requires some way to distinguish between different packets in a stream to establish correspondence between sending times and receiving times for each successfully-arriving reply.

DNS Messages bearing Queries provide for random ID Numbers in the Identification header field, so more than one query may be launched while a previous request is outstanding when the ID Number is used. Therefore, the ID Number MUST be retained at the Src or included with each response packet to disambiguate packet reordering if it occurs.

IF a DNS response does not arrive within Tmax, the response time RTDNS is undefined, and RLDNS = 1. The Message ID SHALL be used to disambiguate the successive queries that are otherwise identical.

Since the ID Number filed is only 16 bits in length, it places a limit on the number of simultaneous outstanding DNS queries during a stress test from a single Src address.

Refer to Section 4.4 of [RFC6673] for expanded discussion of the instruction to "send a Type-P packet back to the Src as quickly as possible" in Section 2.6 of RFC 2681 [RFC2681]. However, the DNS Server is expected to perform all required functions to prepare and send a response, so the response time will include processing time and network delay. Section 8 of [RFC6673] presents additional requirements which SHALL be included in the method of measurement for this metric.

In addition to operations described in [RFC2681], the Src MUST parse the DNS headers of the reply and prepare the information for subsequent reporting as a measured result, along with the Round-Trip Delay.

6.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

Section 11.1.3 of RFC 2681 [RFC2330] provides three methods to generate Poisson sampling intervals. The reciprocal of lambda is the

average packet rate, thus the Run-time Parameter is Reciprocal_lambda = 1/lambda, in seconds.

Method 3 is used, where given a start time (Run-time Parameter), the subsequent send times are all computed prior to measurement by computing the pseudo-random distribution of inter-packet send times, (truncating the distribution as specified in the Run-time Parameters), and the Src sends each packet at the computed times.

Note that Trunc is the upper limit on inter-packet times in the Poisson distribution. A random value greater than Trunc is set equal to Trunc instead.

6.3.3. Traffic Filtering (observation) Details

The measured results based on a filtered version of the packets observed, and this section provides the filter details (when present).

NA

6.3.4. Sampling Distribution

NA

6.3.5. Run-time Parameters and Data Format

Run-time Parameters are input factors that must be determined, configured into the measurement system, and reported with the results for the context to be complete.

Src the IP address of the host in the Src Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the Dst Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a start time is unspecified and Tf is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Tf a time, the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of

[RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a end time date is ignored and Tf is interpreted as the Duration of the measurement interval.

Reciprocal_lambda average packet interval for Poisson Streams expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) with resolution of 0.0001 seconds (0.1 ms), and with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

Trunc Upper limit on Poisson distribution expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) with resolution of 0.0001 seconds (0.1 ms), and with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905] (values above this limit will be clipped and set to the limit value). (if fixed, Trunc = 30.0000 seconds.)

ID The 16-bit identifier assigned by the program that generates the query, and which must vary in successive queries, see Section 4.1.1 of [RFC1035]. This identifier is copied into the corresponding reply and can be used by the requester (Src) to match-up replies to outstanding queries.

QNAME The domain name of the Query, formatted as specified in section 4 of [RFC6991].

QTYPE The Query Type, which will correspond to the IP address family of the query (decimal 1 for IPv4 or 28 for IPv6, formatted as a uint16, as per section 9.2 of [RFC6020].

6.3.6. Roles

Src launches each packet and waits for return transmissions from Dst.

Dst waits for each packet from Src and sends a return packet to Src.

6.4. Output

This category specifies all details of the Output of measurements using the metric.

6.4.1. Type

Raw -- for each DNS Query packet sent, sets of values as defined in the next column, including the status of the response, only assigning delay values to successful query-response pairs.

6.4.2. Reference Definition

For all outputs:

T the time the DNS Query was sent during the measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

dT The time value of the round-trip delay to receive the DNS response, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]. This value is undefined when the response packet is not received at Src within waiting time Tmax seconds.

Rcode The value of the Rcode field in the DNS response header, expressed as a uint64 as specified in section 9.2 of [RFC6020]. Non-zero values convey errors in the response, and such replies must be analyzed separately from successful requests.

6.4.3. Metric Units

RTDNS: Round-trip Delay, dT, is expressed in seconds.

RTLDNS: the Logical value, where 1 = Lost and 0 = Received.

6.4.4. Calibration

Section 3.7.3 of [RFC7679] provides a means to quantify the systematic and random errors of a time measurement. In-situ calibration could be enabled with an internal loopback at the Source host that includes as much of the measurement system as possible, performs address and payload manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface contention. Some portion of the random and systematic error can be characterized this way.

When a measurement controller requests a calibration measurement, the loopback is applied and the result is output in the same format as a

normal measurement with additional indication that it is a calibration result.

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

6.5. Administrative items

6.5.1. Status

Current

6.5.2. Requestor

This RFC number

6.5.3. Revision

1.0

6.5.4. Revision Date

YYYY-MM-DD

6.6. Comments and Remarks

Additional (Informational) details for this entry

7. UDP Poisson One-way Delay and Loss Registry Entries

This section specifies five initial registry entries for the UDP Poisson One-way Delay, and one for UDP Poisson One-way Loss.

IANA Note: Registry "Name" below specifies a single registry entry, whose output format varies according to the <statistic> element of the name that specifies one form of statistical summary. There is an additional metric name for the Loss metric.

All column entries beside the ID, Name, Description, and Output Reference Method categories are the same, thus this section proposes six closely-related registry entries. As a result, IANA is also asked to assign corresponding URNs and URLs to each Named Metric.

7.1. Summary

This category includes multiple indexes to the registry entries, the element ID and metric name.

7.1.1. ID (Identifier)

IANA is asked to assign different numeric identifiers to each of the six Metrics.

7.1.2. Name

OWDelay_Active_IP-UDP-Poisson-
Payload250B_RFCXXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- 95Percentile
- Mean
- Min
- Max
- StdDev

OWLoss_Active_IP-UDP-Poisson-
Payload250B_RFCXXXXsecY_Percent_LossRatio

7.1.3. URI and URL

URI: Prefix urn:ietf:metrics:perf:<name>

URL: http:\\www.iana.org\ ... <name>

7.1.4. Description

OWDelay: This metric assesses the delay of a stream of packets exchanged between two hosts (or measurement points), and reports the <statistic> One-way delay for all successfully exchanged packets based on their conditional delay distribution.

where <statistic> is one of:

- 95Percentile
- Mean

- Min
- Max
- StdDev

OWLoss: This metric assesses the loss ratio of a stream of packets exchanged between two hosts (which are the two measurement points), and the Output is the One-way loss ratio for all successfully received packets expressed as a percentage.

7.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

7.2.1. Reference Definition

For Delay:

Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <http://www.rfc-editor.org/info/rfc7679>.

[RFC7679]

Morton, A., and Stephan, E., "Spatial Composition of Metrics", RFC 6049, January 2011.

[RFC6049]

Section 3.4 of [RFC7679] provides the reference definition of the singleton (single value) One-way delay metric. Section 4.4 of [RFC7679] provides the reference definition expanded to cover a multi-value sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

Only successful packet transfers with finite delay are included in the sample, as prescribed in section 4.1.2 of [RFC6049].

For loss:

Almes, G., Kalidini, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", RFC 7680, DOI 10.17487/RFC7680, January 2016, <http://www.rfc-editor.org/info/rfc7680>.

Section 2.4 of [RFC7680] provides the reference definition of the singleton (single value) one-way loss metric. Section 3.4 of [RFC7680] provides the reference definition expanded to cover a multi-singleton sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

7.2.2. Fixed Parameters

Type-P:

- IPv4 header values:
  - DSCP: set to 0
  - TTL: set to 255
  - Protocol: Set to 17 (UDP)
- IPv6 header values:
  - DSCP: set to 0
  - Hop Count: set to 255
  - Protocol: Set to 17 (UDP)
- UDP header values:
  - Checksum: the checksum MUST be calculated and included in the header
- UDP Payload: TWAMP Test Packet Formats, Section 4.1.2 of [RFC5357]
  - Security features in use influence the number of Padding octets.
  - 250 octets total, including the TWAMP format

Other measurement parameters:

Tmax: a loss threshold waiting time with value 3.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

See the Packet Stream generation category for two additional Fixed Parameters.

7.3. Method of Measurement

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

7.3.1. Reference Method

The methodology for this metric is defined as Type-P-One-way-Delay-Poisson-Stream in section 3.6 of [RFC7679] and section 4.6 of [RFC7679] using the Type-P and Tmax defined under Fixed Parameters.

The reference method distinguishes between long-delayed packets and lost packets by implementing a maximum waiting time for packet arrival. Tmax is the waiting time used as the threshold to declare a packet lost. Lost packets SHALL be designated as having undefined delay, and counted for the OWLoss metric.

The calculations on the one-way delay SHALL be performed on the conditional distribution, conditioned on successful packet arrival within Tmax. Also, when all packet delays are stored, the process which calculates the one-way delay value MAY enforce the Tmax threshold on stored values before calculations. See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The reference method requires some way to distinguish between different packets in a stream to establish correspondence between sending times and receiving times for each successfully-arriving packet. Sequence numbers or other send-order identification MUST be retained at the Src or included with each packet to disambiguate packet reordering if it occurs.

Since a standard measurement protocol is employed [RFC5357], then the measurement process will determine the sequence numbers or timestamps applied to test packets after the Fixed and Runtime parameters are passed to that process. The measurement protocol dictates the format of sequence numbers and time-stamps conveyed in the TWAMP-Test packet payload.

7.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

Section 11.1.3 of RFC 2681 [RFC2330] provides three methods to generate Poisson sampling intervals. The reciprocal of lambda is the average packet spacing, thus the Run-time Parameter is Reciprocal_lambda = 1/lambda, in seconds.

Method 3 SHALL be used, where given a start time (Run-time Parameter), the subsequent send times are all computed prior to measurement by computing the pseudo-random distribution of inter-packet send times, (truncating the distribution as specified in the Parameter Trunc), and the Src sends each packet at the computed times.

Note that Trunc is the upper limit on inter-packet times in the Poisson distribution. A random value greater than Trunc is set equal to Trunc instead.

Reciprocal_lambda average packet interval for Poisson Streams expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) with resolution of 0.0001 seconds (0.1 ms), and with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905]. Reciprocal_lambda = 1 packet per second.

Trunc Upper limit on Poisson distribution expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) with resolution of 0.0001 seconds (0.1 ms), and with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905] (values above this limit will be clipped and set to the limit value). Trunc = 30.0000 seconds.

7.3.3. Traffic Filtering (observation) Details

NA

7.3.4. Sampling Distribution

NA

7.3.5. Run-time Parameters and Data Format

Run-time Parameters are input factors that must be determined, configured into the measurement system, and reported with the results for the context to be complete.

Src the IP address of the host in the Src Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the Dst Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a start time is unspecified and Tf is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Tf a time, the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a end time date is ignored and Tf is interpreted as the Duration of the measurement interval.

7.3.6. Roles

<lists the names of the different roles from the measurement method>

Src launches each packet and waits for return transmissions from Dst. This is the TWAMP Session-Sender.

Dst waits for each packet from Src and sends a return packet to Src. This is the TWAMP Session-Reflector.

7.4. Output

This category specifies all details of the Output of measurements using the metric.

7.4.1. Type

See subsection titles below for Types.

7.4.2. Reference Definition

For all output types ---

T0 the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

Tf the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of

[RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

For LossRatio -- the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 4.1 of [RFC7680].

For each <statistic>, one of the following sub-sections apply:

7.4.2.1. Percentile95

The 95th percentile SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3 of [RFC3393] for details on the percentile statistic (where Round-trip delay should be substituted for "ipdv").

The percentile = 95, meaning that the reported delay, "95Percentile", is the smallest value of one-way delay for which the Empirical Distribution Function (EDF), F(95Percentile) >= 95% of the singleton one-way delay values in the conditional distribution. See section 11.3 of [RFC2330] for the definition of the percentile statistic using the EDF.

95Percentile The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

7.4.2.2. Mean

The mean SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.2.2 of [RFC6049] for details on calculating this statistic, and 4.2.3 of [RFC6049].

Mean The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

7.4.2.3. Min

The minimum SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for details on calculating this statistic, and 4.3.3 of [RFC6049].

Min The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

7.4.2.4. Max

The maximum SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for a closely related method for calculating this statistic, and 4.3.3 of [RFC6049]. The formula is as follows:

```
Max = (FiniteDelay [j])

      such that for some index, j, where 1 <= j <= N
      FiniteDelay[j] >= FiniteDelay[n] for all n
```

Max The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001

seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

7.4.2.5. Std_Dev

The Std_Dev SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for a closely related method for calculating this statistic, and 4.3.3 of [RFC6049]. The formula is the classic calculation for standard deviation of a population.

Std_Dev The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

7.4.3. Metric Units

The <statistic> of One-way Delay is expressed in seconds.

The One-way Loss Ratio is expressed as a percentage of lost packets to total packets sent.

7.4.4. Calibration

Section 3.7.3 of [RFC7679] provides a means to quantify the systematic and random errors of a time measurement. In-situ calibration could be enabled with an internal loopback that includes as much of the measurement system as possible, performs address manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface contention. Some portion of the random and systematic error can be characterized this way.

For one-way delay measurements, the error calibration must include an assessment of the internal clock synchronization with its external reference (this internal clock is supplying timestamps for measurement). In practice, the time offsets of clocks at both the source and destination are needed to estimate the systematic error due to imperfect clock synchronization (the time offsets are

smoothed, thus the random variation is not usually represented in the results).

time_offset The time value of the result is expressed in units of seconds, as a signed value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

When a measurement controller requests a calibration measurement, the loopback is applied and the result is output in the same format as a normal measurement with additional indication that it is a calibration result. In any measurement, the measurement function SHOULD report its current estimate of time offset as an indicator of the degree of synchronization.

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

7.5. Administrative items

7.5.1. Status

Current

7.5.2. Requestor

This REFC number

7.5.3. Revision

1.0

7.5.4. Revision Date

YYYY-MM-DD

7.6. Comments and Remarks

Additional (Informational) details for this entry

8. UDP Periodic One-way Delay and Loss Registry Entries

This section specifies five initial registry entries for the UDP Periodic One-way Delay, and one for UDP Periodic One-way Loss.

IANA Note: Registry "Name" below specifies a single registry entry, whose output format varies according to the <statistic> element of the name that specifies one form of statistical summary. There is an additional metric name for the Loss metric.

All column entries beside the ID, Name, Description, and Output Reference Method categories are the same, thus this section proposes six closely-related registry entries. As a result, IANA is also asked to assign corresponding URNs and URLs to each Named Metric.

8.1. Summary

This category includes multiple indexes to the registry entries, the element ID and metric name.

8.1.1. ID (Identifier)

IANA is asked to assign a different numeric identifiers to each of the six Metrics.

8.1.2. Name

OWDelay_Active_IP-UDP-Periodic20m-
Payload142B_RFCXXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- 95Percentile
- Mean
- Min
- Max
- StdDev

OWLoss_Active_IP-UDP-Periodic-
Payload142B_RFCXXXXsecY_Percent_LossRatio

8.1.3. URIs

URI: Prefix urn:ietf:metrics:perf:<name>

URL: http:\\www.iana.org\ ... <name>

8.1.4. Description

OWDelay: This metric assesses the delay of a stream of packets exchanged between two hosts (or measurement points), and reports the <statistic> One-way delay for all successfully exchanged packets based on their conditional delay distribution.

where <statistic> is one of:

- 95Percentile
- Mean
- Min
- Max
- StdDev

OWLoss: This metric assesses the loss ratio of a stream of packets exchanged between two hosts (which are the two measurement points), and the Output is the One-way loss ratio for all successfully received packets expressed as a percentage.

8.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

8.2.1. Reference Definition

For Delay:

Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <http://www.rfc-editor.org/info/rfc7679>.

[RFC7679]

Morton, A., and Stephan, E., "Spatial Composition of Metrics", RFC 6049, January 2011.

[RFC6049]

Section 3.4 of [RFC7679] provides the reference definition of the singleton (single value) One-way delay metric. Section 4.4 of [RFC7679] provides the reference definition expanded to cover a multi-value sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

Only successful packet transfers with finite delay are included in the sample, as prescribed in section 4.1.2 of [RFC6049].

For loss:

Almes, G., Kalidini, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", RFC 7680, DOI 10.17487/RFC7680, January 2016, <http://www.rfc-editor.org/info/rfc7680>.

Section 2.4 of [RFC7680] provides the reference definition of the singleton (single value) one-way loss metric. Section 3.4 of [RFC7680] provides the reference definition expanded to cover a multi-singleton sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

8.2.2. Fixed Parameters

Type-P:

- IPv4 header values:
  - DSCP: set to 0
  - TTL: set to 255
  - Protocol: Set to 17 (UDP)
- IPv6 header values:
  - DSCP: set to 0
  - Hop Count: set to 255
  - Protocol: Set to 17 (UDP)
- UDP header values:

* Checksum: the checksum MUST be calculated and included in the header

o UDP Payload: TWAMP Test Packet Formats, Section 4.1.2 of [RFC5357]

* Security features in use influence the number of Padding octets.

* 142 octets total, including the TWAMP format (if used)

Other measurement parameters:

Tmax: a loss threshold waiting time with value 3.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

See the Packet Stream generation category for two additional Fixed Parameters.

8.3. Method of Measurement

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

8.3.1. Reference Method

The methodology for this metric is defined as Type-P-One-way-Delay-Poisson-Stream in section 3.6 of [RFC7679] and section 4.6 of [RFC7679] using the Type-P and Tmax defined under Fixed Parameters. However, a Periodic stream is used, as defined in [RFC3432].

The reference method distinguishes between long-delayed packets and lost packets by implementing a maximum waiting time for packet arrival. Tmax is the waiting time used as the threshold to declare a packet lost. Lost packets SHALL be designated as having undefined delay, and counted for the OWLoss metric.

The calculations on the one-way delay SHALL be performed on the conditional distribution, conditioned on successful packet arrival within Tmax. Also, when all packet delays are stored, the process which calculates the one-way delay value MAY enforce the Tmax threshold on stored values before calculations. See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The reference method requires some way to distinguish between different packets in a stream to establish correspondence between sending times and receiving times for each successfully-arriving packet. Sequence numbers or other send-order identification MUST be retained at the Src or included with each packet to disambiguate packet reordering if it occurs.

Since a standard measurement protocol is employed [RFC5357], then the measurement process will determine the sequence numbers or timestamps applied to test packets after the Fixed and Runtime parameters are passed to that process. The measurement protocol dictates the format of sequence numbers and time-stamps conveyed in the TWAMP-Test packet payload.

8.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

Section 3 of [RFC3432] prescribes the method for generating Periodic streams using associated parameters.

incT the nominal duration of inter-packet interval, first bit to first bit, with value 0.0200 expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

dT the duration of the interval for allowed sample start times, with value 1.0000, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

T0 the actual start time of the periodic stream, determined from T0 and dT.

NOTE: an initiation process with a number of control exchanges resulting in unpredictable start times (within a time interval) may be sufficient to avoid synchronization of periodic streams, and therefore a valid replacement for selecting a start time at random from a fixed interval.

These stream parameters will be specified as Run-time parameters.

8.3.3. Traffic Filtering (observation) Details

NA

8.3.4. Sampling Distribution

NA

8.3.5. Run-time Parameters and Data Format

Run-time Parameters are input factors that must be determined, configured into the measurement system, and reported with the results for the context to be complete.

Src the IP address of the host in the Src Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the Dst Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a start time is unspecified and Tf is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Tf a time, the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a end time date is ignored and Tf is interpreted as the Duration of the measurement interval.

8.3.6. Roles

Src launches each packet and waits for return transmissions from Dst. This is the TWAMP Session-Sender.

Dst waits for each packet from Src and sends a return packet to Src. This is the TWAMP Session-Reflector.

8.4. Output

This category specifies all details of the Output of measurements using the metric.

8.4.1. Type

<insert name of the output type, raw or a selected summary statistic>

See subsection titles in Reference Definition for Latency Types.

8.4.2. Reference Definition

For all output types ---

T0 the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

Tf the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

For LossRatio -- the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 4.1 of [RFC7680].

For each <statistic>, one of the following sub-sections apply:

8.4.2.1. Percentile95

The 95th percentile SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3 of [RFC3393] for details on the percentile statistic (where Round-trip delay should be substituted for "ipdv").

The percentile = 95, meaning that the reported delay, "95Percentile", is the smallest value of one-way delay for which the Empirical Distribution Function (EDF), F(95Percentile) >= 95% of the singleton one-way delay values in the conditional distribution. See section 11.3 of [RFC2330] for the definition of the percentile statistic using the EDF.

95Percentile The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction

digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

8.4.2.2. Mean

The mean SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.2.2 of [RFC6049] for details on calculating this statistic, and 4.2.3 of [RFC6049].

Mean The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

8.4.2.3. Min

The minimum SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for details on calculating this statistic, and 4.3.3 of [RFC6049].

Min The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

8.4.2.4. Max

The maximum SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for a closely related method for calculating this statistic, and 4.3.3 of [RFC6049]. The formula is as follows:

```
Max = (FiniteDelay [j])

      such that for some index, j, where 1 <= j <= N
      FiniteDelay[j] >= FiniteDelay[n] for all n
```

Max The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

8.4.2.5. Std_Dev

The Std_Dev SHALL be calculated using the conditional distribution of all packets with a finite value of One-way delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for a closely related method for calculating this statistic, and 4.3.3 of [RFC6049]. The formula is the classic calculation for standard deviation of a population.

Std_Dev The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

8.4.3. Metric Units

The <statistic> of One-way Delay is expressed in seconds, where <statistic> is one of:

- 95Percentile
- Mean

- Min
- Max
- StdDev

The One-way Loss Ratio is expressed as a percentage of lost packets to total packets sent.

8.4.4. Calibration

Section 3.7.3 of [RFC7679] provides a means to quantify the systematic and random errors of a time measurement. In-situ calibration could be enabled with an internal loopback that includes as much of the measurement system as possible, performs address manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface contention. Some portion of the random and systematic error can be characterized this way.

For one-way delay measurements, the error calibration must include an assessment of the internal clock synchronization with its external reference (this internal clock is supplying timestamps for measurement). In practice, the time offsets of clocks at both the source and destination are needed to estimate the systematic error due to imperfect clock synchronization (the time offsets are smoothed, thus the random variation is not usually represented in the results).

time_offset The time value of the result is expressed in units of seconds, as a signed value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

When a measurement controller requests a calibration measurement, the loopback is applied and the result is output in the same format as a normal measurement with additional indication that it is a calibration result. In any measurement, the measurement function SHOULD report its current estimate of time offset as an indicator of the degree of synchronization.

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

8.5. Administrative items

8.5.1. Status

Current

8.5.2. Requestor

This RFC number

8.5.3. Revision

1.0

8.5.4. Revision Date

YYYY-MM-DD

8.6. Comments and Remarks

9. ICMP Round-trip Latency and Loss Registry Entries

This section specifies three initial registry entries for the ICMP Round-trip Latency, and another entry for ICMP Round-trip Loss Ratio.

This section specifies four Registry entries with many common columns.

All column entries beside the ID, Name, Description, and Output Reference Method categories are the same, thus this section proposes two closely-related registry entries. As a result, IANA is also asked to assign four corresponding URNs and URLs to each Named Metric.

9.1. Summary

This category includes multiple indexes to the registry entry: the element ID and metric name.

9.1.1. ID (Identifier)

IANA is asked to assign different numeric identifiers to each of the four Named Metrics.

9.1.2. Name

RTDelay_Active_IP-ICMP-SendOnRcv_RFCXXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- Mean
- Min
- Max

RTLoss_Active_IP-ICMP-SendOnRcv_RFCXXXXsecY_Percent_LossRatio

9.1.3. URIs

URN: Prefix urn:ietf:metrics:perf:<name>

URL: http://<TBD by IANA>/<name>

9.1.4. Description

RTDelay: This metric assesses the delay of a stream of ICMP packets exchanged between two hosts (which are the two measurement points), and the Output is the Round-trip delay for all successfully exchanged packets expressed as the <statistic> of their conditional delay distribution, where <statistic> is one of:

- Mean
- Min
- Max

RTLoss: This metric assesses the loss ratio of a stream of ICMP packets exchanged between two hosts (which are the two measurement points), and the Output is the Round-trip loss ratio for all successfully exchanged packets expressed as a percentage.

9.1.5. Change Controller

IETF

9.1.6. Version (of Registry Format)

1.0

9.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

9.2.1. Reference Definition

Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.

[RFC2681]

Section 2.4 of [RFC2681] provides the reference definition of the singleton (single value) Round-trip delay metric. Section 3.4 of [RFC2681] provides the reference definition expanded to cover a multi-singleton sample. Note that terms such as singleton and sample are defined in Section 11 of [RFC2330].

Note that although the [RFC2681] definition of "Round-trip-Delay between Src and Dst" is directionally ambiguous in the text, this metric tightens the definition further to recognize that the host in the "Src" role will send the first packet to "Dst", and ultimately receive the corresponding return packet from "Dst" (when neither are lost).

Finally, note that the variable "dT" is used in [RFC2681] to refer to the value of Round-trip delay in metric definitions and methods. The variable "dT" has been re-used in other IPPM literature to refer to different quantities, and cannot be used as a global variable name.

Morton, A., "Round-trip Packet Loss Metrics", RFC 6673, August 2012.

[RFC6673]

Both delay and loss metrics employ a maximum waiting time for received packets, so the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 6.1 of [RFC6673].

9.2.2. Fixed Parameters

Type-P as defined in Section 13 of [RFC2330]:

- IPv4 header values:
  - DSCP: set to 0

      * TTL: set to 255

      * Protocol: Set to 01 (ICMP)

   o IPv6 header values:

      * DSCP: set to 0

      * Hop Limit: set to 255

      * Protocol: Set to 01 (ICMP)

   o ICMP header values:

      * Type: 8 (Echo Request)

      * Code: 0

      * Checksum: the checksum MUST be calculated and included in the header

      * (Identifier and Sequence Number set at Run-Time)

   o ICMP Payload

      * total of 32 bytes of random info

   Other measurement parameters:

   o Tmax: a loss threshold waiting time

      * 3.0, expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 4 (see section 9.3 of [RFC6020]) and with resolution of 0.0001 seconds (0.1 ms), with lossless conversion to/from the 32-bit NTP timestamp as per section 6 of [RFC5905].

9.3. Method of Measurement

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

9.3.1. Reference Method

The methodology for this metric is defined as Type-P-Round-trip-Delay-Poisson-Stream in section 2.6 of RFC 2681 [RFC2681] and section

3.6 of RFC 2681 [RFC2681] using the Type-P and Tmax defined under Fixed Parameters.

The reference method distinguishes between long-delayed packets and lost packets by implementing a maximum waiting time for packet arrival. Tmax is the waiting time used as the threshold to declare a packet lost. Lost packets SHALL be designated as having undefined delay, and counted for the RTLoss metric.

The calculations on the delay (RTD) SHALL be performed on the conditional distribution, conditioned on successful packet arrival within Tmax. Also, when all packet delays are stored, the process which calculates the RTD value MAY enforce the Tmax threshold on stored values before calculations. See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

The reference method requires some way to distinguish between different packets in a stream to establish correspondence between sending times and receiving times for each successfully-arriving packet. Sequence numbers or other send-order identification MUST be retained at the Src or included with each packet to disambiguate packet reordering if it occurs.

The measurement process will determine the sequence numbers applied to test packets after the Fixed and Runtime parameters are passed to that process. The ICMP measurement process and protocol will dictate the format of sequence numbers and other identifiers.

Refer to Section 4.4 of [RFC6673] for expanded discussion of the instruction to "send a Type-P packet back to the Src as quickly as possible" in Section 2.6 of RFC 2681 [RFC2681]. Section 8 of [RFC6673] presents additional requirements which MUST be included in the method of measurement for this metric.

9.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

The ICMP metrics use a sending discipline called "SendOnRcv" or Send On Receive. This is a modification of Section 3 of [RFC3432], which prescribes the method for generating Periodic streams using associated parameters:

incT the nominal duration of inter-packet interval, first bit to first bit

dT the duration of the interval for allowed sample start times

T0 the actual start time of the periodic stream

The incT and T0 stream parameters will be specified as Run-time parameters, dT is not used in SendOnRcv.

A SendOnRcv sender behaves exactly like a Periodic stream generator while all reply packets arrive with RTD < incT, and the inter-packet interval will be constant.

If a reply packet arrives with RTD >= incT, then the inter-packet interval for the next sending time is nominally RTD.

If a reply packet fails to arrive within Tmax, then the inter-packet interval for the next sending time is nominally Tmax.

If an immediate send on reply arrival is desired, then set incT=0.

9.3.3. Traffic Filtering (observation) Details

The measured results based on a filtered version of the packets observed, and this section provides the filter details (when present).

NA

9.3.4. Sampling Distribution

NA

9.3.5. Run-time Parameters and Data Format

Run-time Parameters are input factors that must be determined, configured into the measurement system, and reported with the results for the context to be complete.

Src the IP address of the host in the Src Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the Dst Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a start time is unspecified and Tf is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Count The total count of ICMP Echo Requests to send, formatted as a uint16, as per section 9.2 of [RFC6020].

(see the Packet Stream Generation section for additional Run-time parameters)

9.3.6. Roles

Src launches each packet and waits for return transmissions from Dst.

Dst waits for each packet from Src and sends a return packet to Src.

9.4. Output

This category specifies all details of the Output of measurements using the metric.

9.4.1. Type

See subsection titles in Reference Definition for Latency Types.

LossRatio -- the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 6.1 of [RFC6673].

9.4.2. Reference Definition

For all output types ---

T0 the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

Tf the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

TotalCount the count of packets actually sent by the Src to Dst during the measurement interval.

For LossRatio -- the count of lost packets to total packets sent is the basis for the loss ratio calculation as per Section 4.1 of [RFC7680].

For each <statistic>, one of the following sub-sections apply:

9.4.2.1. Mean

The mean SHALL be calculated using the conditional distribution of all packets with a finite value of Round-trip delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.2.2 of [RFC6049] for details on calculating this statistic, and 4.2.3 of [RFC6049].

Mean The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

9.4.2.2. Min

The minimum SHALL be calculated using the conditional distribution of all packets with a finite value of Round-trip delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for details on calculating this statistic, and 4.3.3 of [RFC6049].

Min The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

9.4.2.3. Max

The maximum SHALL be calculated using the conditional distribution of all packets with a finite value of Round-trip delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for a closely related method for calculating this statistic, and 4.3.3 of [RFC6049]. The formula is as follows:

```
Max = (FiniteDelay [j])

      such that for some index, j, where 1 <= j <= N
      FiniteDelay[j] >= FiniteDelay[n] for all n
```

Max The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

9.4.3. Metric Units

The <statistic> of Round-trip Delay is expressed in seconds, where <statistic> is one of:

- Mean
- Min
- Max

The Round-trip Loss Ratio is expressed as a percentage of lost packets to total packets sent.

9.4.4. Calibration

Section 3.7.3 of [RFC7679] provides a means to quantify the systematic and random errors of a time measurement. In-situ calibration could be enabled with an internal loopback at the Source host that includes as much of the measurement system as possible, performs address manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface

contention. Some portion of the random and systematic error can be characterized this way.

When a measurement controller requests a calibration measurement, the loopback is applied and the result is output in the same format as a normal measurement with additional indication that it is a calibration result.

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

9.5. Administrative items

9.5.1. Status

Current

9.5.2. Requestor

This RFC number

9.5.3. Revision

1.0

9.5.4. Revision Date

YYYY-MM-DD

9.6. Comments and Remarks

None

10. TCP Round-Trip Delay and Loss Registry Entries

This section specifies three initial registry entries for the Passive assessment of TCP Round-Trip Delay (RTD) and another entry for TCP Round-trip Loss Count.

This section specifies four Registry entries with many common columns.

All column entries beside the ID, Name, Description, and Output Reference Method categories are the same, thus this section proposes four closely-related registry entries. As a result, IANA is also

asked to assign four corresponding URNs and URLs to each Named Metric.

10.1. Summary

This category includes multiple indexes to the registry entry: the element ID and metric name.

10.1.1. ID (Identifier)

IANA is asked to assign different numeric identifiers to each of the four Named Metrics.

10.1.2. Name

RTDelay_Passive_IP-TCP_RFCXXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- Mean
- Min
- Max

RTDelay_Passive_IP-TCP-HS_RFCXXXXsecY_Seconds_Singleton

Note that a mid-point observer only has the opportuinty to compose a single RTDelay on the TCP Hand Shake.

RTLoss_Passive_IP-TCP_RFCXXXXsecY_Packet_Count

10.1.3. URIs

URN: Prefix urn:ietf:metrics:perf:<name>

URL: http://<TBD by IANA>/<name>

10.1.4. Description

RTDelay: This metric assesses the round-trip delay of TCP packets constituting a single connection, exchanged between two hosts. We consider the measurement of round-trip delay based on a single Observation Point [RFC7011] somewhere in the network. The Output is the Round-trip delay for all successfully exchanged packets expressed as the <statistic> of their conditional delay distribution, where <statistic> is one of:

- Mean
- Min
- Max

RTLoss: This metric assesses the estimated loss count for TCP packets constituting a single connection, exchanged between two hosts. We consider the measurement of round-trip delay based on a single Observation Point [RFC7011] somewhere in the network. The Output is the estimated Loss Count for the measurement interval.

10.1.5. Change Controller

IETF

10.1.6. Version (of Registry Format)

1.0

10.2. Metric Definition

This category includes columns to prompt the entry of all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters.

10.2.1. Reference Definitions

Although there is no RFC that describes passive measurement of Round-Trip Delay, the parallel definition for Active measurement is:

Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.

[RFC2681]

This metric definition uses the terms singleton and sample as defined in Section 11 of [RFC2330]. (Section 2.4 of [RFC2681] provides the reference definition of the singleton (single value) Round-trip delay metric. Section 3.4 of [RFC2681] provides the reference definition expanded to cover a multi-singleton sample.)

With the Observation Point [RFC7011] (OP) typically located between the hosts participating in the TCP connection, the Round-trip Delay metric requires two individual measurements between the OP and each host, such that the Spatial Composition [RFC6049]of the measurements yields a Round-trip Delay singleton (we are extending the composition of one-way subpath delays to subpath round-trip delay).

Using the direction of TCP SYN transmission to anchor the nomenclature, host A sends the SYN and host B replies with SYN-ACK during connection establishment. The direction of SYN transfer is considered the Forward direction of transmission, from A through OP to B (Reverse is B through OP to A).

Traffic filters reduce the packet stream at the OP to a Qualified bidirectional flow packets.

In the definitions below, Corresponding Packets are transferred in different directions and convey a common value in a TCP header field that establishes correspondence (to the extent possible). Examples may be found in the TCP timestamp fields.

For a real number, RTD_fwd, >> the Round-trip Delay in the Forward direction from OP to host B at time T' is RTD_fwd << REQUIRES that OP observed a Qualified Packet to host B at wire-time T', that host B received that packet and sent a Corresponding Packet back to host A, and OP observed the Corresponding Packet at wire-time T' + RTD_fwd.

For a real number, RTD_rev, >> the Round-trip Delay in the Reverse direction from OP to host A at time T'' is RTD_rev << REQUIRES that OP observed a Qualified Packet to host A at wire-time T'', that host A received that packet and sent a Corresponding Packet back to host B, and that OP observed the Corresponding Packet at wire-time T'' + RTD_rev.

Ideally, the packet sent from host B to host A in both definitions above SHOULD be the same packet (or, when measuring RTD_rev first, the packet from host A to host B in both definitions should be the same).

The REQUIRED Composition Function for a singleton of Round-trip Delay at time T (where T is the earliest of T' and T'' above) is:

RTDelay = RTD_fwd + RTD_rev

Note that when OP is located at host A or host B, one of the terms composing RTDelay will be zero or negligible.

When the Qualified and Corresponding Packets are a TCP-SYN and a TCP-SYN-ACK, then RTD_fwd == RTD_HS_fwd.

When the Qualified and Corresponding Packets are a TCP-SYN-ACK and a TCP-ACK, then RTD_rev == RTD_HS_rev.

The REQUIRED Composition Function for a singleton of Round-trip Delay for the connection Hand Shake:

RTDelay_HS = RTD_HS_fwd + RTD_HS_rev

The definition of Round-trip Loss Count uses the nomenclature developed above, based on observation of the TCP header sequence numbers and storing the sequence number gaps observed. Packet Losses can be inferred from:

- Out-of-order segments: TCP segments are transmitted with monotonically increasing sequence numbers, but these segments may be received out of order. Section 3 of [RFC4737] describes the notion of "next expected" sequence numbers which can be adapted to TCP segments (for the purpose of detecting reordered packets). Observation of out-of-order segments indicates loss on the path prior to the OP, and creates a gap.

- Duplicate segments: Section 2 of [RFC5560] defines identical packets and is suitable for evaluation of TCP packets to detect duplication. Observation of duplicate segments *without a corresponding gap* indicates loss on the path following the OP (because they overlap part of the delivered sequence numbers already observed at OP).

Each observation of an out-of-order or duplicate infers a singleton of loss, but composition of Round-trip Loss Counts will be conducted over a measurement interval which is synonymous with a single TCP connection.

With the above observations in the Forward direction over a measurement interval, the count of out-of-order and duplicate segments is defined as RTL_fwd. Comparable observations in the Reverse direction are defined as RTL_rev.

For a measurement interval (corresponding to a single TCP connection), T0 to Tf, the REQUIRED Composition Function for a the two single-direction counts of inferred loss is:

RTLoss = RTL_fwd + RTL_rev

10.2.2. Fixed Parameters

<list and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed>

Traffic Filters:

- IPv4 header values:

      * DSCP: set to 0

      * Protocol: Set to 06 (TCP)

   o IPv6 header values:

      * DSCP: set to 0

      * Protocol: Set to 06 (TCP)

   o TCP header values:

      * Flags: ACK, SYN, FIN, @@@@@ others??

      * Timestamp Option (TSopt): Set

         + Kind: 8

         + Length: 10 bytes

   o

10.3. Method of Measurement

   This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous methods for implementations.

10.3.1. Reference Methods

   The foundation methodology for this metric is defined in Section 4 of [RFC7323] using the Timestamp Option with modifications that allow application at a mid-path Observation Point (OP) [RFC7011]. Further details and applicable heuristics were derived from [Strowes] and [Trammell-14].

   The Traffic Filter at the OP is configured to observe a single TCP connection. When the SYN, SYN-ACK, ACK handshake occurs, it offers the first opportunity to measure both RTD_fwd (on the SYN to SYN-ACK pair) and RTD_rev (on the SYN-ACK to ACK pair). Label this singleton of RTDelay as RTDelay_HS (composed using the forward and reverse measurement pair). RTDelay_HS SHALL be treated separately from other RTDelays on data-bearing packets and their ACKs. The RTDelay_HS value MAY be used as a sanity check on other Composed values of RTDelay.

   For payload bearing packets, the OP measures the time interval between observation of a packet with Sequence Number s, and the

corresponding ACK with same Sequence number. When the payload is transferred from host A to host B, the observed interval is RTD_fwd.

Because many data transfers are unidirectional (say, in the Forward direction from host A to host B), it is necessary to use pure ACK packets with Timestamp (TSval) and their Timestamp value echo to perform a RTD_rev measurement. The time interval between observation of the ACK from B to A, and the corresponding packet with Timestamp echo (TSecr) is the RTD_rev.

Delay Measurement Filtering Heuristics:

If Data payloads were transferred in both Forward and Reverse directions, then the Round-Trip Time Measurement Rule in Section 4.1 of [RFC7323] could be applied. This rule essentially excludes any measurement using a packet unless it makes progress in the transfer (advances the left edge of the send window, consistent with[Strowes]).

A different heuristic from [Trammell-14] is to exclude any RTD_rev that is larger than previously observed values. This would tend to exclude Reverse measurements taken when the Application has no data ready to send, because considerable time could be added to RTD_rev from this source of error.

Note that the above Heuristic assumes that host A is sending data. Host A expecting a download would mean that this heuristic should be applied to RTD_fwd.

The statistic calculations to summarize the delay (RTDelay) SHALL be performed on the conditional distribution, conditioned on successful Forward and Reverse measurements which follow the Heuristics.

Method for Inferring Loss:

The OP tracks sequence numbers and stores gaps for each direction of transmission, as well as the next-expected sequence number as in [Trammell-14] and [RFC4737]. Loss is inferred from Out-of-order segments and Duplicate segments.

Loss Measurement Filtering Heuristics:

[Trammell-14] adds a window of evaluation based on the RTDelay.

Distinguish Re-ordered from OOO due to loss, because sequence number gap is filled during the same RTDelay window. Segments detected as re-ordered according to [RFC4737] MUST reduce the Loss Count inferred from Out-of-order segments.

Spurious (unneeded) retransmissions (observed as duplicates) can also be reduced this way, as described in [Trammell-14].

Sources of Error:

The principal source of RTDelay error is the host processing time to return a packet that defines the termination of a time interval. The heuristics above intend to mitigate these errors by excluding measurements where host processing time is a significant part of RTD_fwd or RTD_rev.

A key source of RTLoss error is observation loss, described in section 3 of [Trammell-14].

10.3.2. Packet Stream Generation

This section gives the details of the packet traffic which is the basis for measurement. In IPPM metrics, this is called the Stream, and can easily be described by providing the list of stream parameters.

NA

10.3.3. Traffic Filtering (observation) Details

The measured results based on a filtered version of the packets observed, and this section provides the filter details (when present).

The Fixed Parameters above give a portion of the Traffic Filter. Other aspects will be supplied as Run-time Parameters (below).

10.3.4. Sampling Distribution

This metric requires a complete sample of all packets that qualify according to the Traffic Filter criteria.

10.3.5. Run-time Parameters and Data Format

Run-time Parameters are input factors that must be determined, configured into the measurement system, and reported with the results for the context to be complete.

Src the IP address of the host in the host A Role (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see Section 4 of [RFC6991])

Dst the IP address of the host in the host B (format ipv4-address-no-zone value for IPv4, or ipv6-address-no-zone value for IPv6, see section 4 of [RFC6991])

T0 a time, the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. When T0 is "all-zeros", a start time is unspecified and Td is to be interpreted as the Duration of the measurement interval. The start time is controlled through other means.

Td Optionally, the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]), or the duration (see T0). The UTC Time Zone is required by Section 6.1 of [RFC2330]. Alternatively, the end of the measurement interval MAY be controlled by the measured connection, where the second pair of FIN and ACK packets exchanged between host A and B effectively ends the interval.

TTL or Hop Limit Set at desired value.

10.3.6. Roles

host A launches the SYN packet to open the connection, and synonymous with an IP address.

host B replies with the SYN-ACK packet to open the connection, and synonymous with an IP address.

10.4. Output

This category specifies all details of the Output of measurements using the metric.

10.4.1. Type

See subsection titles in Reference Definition for RTDelay Types.

For RTLoss -- the count of lost packets.

10.4.2. Reference Definition

For all output types ---

T0 the start of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330].

Tf the end of a measurement interval, (format "date-and-time" as specified in Section 5.6 of [RFC3339], see also Section 3 of [RFC6991]). The UTC Time Zone is required by Section 6.1 of [RFC2330]. The end of the measurement interval MAY be controlled by the measured connection, where the second pair of FIN and ACK packets exchanged between host A and B effectively ends the interval.

... ...

For RTDelay_HS -- the Round trip delay of the Handshake.

For RTLoss -- the count of lost packets.

For each <statistic>, one of the following sub-sections apply:

10.4.2.1. Mean

The mean SHALL be calculated using the conditional distribution of all packets with a finite value of Round-trip delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.2.2 of [RFC6049] for details on calculating this statistic, and 4.2.3 of [RFC6049].

Mean The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

10.4.2.2. Min

The minimum SHALL be calculated using the conditional distribution of all packets with a finite value of Round-trip delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for details on calculating this statistic, and 4.3.3 of [RFC6049].

Min The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

10.4.2.3. Max

The maximum SHALL be calculated using the conditional distribution of all packets with a finite value of Round-trip delay (undefined delays are excluded), a single value as follows:

See section 4.1 of [RFC3393] for details on the conditional distribution to exclude undefined values of delay, and Section 5 of [RFC6703] for background on this analysis choice.

See section 4.3.2 of [RFC6049] for a closely related method for calculating this statistic, and 4.3.3 of [RFC6049]. The formula is as follows:

```
Max = (FiniteDelay [j])

      such that for some index, j, where 1 <= j <= N
      FiniteDelay[j] >= FiniteDelay[n] for all n
```

Max The time value of the result is expressed in units of seconds, as a positive value of type decimal64 with fraction digits = 9 (see section 9.3 of [RFC6020]) with resolution of 0.000000001 seconds (1.0 ns), and with lossless conversion to/from the 64-bit NTP timestamp as per section 6 of RFC [RFC5905]

10.4.3. Metric Units

The <statistic> of Round-trip Delay is expressed in seconds, where <statistic> is one of:

- Mean
- Min
- Max

The Round-trip Delay of the Hand Shake is expressed in seconds.

The Round-trip Loss Count is expressed as a number of packets.

10.4.4. Calibration

Passive measurements at an OP could be calibrated against an active measurement (with loss emulation) at host A or B, where the active measurement represents the ground-truth.

10.5. Administrative items

10.5.1. Status

Current

10.5.2. Requestor

This RFC

10.5.3. Revision

1.0

10.5.4. Revision Date

YYYY-MM-DD

10.6. Comments and Remarks

None.

11. Security Considerations

These registry entries represent no known implications for Internet Security. Each referenced Metric contains a Security Considerations section.

12. IANA Considerations

IANA is requested to populate The Performance Metric Registry defined in [I-D.ietf-ippm-metric-registry] with the values defined above.

See the IANA Considerations section of [I-D.ietf-ippm-metric-registry] for additional requests and considerations.

13. Acknowledgements

The authors thank Brian Trammell for suggesting the term "Run-time Parameters", which led to the distinction between run-time and fixed parameters implemented in this memo, for identifying the IPFIX metric

with Flow Key as an example, for suggesting the Passive TCP RTD metric and supporting references, and for many other productive suggestions. Thanks to Peter Koch, who provided several useful suggestions for disambiguating successive DNS Queries in the DNS Response time metric.

The authors also acknowledge the constructive reviews and helpful suggestions from Barbara Stark, Juergen Schoenwaelder, Tim Carey, Yaakov Stein, and participants in the LMAP working group. Thanks to Michelle Cotton for her early IANA review, and to Amanda Barber for answering questions related to the presentation of the registry and accessibility of the complete template via URL.

14. References

14.1. Normative References

[I-D.ietf-ippm-metric-registry]
Bagnulo, M., Claise, B., Eardley, P., and A. Morton, "Registry for Performance Metrics", Internet Draft (work in progress) draft-ietf-ippm-metric-registry, 2014.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

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

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <https://www.rfc-editor.org/info/rfc2330>.

[RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, DOI 10.17487/RFC2679, September 1999, <https://www.rfc-editor.org/info/rfc2679>.

[RFC2680] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Packet Loss Metric for IPPM", RFC 2680, DOI 10.17487/RFC2680, September 1999, <https://www.rfc-editor.org/info/rfc2680>.

[RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <https://www.rfc-editor.org/info/rfc2681>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <https://www.rfc-editor.org/info/rfc3393>.

[RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, DOI 10.17487/RFC3432, November 2002, <https://www.rfc-editor.org/info/rfc3432>.

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, DOI 10.17487/RFC4737, November 2006, <https://www.rfc-editor.org/info/rfc4737>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5560] Uijterwaal, H., "A One-Way Packet Duplication Metric", RFC 5560, DOI 10.17487/RFC5560, May 2009, <https://www.rfc-editor.org/info/rfc5560>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, DOI 10.17487/RFC6049, January 2011, <https://www.rfc-editor.org/info/rfc6049>.

[RFC6673] Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, DOI 10.17487/RFC6673, August 2012, <https://www.rfc-editor.org/info/rfc6673>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, Ed., "TCP Extensions for High Performance", RFC 7323, DOI 10.17487/RFC7323, September 2014, <https://www.rfc-editor.org/info/rfc7323>.

[RFC7679] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <https://www.rfc-editor.org/info/rfc7679>.

[RFC7680] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", STD 82, RFC 7680, DOI 10.17487/RFC7680, January 2016, <https://www.rfc-editor.org/info/rfc7680>.

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

14.2. Informative References

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

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <https://www.rfc-editor.org/info/rfc3611>.

[RFC4148] Stephan, E., "IP Performance Metrics (IPPM) Metrics Registry", BCP 108, RFC 4148, DOI 10.17487/RFC4148, August 2005, <https://www.rfc-editor.org/info/rfc4148>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <https://www.rfc-editor.org/info/rfc4566>.

[RFC5472] Zseby, T., Boschi, E., Brownlee, N., and B. Claise, "IP Flow Information Export (IPFIX) Applicability", RFC 5472, DOI 10.17487/RFC5472, March 2009, <https://www.rfc-editor.org/info/rfc5472>.

[RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, DOI 10.17487/RFC5477, March 2009, <https://www.rfc-editor.org/info/rfc5477>.

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <https://www.rfc-editor.org/info/rfc5481>.

[RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, DOI 10.17487/RFC6248, April 2011, <https://www.rfc-editor.org/info/rfc6248>.

[RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, DOI 10.17487/RFC6390, October 2011, <https://www.rfc-editor.org/info/rfc6390>.

[RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View", RFC 6703, DOI 10.17487/RFC6703, August 2012, <https://www.rfc-editor.org/info/rfc6703>.

[RFC6776] Clark, A. and Q. Wu, "Measurement Identity and Information Reporting Using a Source Description (SDES) Item and an RTCP Extended Report (XR) Block", RFC 6776, DOI 10.17487/RFC6776, October 2012, <https://www.rfc-editor.org/info/rfc6776>.

[RFC6792] Wu, Q., Ed., Hunt, G., and P. Arden, "Guidelines for Use of the RTP Monitoring Framework", RFC 6792, DOI 10.17487/RFC6792, November 2012, <https://www.rfc-editor.org/info/rfc6792>.

[RFC7003] Clark, A., Huang, R., and Q. Wu, Ed., "RTP Control Protocol (RTCP) Extended Report (XR) Block for Burst/Gap Discard Metric Reporting", RFC 7003, DOI 10.17487/RFC7003, September 2013, <https://www.rfc-editor.org/info/rfc7003>.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/info/rfc7594>.

[Strowes] Strowes, S., "Passively Measuring TCP Round Trip Times, Communications of the ACM, Vol. 56 No. 10, Pages 57-64", September 2013.

[Trammell-14] Trammell, B., "Inline Data Integrity Signals for Passive Measurement, TMA 2014 https://trammell.ch/pdf/qof-tma14.pdf", March 2014.

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/

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

Philip Eardley
BT
Adastral Park, Martlesham Heath
Ipswich
ENGLAND

Email: philip.eardley@bt.com

Kevin D'Souza
AT&T Labs
200 Laurel Avenue South
Middletown,, NJ 07748
USA

Phone: +1 732 420 xxxx
Email: kld@att.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: April 23, 2019

F. Brockners
S. Bhandari
C. Pignataro
Cisco
H. Gredler
RtBrick Inc.
J. Leddy
Comcast
S. Youell
JPMC
T. Mizrahi
Huawei Network.IO Innovation Lab
D. Mozes

P. Lapukhov
Facebook
R. Chang
Barefoot Networks
D. Bernier
Bell Canada
J. Lemon
Broadcom

October 20, 2018

Data Fields for In-situ OAM
draft-ietf-ippm-ioam-data-04

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in the packet while the packet traverses a path between two points in the network. This document discusses the data fields and associated data types for in-situ OAM. In-situ OAM data fields can be embedded into a variety of transports such as NSH, Segment Routing, Geneve, native IPv6 (via extension header), or IPv4. In-situ OAM can be used to complement OAM mechanisms based on e.g. ICMP or other types of probe packets.

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 https://datatracker.ietf.org/drafts/current/.

Table of Contents

1. Introduction

This document defines data fields for "in-situ" Operations, Administration, and Maintenance (IOAM). In-situ OAM records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the OAM data is added to the data packets rather than is being sent within packets specifically dedicated to OAM. IOAM is to complement mechanisms such as Ping or Traceroute, or more recent active probing mechanisms as described in [I-D.lapukhov-dataplane-probe]. In terms of "active" or "passive" OAM, "in-situ" OAM can be considered a hybrid OAM type. While no extra packets are sent, IOAM adds information to the packets therefore cannot be considered passive. In terms of the classification given in [RFC7799] IOAM could be portrayed as Hybrid Type 1. "In-situ" mechanisms do not require extra packets to be sent and hence don't change the packet traffic mix within the network. IOAM mechanisms can be leveraged where mechanisms using e.g. ICMP do not apply or do not offer the desired results, such as proving that a certain traffic flow takes a pre-defined path, SLA verification for the live data traffic, detailed statistics on traffic distribution paths in networks that distribute traffic across multiple paths, or scenarios in which probe traffic is potentially handled differently from regular data traffic by the network devices.

2. Conventions

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].

Abbreviations used in this document:

E2E Edge to Edge

Geneve: Generic Network Virtualization Encapsulation [I-D.ietf-nvo3-geneve]

IOAM: In-situ Operations, Administration, and Maintenance

MTU: Maximum Transmit Unit

NSH: Network Service Header [RFC8300]

OAM: Operations, Administration, and Maintenance

POT: Proof of Transit

SFC: Service Function Chain

SID: Segment Identifier

SR: Segment Routing

VXLAN-GPE: Virtual eXtensible Local Area Network, Generic Protocol Extension [I-D.ietf-nvo3-vxlan-gpe]

3. Scope, Applicability, and Assumptions

IOAM deployment assumes a set of constraints, requirements, and guiding principles which are described in this section.

Scope: This document defines the data fields and associated data types for in-situ OAM. The in-situ OAM data field can be transported by a variety of transport protocols, including NSH, Segment Routing, Geneve, IPv6, or IPv4. Specification details for these different transport protocols are outside the scope of this document.

Deployment domain (or scope) of in-situ OAM deployment: IOAM is a network domain focused feature, with "network domain" being a set of network devices or entities within a single administration. For example, a network domain can include an enterprise campus using physical connections between devices or an overlay network using virtual connections / tunnels for connectivity between said devices. A network domain is defined by its perimeter or edge. Designers of carrier protocols for IOAM must specify mechanisms to ensure that IOAM data stays within an IOAM domain. In addition, the operator of such a domain is expected to put provisions in place to ensure that IOAM data does not leak beyond the edge of an IOAM domain, e.g. using for example packet filtering methods. The operator should consider potential operational impact of IOAM to mechanisms such as ECMP processing (e.g. load-balancing schemes based on packet length could be impacted by the increased packet size due to IOAM), path MTU (i.e. ensure that the MTU of all links within a domain is sufficiently large to support the increased packet size due to IOAM) and ICMP message handling (i.e. in case of a native IPv6 transport, IOAM

support for ICMPv6 Echo Request/Reply could desired which would translate into ICMPv6 extensions to enable IOAM data fields to be copied from an Echo Request message to an Echo Reply message).

IOAM control points: IOAM data fields are added to or removed from the live user traffic by the devices which form the edge of a domain. Devices within an IOAM domain can update and/or add IOAM data-fields. Domain edge devices can be hosts or network devices.

Traffic-sets that IOAM is applied to: IOAM can be deployed on all or only on subsets of the live user traffic. It SHOULD be possible to enable IOAM on a selected set of traffic (e.g., per interface, based on an access control list or flow specification defining a specific set of traffic, etc.) The selected set of traffic can also be all traffic.

Encapsulation independence: Data formats for IOAM SHOULD be defined in a transport-independent manner. IOAM applies to a variety of encapsulating protocols. A definition of how IOAM data fields are carried by different transport protocols is outside the scope of this document.

Layering: If several encapsulation protocols (e.g., in case of tunneling) are stacked on top of each other, IOAM data-records could be present at every layer. The behavior follows the ships-in-the-night model, i.e. IOAM data in one layer is independent from IOAM data in another layer. Layering allows operators to instrument the protocol layer they want to measure. The different layers could, but do not have to share the same IOAM encapsulation and decapsulation.

Combination with active OAM mechanisms: IOAM should be usable for active network probing, enabling for example a customized version of traceroute. Decapsulating IOAM nodes may have an ability to send the IOAM information retrieved from the packet back to the source address of the packet or to the encapsulating node.

IOAM implementation: The IOAM data-field definitions take the specifics of devices with hardware data-plane and software data-plane into account.

4. IOAM Data Types and Formats

This section defines IOAM data types and data fields and associated data types required for IOAM.

To accommodate the different uses of IOAM, IOAM data fields fall into different categories, e.g. edge-to-edge, per node tracing, or for proof of transit. In IOAM these categories are referred to as IOAM-

Types. A common registry is maintained for IOAM-Types, see Section 7.2 for details. Corresponding to these IOAM-Types, different IOAM data fields are defined. IOAM data fields can be encapsulated into a variety of protocols, such as NSH, Geneve, IPv6, etc. The definition of how IOAM data fields are encapsulated into other protocols is outside the scope of this document.

IOAM is expected to be deployed in a specific domain rather than on the overall Internet. The part of the network which employs IOAM is referred to as the "IOAM-domain". IOAM data is added to a packet upon entering the IOAM-domain and is removed from the packet when exiting the domain. Within the IOAM-domain, the IOAM data may be updated by network nodes that the packet traverses. The device which adds an IOAM data container to the packet to capture IOAM data is called the "IOAM encapsulating node", whereas the device which removes the IOAM data container is referred to as the "IOAM decapsulating node". Nodes within the domain which are aware of IOAM data and read and/or write or process the IOAM data are called "IOAM transit nodes". IOAM nodes which add or remove the IOAM data container can also update the IOAM data fields at the same time. Or in other words, IOAM encapsulation or decapsulating nodes can also serve as IOAM transit nodes at the same time. Note that not every node in an IOAM domain needs to be an IOAM transit node. For example, a Segment Routing deployment might require the segment routing path to be verified. In that case, only the SR nodes would also be IOAM transit nodes rather than all nodes.

4.1. IOAM Namespaces

IOAM data fields are defined within an IOAM namespace. An IOAM namespace is identified by a 16-bit namespace identifier (Namespace-ID). Namespace identifiers MUST be present and populated in all IOAM option headers. The Namespace-ID value is divided into two sub-ranges:

- An operator-assigned range from 0x0001 to 0x7FFF
- An IANA-assigned range from 0x8000 to 0xFFFF

The IANA-assigned range is intended to allow future extensions to have new and interoperable IOAM functionality, while the operator-assigned range is intended to be domain specific, and managed by the network operator. The Namespace-ID value of 0x0000 is default and known to all the nodes implementing IOAM.

Namespace identifiers allow devices which are IOAM capable to determine:

- whether IOAM option header(s) need to be processed by a device: If the Namespace-ID contained in a packet does not match any Namespace-ID the node is configured to operate on, then the node MUST NOT change the contents of the IOAM data fields.

- which IOAM option headers need to be processed/updated in case there are multiple IOAM option headers present in the packet. Multiple option headers can be present in a packet in case of overlapping IOAM domains or in case of a layered IOAM deployment.

- whether IOAM option header(s) should be removed from the packet, e.g. at a domain edge or domain boundary.

IOAM namespaces support several different uses:

- Namespaces can be used by an operator to distinguish different operational domains. Devices at domain edges can filter on Namespace-IDs to provide for proper IOAM domain isolation.

- Namespaces provide additional context for IOAM data fields and thus ensure that IOAM data is unique. While, for example, the IOAM node identifier (Node-ID) does not have to be unique in a deployment, the combination of Node-ID and Namespace-ID will always be unique. Similarly, namespaces can be used to define how certain IOAM data fields are interpreted: IOAM offers three different timestamp format options. The Namespace-ID can be used to determine the timestamp format.

- Namespaces can be used to identify different sets of devices (e.g., different types of devices) in a deployment: If an operator desires to insert different IOAM data based on the device, the devices could be grouped into multiple namespaces. This could be due to the fact that the IOAM feature set differs between different sets of devices, or it could be for reasons of optimized space usage in the packet header. This could also stem from hardware or operational limitations on the size of the trace data that can be added and processed, preventing collection of a full trace for a flow.

  * Assigning different Namespace-IDs to different sets of nodes or network partitions and using the Namespace-ID as a selector at the IOAM encapsulating node, a full trace for a flow could be collected and constructed via partial traces in different packets of the same flow. Example: An operator could choose to group the devices of a domain into two namespaces, in a way that at average, only every second hop would be recorded by any device. To retrieve a full view of the deployment, the

captured IOAM data fields of the two namespaces need to be correlated.

* Assigning different Namespace-IDs to different sets of nodes or network partitions and using a separate IOAM header for each Namespace-ID, a full trace for a flow could be collected and constructed via partial traces from each IOAM header in each of the packets in the flow. Example: An operator could choose to group the devices of a domain into two namespaces, in a way that each namespace is represented by one of two IOAM headers in the packet. Each node would record data only for the IOAM namespace that it belongs to, ignoring the other IOAM header with a namespace to which it doesn't belong. To retrieve a full view of the deployment, the captured IOAM data fields of the two namespaces need to be correlated.

4.2. IOAM Tracing Options

"IOAM tracing data" is expected to be collected at every node that a packet traverses to ensure visibility into the entire path a packet takes within an IOAM domain, i.e., in a typical deployment all nodes in an in-situ OAM-domain would participate in IOAM and thus be IOAM transit nodes, IOAM encapsulating or IOAM decapsulating nodes. If not all nodes within a domain are IOAM capable, IOAM tracing information will only be collected on those nodes which are IOAM capable. Nodes which are not IOAM capable will forward the packet without any changes to the IOAM data fields. The maximum number of hops and the minimum path MTU of the IOAM domain is assumed to be known.

To optimize hardware and software implementations tracing is defined as two separate options. Any deployment MAY choose to configure and support one or both of the following options. An implementation of the transport protocol that carries these in-situ OAM data MAY choose to support only one of the options. In the event that both options are utilized at the same time, the Incremental Trace Option MUST be placed before the Pre-allocated Trace Option. Given that the operator knows which equipment is deployed in a particular IOAM, the operator will decide by means of configuration which type(s) of trace options will be enabled for a particular domain.

Pre-allocated Trace Option: This trace option is defined as a container of node data fields with pre-allocated space for each node to populate its information. This option is useful for software implementations where it is efficient to allocate the space once and index into the array to populate the data during transit. The IOAM encapsulating node allocates the option header and sets the fields in the option header. The in situ OAM

encapsulating node allocates an array which is used to store operational data retrieved from every node while the packet traverses the domain. IOAM transit nodes update the content of the array. A pointer which is part of the IOAM trace data points to the next empty slot in the array, which is where the next IOAM transit node fills in its data.

Incremental Trace Option: This trace option is defined as a container of node data fields where each node allocates and pushes its node data immediately following the option header. This type of trace recording is useful for some of the hardware implementations as this eliminates the need for the transit network elements to read the full array in the option and allows for arbitrarily long packets as the MTU allows. The in-situ OAM encapsulating node allocates the option header. The in-situ OAM encapsulating node based on operational state and configuration sets the fields in the header that control what node data fields should be collected, and how large the node data list can grow. The in-situ OAM transit nodes push their node data to the node data list, decrease the remaining length available to subsequent nodes, and adjust the lengths and possibly checksums in outer headers.

Every node data entry is to hold information for a particular IOAM transit node that is traversed by a packet. The in-situ OAM decapsulating node removes the IOAM data and processes and/or exports the metadata. IOAM data uses its own name-space for information such as node identifier or interface identifier. This allows for a domain-specific definition and interpretation. For example: In one case an interface-id could point to a physical interface (e.g., to understand which physical interface of an aggregated link is used when receiving or transmitting a packet) whereas in another case it could refer to a logical interface (e.g., in case of tunnels).

The following IOAM data is defined for IOAM tracing:

- Identification of the IOAM node. An IOAM node identifier can match to a device identifier or a particular control point or subsystem within a device.

- Identification of the interface that a packet was received on, i.e. ingress interface.

- Identification of the interface that a packet was sent out on, i.e. egress interface.

- Time of day when the packet was processed by the node. Different definitions of processing time are feasible and expected, though

it is important that all devices of an in-situ OAM domain follow the same definition.

- Generic data: Format-free information where syntax and semantic of the information is defined by the operator in a specific deployment. For a specific deployment, all IOAM nodes should interpret the generic data the same way. Examples for generic IOAM data include geo-location information (location of the node at the time the packet was processed), buffer queue fill level or cache fill level at the time the packet was processed, or even a battery charge level.

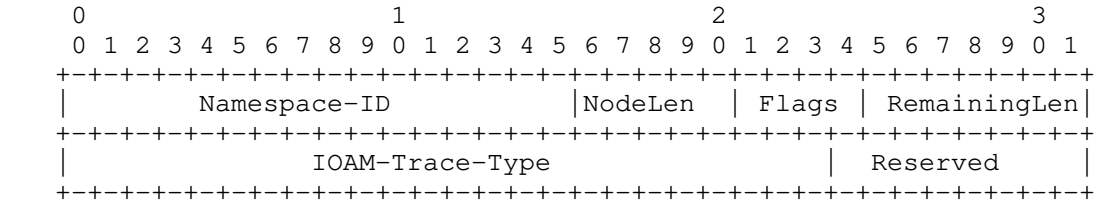
- A mechanism to detect whether IOAM trace data was added at every hop or whether certain hops in the domain weren't in-situ OAM transit nodes.

The "node data list" array in the packet is populated iteratively as the packet traverses the network, starting with the last entry of the array, i.e., "node data list [n]" is the first entry to be populated, "node data list [n-1]" is the second one, etc.

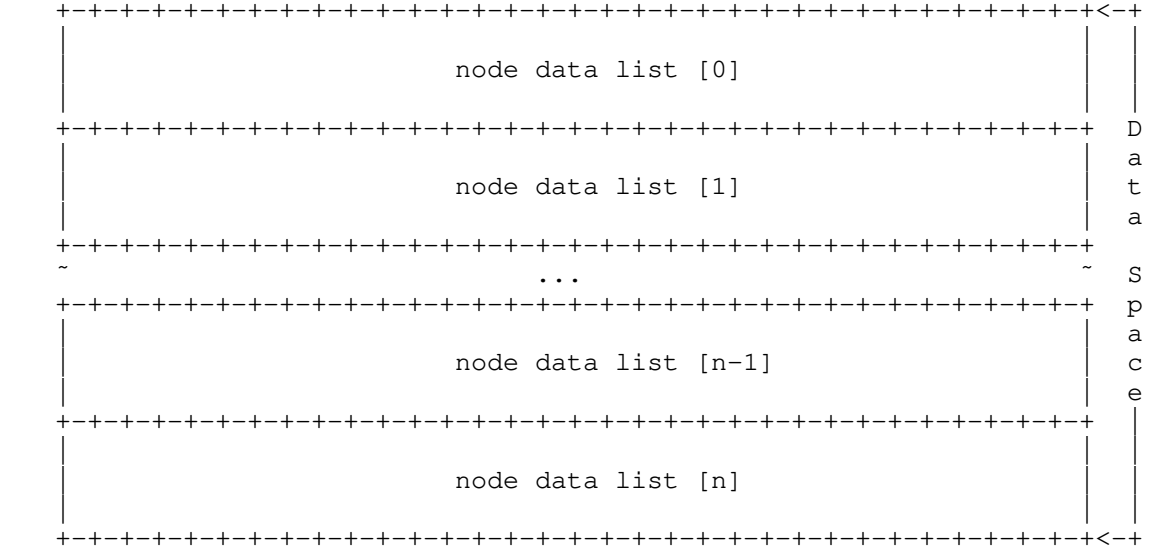
4.2.1. Pre-allocated and Incremental Trace Options

The in-situ OAM pre-allocated trace option and the in-situ OAM incremental trace option have similar formats. Except where noted below, the internal formats and fields of the two trace options are identical.

Pre-allocated and incremental trace option headers:



The trace option data MUST be 4-octet aligned:



Namespace-ID: 16-bit identifier of an IOAM namespace. The Namespace-ID value of 0x0000 is defined as the default value and MUST be known to all the nodes implementing IOAM. For any other Namespace-ID value that does not match any Namespace-ID the node is configured to operate on, the node MUST NOT change the contents of the IOAM data fields.

NodeLen: 5-bit unsigned integer. This field specifies the length of data added by each node in multiples of 4-octets, excluding the length of the "Opaque State Snapshot" field.

If IOAM-Trace-Type bit 7 is not set, then NodeLen specifies the actual length added by each node. If IOAM-Trace-Type bit 7 is

set, then the actual length added by a node would be (NodeLen + Opaque Data Length).

For example, if 3 IOAM-Trace-Type bits are set and none of them are wide, then NodeLen would be 3. If 3 IOAM-Trace-Type bits are set and 2 of them are wide, then NodeLen would be 5.

An IOAM encapsulating node must set NodeLen.

A node receiving an IOAM Pre-allocated or Incremental Trace Option may rely on the NodeLen value, or it may ignore the NodeLen value and calculate the node length from the IOAM-Trace-Type bits.

Flags 4-bit field. Following flags are defined:

Bit 0 "Overflow" (O-bit) (most significant bit). This bit is set by the network element if there is not enough number of octets left to record node data, no field is added and the overflow "O-bit" must be set to "1" in the header. This is useful for transit nodes to ignore further processing of the option.

Bit 1 "Loopback" (L-bit). Loopback mode is used to send a copy of a packet back towards the source. Loopback mode assumes that a return path from transit nodes and destination nodes towards the source exists. The encapsulating node decides (e.g. using a filter) which packets loopback mode is enabled for by setting the loopback bit. The encapsulating node also needs to ensure that sufficient space is available in the IOAM header for loopback operation. The loopback bit when set indicates to the transit nodes processing this option to create a copy of the packet received and send this copy of the packet back to the source of the packet while it continues to forward the original packet towards the destination. The source address of the original packet is used as destination address in the copied packet. The address of the node performing the copy operation is used as the source address. The L-bit MUST be cleared in the copy of the packet that a node sends back towards the source. On its way back towards the source, the packet is processed like a regular packet with IOAM information. Once the return packet reaches the IOAM domain boundary IOAM decapsulation occurs as with any other packet containing IOAM information.

Bit 2-3 Reserved: Must be zero.

RemainingLen: 7-bit unsigned integer. This field specifies the data space in multiples of 4-octets remaining for recording the node data, before the node data list is considered to have overflowed.

When RemainingLen reaches 0, nodes are no longer allowed to add node data. Given that the sender knows the minimum path MTU, the sender MAY set the initial value of RemainingLen according to the number of node data bytes allowed before exceeding the MTU. Subsequent nodes can carry out a simple comparison between RemainingLen and NodeLen, along with the length of the "Opaque State Snapshot" if applicable, to determine whether or not data can be added by this node. When node data is added, the node MUST decrease RemainingLen by the amount of data added. In the pre-allocated trace option, this is used as an offset in data space to record the node data element.

IOAM-Trace-Type: A 16-bit identifier which specifies which data types are used in this node data list.

The IOAM-Trace-Type value is a bit field. The following bit fields are defined in this document, with details on each field described in the Section 4.2.2. The order of packing the data fields in each node data element follows the bit order of the IOAM-Trace-Type field, as follows:

Bit 0 (Most significant bit) When set indicates presence of Hop_Lim and node_id in the node data.

Bit 1 When set indicates presence of ingress_if_id and egress_if_id (short format) in the node data.

Bit 2 When set indicates presence of timestamp seconds in the node data.

Bit 3 When set indicates presence of timestamp subseconds in the node data.

Bit 4 When set indicates presence of transit delay in the node data.

Bit 5 When set indicates presence of namespace specific data (short format) in the node data.

Bit 6 When set indicates presence of queue depth in the node data.

Bit 7 When set indicates presence of variable length Opaque State Snapshot field.

Bit 8 When set indicates presence of Hop_Lim and node_id in wide format in the node data.

Bit 9 When set indicates presence of ingress_if_id and egress_if_id in wide format in the node data.

Bit 10 When set indicates presence of namespace specific data in wide format in the node data.

Bit 11 When set indicates presence of buffer occupancy in the node data.

Bit 12-22 Undefined. An IOAM encapsulating node must set the value of each of these bits to 0. If an IOAM transit node receives a packet with one or more of these bits set to 1, it must either:

1. Add corresponding node data filled with the reserved value 0xFFFFFFFF, after the node data fields for the IOAM-Trace-Type bits defined above, such that the total node data added by this node in units of 4-octets is equal to NodeLen, or

2. Not add any node data fields to the packet, even for the IOAM-Trace-Type bits defined above.

Bit 23 When set indicates presence of the Checksum Complement node data.

Section 4.2.2 describes the IOAM data types and their formats. Within an in-situ OAM domain possible combinations of these bits making the IOAM-Trace-Type can be restricted by configuration knobs.

Reserved: 8-bits. Must be zero.

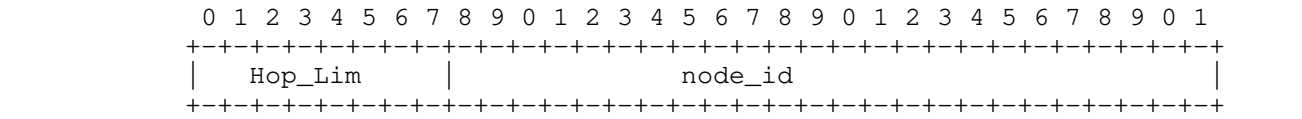
Node data List [n]: Variable-length field. The type of which is determined by the IOAM-Trace-Type bit representing the n-th node data in the node data list. The node data list is encoded starting from the last node data of the path. The first element of the node data list (node data list [0]) contains the last node of the path while the last node data of the node data list (node data list[n]) contains the first node data of the path traced. In the pre-allocated trace option, the index contained in RemainingLen identifies the offset for current active node data to be populated.

4.2.2. IOAM node data fields and associated formats

All the data fields MUST be 4-octet aligned. If a node which is supposed to update an IOAM data field is not capable of populating the value of a field set in the IOAM-Trace-Type, the field value MUST be set to 0xFFFFFFFF for 4-octet fields or 0xFFFFFFFFFFFFFFFF for 8-octet fields, indicating that the value is not populated, except when explicitly specified in the field description below.

Data field and associated data type for each of the data field is shown below:

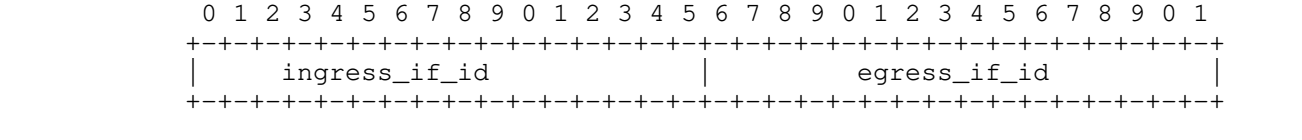
Hop_Lim and node_id: 4-octet field defined as follows:



Hop_Lim: 1-octet unsigned integer. It is set to the Hop Limit value in the packet at the node that records this data. Hop Limit information is used to identify the location of the node in the communication path. This is copied from the lower layer, e.g., TTL value in IPv4 header or hop limit field from IPv6 header of the packet when the packet is ready for transmission. The semantics of the Hop_Lim field depend on the lower layer protocol that IOAM is encapsulated over, and therefore its specific semantics are outside the scope of this memo.

node_id: 3-octet unsigned integer. Node identifier field to uniquely identify a node within in-situ OAM domain. The procedure to allocate, manage and map the node_ids is beyond the scope of this document.

ingress_if_id and egress_if_id: 4-octet field defined as follows:



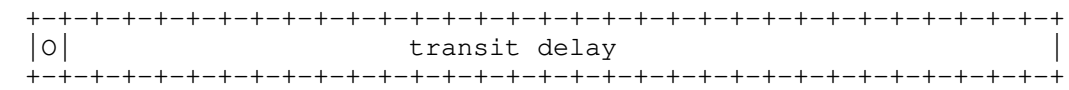
ingress_if_id: 2-octet unsigned integer. Interface identifier to record the ingress interface the packet was received on.

egress_if_id: 2-octet unsigned integer. Interface identifier to record the egress interface the packet is forwarded out of.

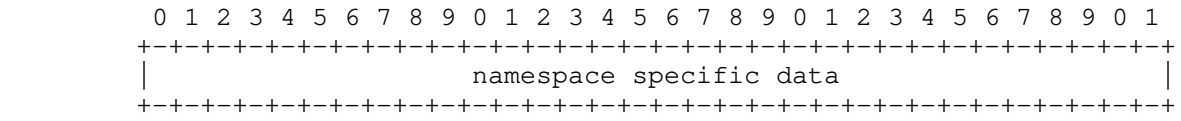
timestamp seconds: 4-octet unsigned integer. Absolute timestamp in seconds that specifies the time at which the packet was received by the node. This field has three possible formats; based on either PTP [IEEE1588v2], NTP [RFC5905], or POSIX [POSIX]. The three timestamp formats are specified in Section 5. In all three cases, the Timestamp Seconds field contains the 32 most significant bits of the timestamp format that is specified in Section 5. If a node is not capable of populating this field, it assigns the value 0xFFFFFFFF. Note that this is a legitimate value that is valid for 1 second in approximately 136 years; the analyzer should correlate several packets or compare the timestamp value to its own time-of-day in order to detect the error indication.

timestamp subseconds: 4-octet unsigned integer. Absolute timestamp in subseconds that specifies the time at which the packet was received by the node. This field has three possible formats; based on either PTP [IEEE1588v2], NTP [RFC5905], or POSIX [POSIX]. The three timestamp formats are specified in Section 5. In all three cases, the Timestamp Subseconds field contains the 32 least significant bits of the timestamp format that is specified in Section 5. If a node is not capable of populating this field, it assigns the value 0xFFFFFFFF. Note that this is a legitimate value in the NTP format, valid for approximately 233 picoseconds in every second. If the NTP format is used the analyzer should correlate several packets in order to detect the error indication.

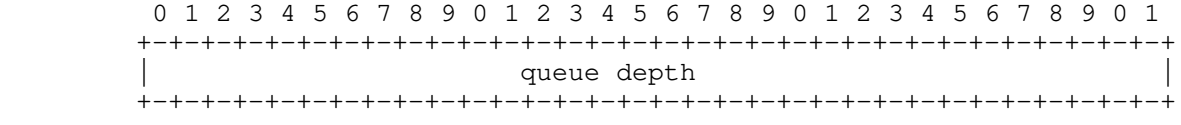
transit delay: 4-octet unsigned integer in the range 0 to 2^31-1. It is the time in nanoseconds the packet spent in the transit node. This can serve as an indication of the queuing delay at the node. If the transit delay exceeds 2^31-1 nanoseconds then the top bit 'O' is set to indicate overflow and value set to 0x80000000. When this field is part of the data field but a node populating the field is not able to fill it, the field position in the field must be filled with value 0xFFFFFFFF to mean not populated.



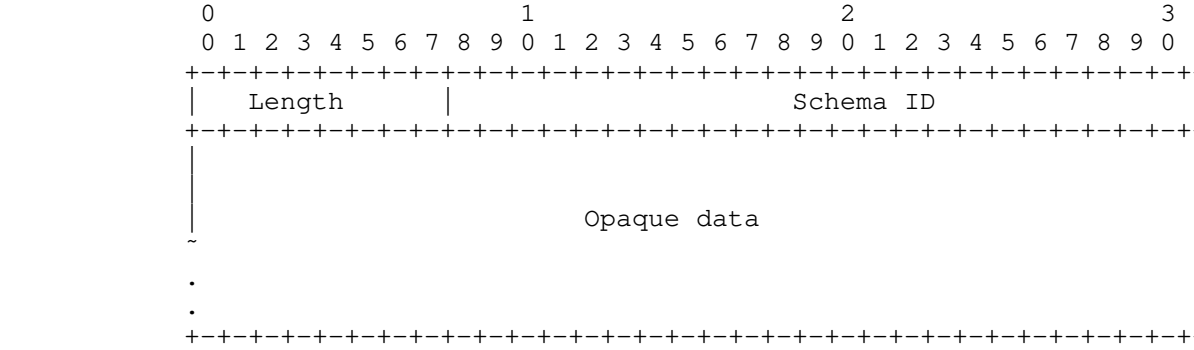
namespace specific data: 4-octet field which can be used by the node to add namespace specific data. This represents a "free-format" 4-octet bit field with its semantics defined in the context of a specific namespace.



queue depth: 4-octet unsigned integer field. This field indicates the current length of the egress interface queue of the interface from where the packet is forwarded out. The queue depth is expressed as the current number of memory buffers used by the queue (a packet may consume one or more memory buffers, depending on its size).



Opaque State Snapshot: Variable length field. It allows the network element to store an arbitrary state in the node data field, without a pre-defined schema. The schema is to be defined within the context of a namespace. The schema needs to be made known to the analyzer by some out-of-band mechanism. The specification of this mechanism is beyond the scope of this document. A 24-bit "Schema Id" field, interpreted within the context of a namespace, indicates which particular schema is used, and should be configured on the network element by the operator.



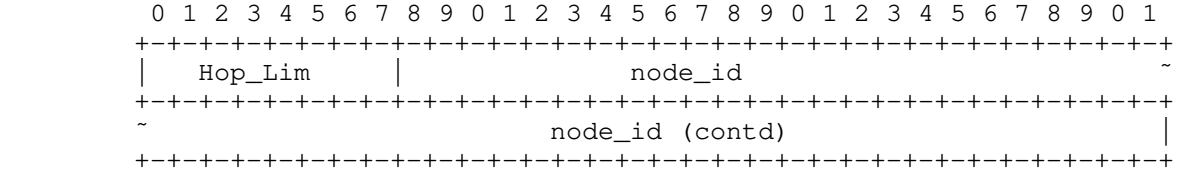
Length: 1-octet unsigned integer. It is the length in multiples of 4-octets of the Opaque data field that follows Schema Id.

Schema ID: 3-octet unsigned integer identifying the schema of Opaque data.

Opaque data: Variable length field. This field is interpreted as specified by the schema identified by the Schema ID.

When this field is part of the data field but a node populating the field has no opaque state data to report, the Length must be set to 0 and the Schema ID must be set to 0xFFFFFF to mean no schema.

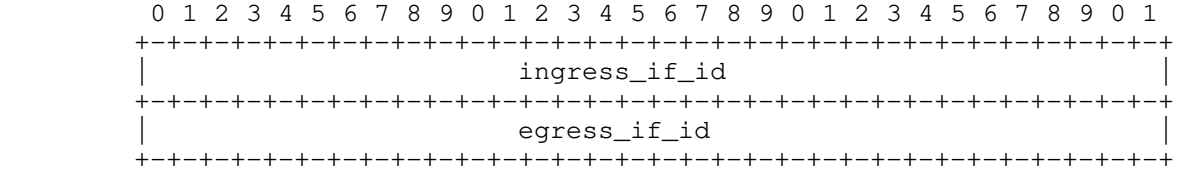
Hop_Lim and node_id wide: 8-octet field defined as follows:



Hop_Lim: 1-octet unsigned integer. It is set to the Hop Limit value in the packet at the node that records this data. Hop Limit information is used to identify the location of the node in the communication path. This is copied from the lower layer for e.g. TTL value in IPv4 header or hop limit field from IPv6 header of the packet. The semantics of the Hop_Lim field depend on the lower layer protocol that IOAM is encapsulated over, and therefore its specific semantics are outside the scope of this memo.

node_id: 7-octet unsigned integer. Node identifier field to uniquely identify a node within in-situ OAM domain. The procedure to allocate, manage and map the node_ids is beyond the scope of this document.

ingress_if_id and egress_if_id wide: 8-octet field defined as follows:

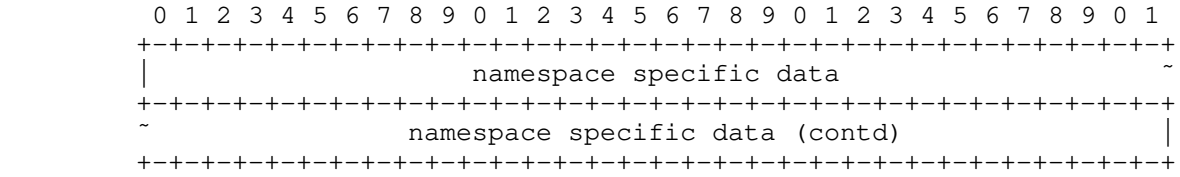


ingress_if_id: 4-octet unsigned integer. Interface identifier to record the ingress interface the packet was received on.

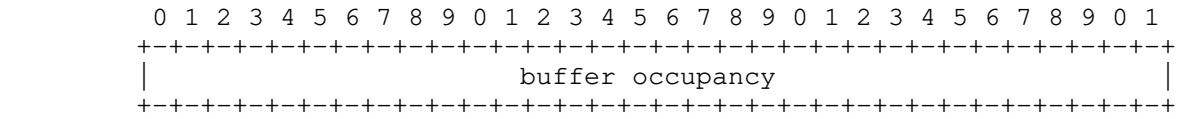
egress_if_id: 4-octet unsigned integer. Interface identifier to record the egress interface the packet is forwarded out of.

namespace specific data wide: 8-octet field which can be used by the node to add namespace specific data. This represents a "free-

format" 8-octet bit field with its semantics defined in the context of a specific namespace.



buffer occupancy: 4-octet unsigned integer field. This field indicates the current status of the buffer occupancy. The buffer occupancy is expressed as the current number of memory buffers used by the set of queues that share a common buffer pool.



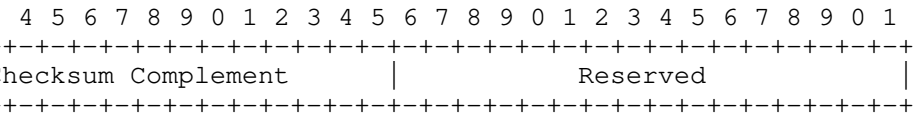
Checksum Complement: 4-octet node data which contains a two-octet Checksum Complement field, and a 2-octet reserved field. The Checksum Complement is useful when IOAM is transported over encapsulations that make use of a UDP transport, such as VXLAN-GPE or Geneve. Without the Checksum Complement, nodes adding IOAM node data must update the UDP Checksum field. When the Checksum Complement is present, an IOAM encapsulating node or IOAM transit node adding node data MUST carry out one of the following two alternatives in order to maintain the correctness of the UDP Checksum value:

1. Recompute the UDP Checksum field.

2. Use the Checksum Complement to make a checksum-neutral update in the UDP payload; the Checksum Complement is assigned a value that complements the rest of the node data fields that were added by the current node, causing the existing UDP Checksum field to remain correct.

IOAM decapsulating nodes MUST recompute the UDP Checksum field, since they do not know whether previous hops modified the UDP Checksum field or the Checksum Complement field.

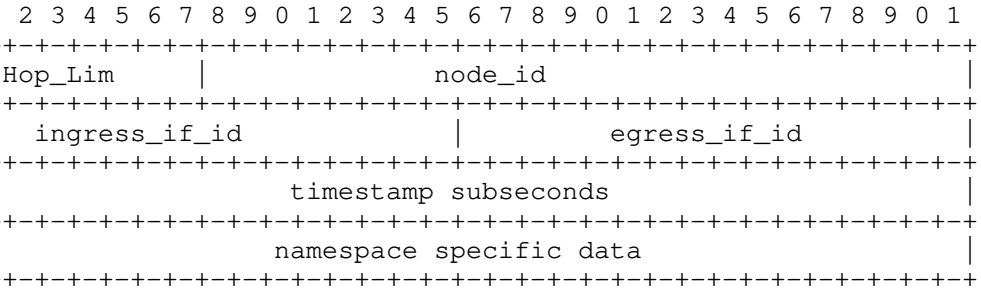
Checksum Complement fields are used in a similar manner in [RFC7820] and [RFC7821].



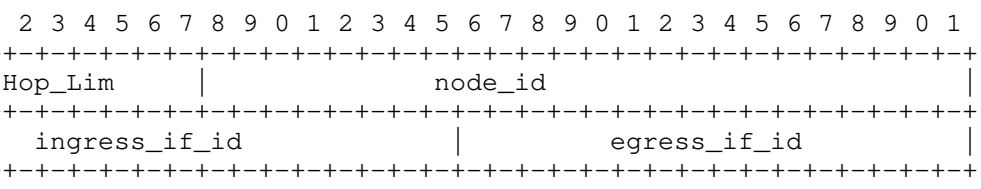
4.2.3. Examples of IOAM node data

An entry in the "node data list" array can have different formats, following the needs of the deployment. Some deployments might only be interested in recording the node identifiers, whereas others might be interested in recording node identifier and timestamp. The section defines different types that an entry in "node data list" can take.

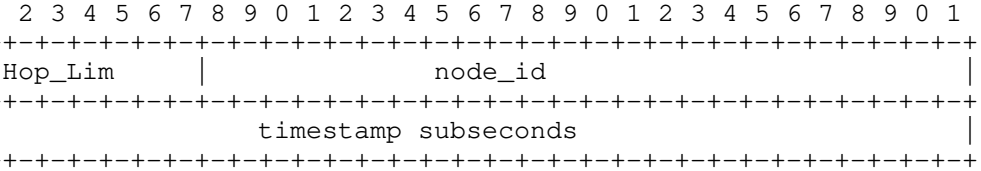
0xD400: IOAM-Trace-Type is 0xD400 then the format of node data is:



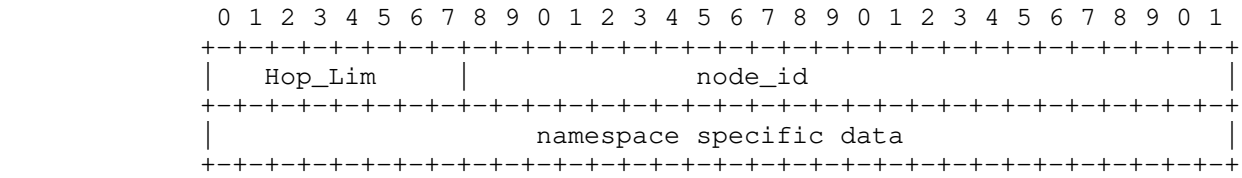
0xC000: IOAM-Trace-Type is 0xC000 then the format is:



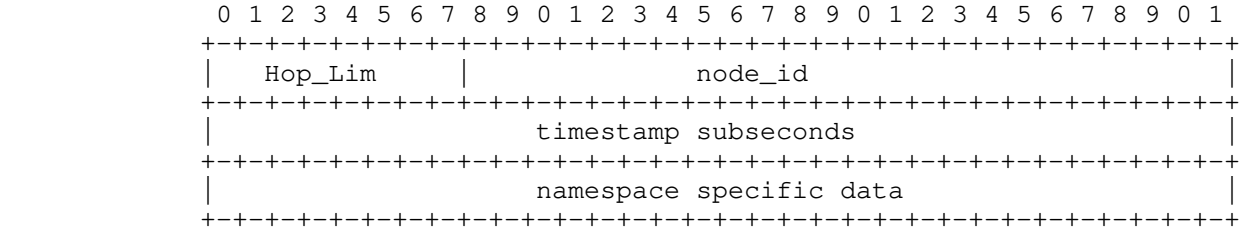
0x9000: IOAM-Trace-Type is 0x9000 then the format is:



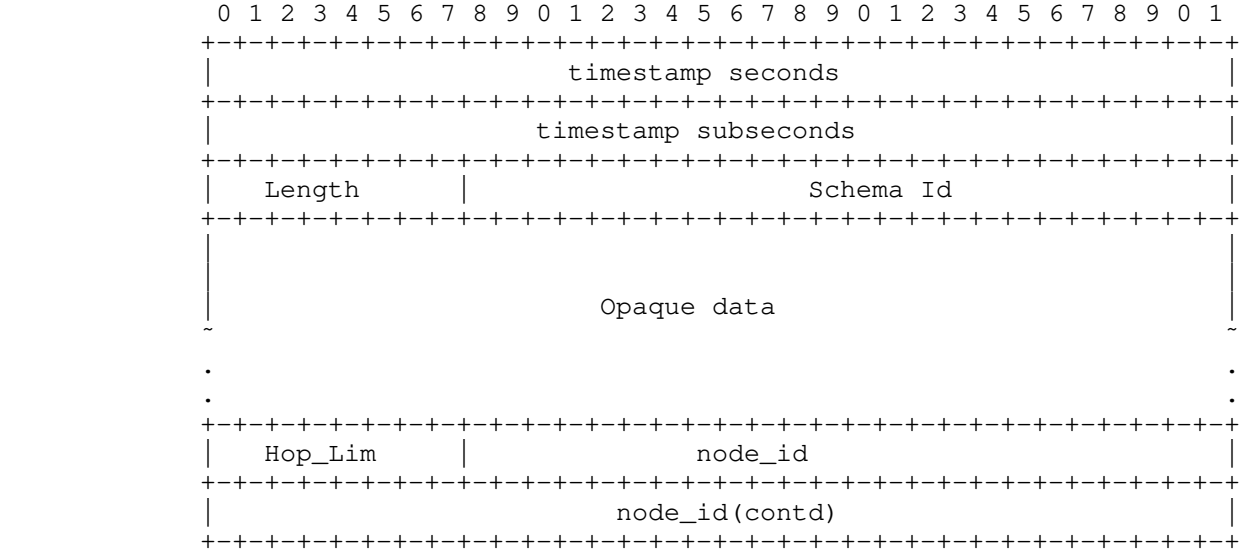
0x8400: IOAM-Trace-Type is 0x8400 then the format is:



0x9400: IOAM-Trace-Type is 0x9400 then the format is:



0x3180: IOAM-Trace-Type is 0x3180 then the format is:



4.3. IOAM Proof of Transit Option

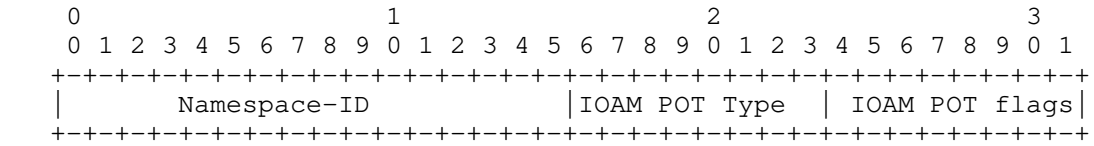
IOAM Proof of Transit data is to support the path or service function chain [RFC7665] verification use cases. Proof-of-transit uses methods like nested hashing or nested encryption of the IOAM data or mechanisms such as Shamir's Secret Sharing Schema (SSSS). While details on how the IOAM data for the proof of transit option is processed at IOAM encapsulating, decapsulating and transit nodes are outside the scope of the document, all of these approaches share the need to uniquely identify a packet as well as iteratively operate on a set of information that is handed from node to node. Correspondingly, two pieces of information are added as IOAM data to the packet:

- Random: Unique identifier for the packet (e.g., 64-bits allow for the unique identification of 2^64 packets).

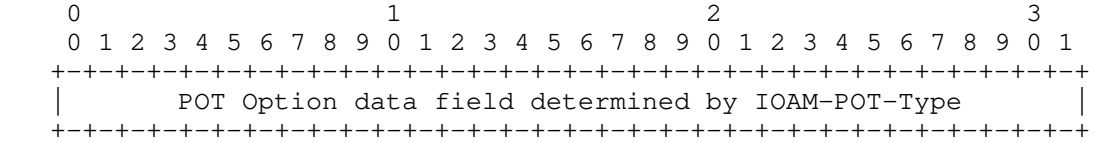
- Cumulative: Information which is handed from node to node and updated by every node according to a verification algorithm.

IOAM proof of transit option:

IOAM proof of transit option header:



IOAM proof of transit option data MUST be 4-octet aligned.:



Namespace-ID: 16-bit identifier of an IOAM namespace. The Namespace-ID value of 0x0000 is defined as the default value and MUST be known to all the nodes implementing IOAM. For any other Namespace-ID value that does not match any Namespace-ID the node is configured to operate on, the node MUST NOT change the contents of the IOAM data fields.

IOAM POT Type: 8-bit identifier of a particular POT variant that specifies the POT data that is included. This document defines POT Type 0:

0: POT data is a 16 Octet field as described below.

IOAM POT flags: 8-bit. Following flags are defined:

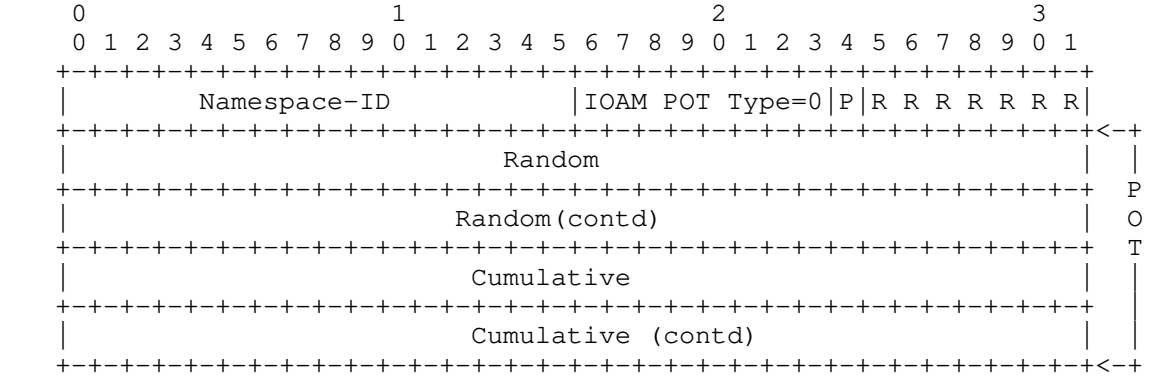
Bit 0 "Profile-to-use" (P-bit) (most significant bit). For IOAM POT types that use a maximum of two profiles to drive computation, indicates which POT-profile is used. The two profiles are numbered 0, 1.

Bit 1-7 Reserved: Must be set to zero upon transmission and ignored upon receipt.

POT Option data: Variable-length field. The type of which is determined by the IOAM-POT-Type.

4.3.1. IOAM Proof of Transit Type 0

IOAM proof of transit option of IOAM POT Type 0:



Namespace-ID: 16-bit identifier of an IOAM namespace. The Namespace-ID value of 0x0000 is defined as the default value and MUST be known to all the nodes implementing IOAM. For any other Namespace-ID value that does not match any Namespace-ID the node is configured to operate on, the node MUST NOT change the contents of the IOAM data fields.

IOAM POT Type: 8-bit identifier of a particular POT variant that specifies the POT data that is included. This section defines the POT data when the IOAM POT Type is set to the value 0.

P bit: 1-bit. "Profile-to-use" (P-bit) (most significant bit). Indicates which POT-profile is used to generate the Cumulative. Any node participating in POT will have a maximum of 2 profiles configured that drive the computation of cumulative. The two profiles are numbered 0, 1. This bit conveys whether profile 0 or profile 1 is used to compute the Cumulative.

R (7 bits): 7-bit IOAM POT flags for future use. MUST be set to zero upon transmission and ignored upon receipt.

Random: 64-bit Per packet Random number.

Cumulative: 64-bit Cumulative that is updated at specific nodes by processing per packet Random number field and configured parameters.

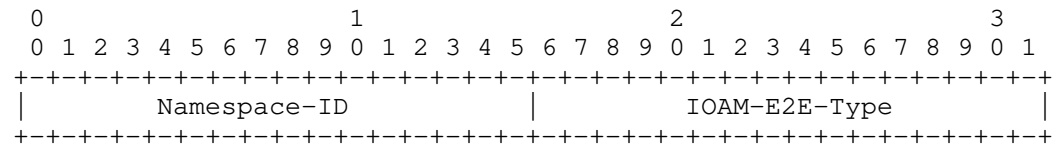
Note: Larger or smaller sizes of "Random" and "Cumulative" data are feasible and could be required for certain deployments (e.g. in case of space constraints in the transport protocol used). Future versions of this document will address different sizes of data for "proof of transit".

4.4. IOAM Edge-to-Edge Option

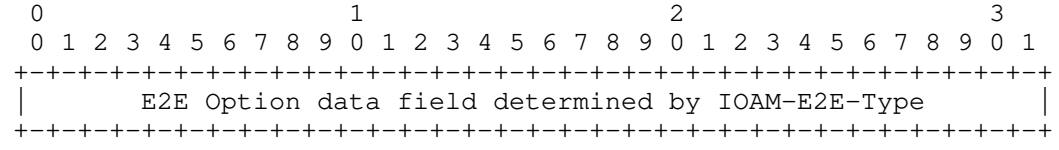
The IOAM edge-to-edge option is to carry data that is added by the IOAM encapsulating node and interpreted by IOAM decapsulating node. The IOAM transit nodes MAY process the data without modifying it.

IOAM edge-to-edge option:

IOAM edge-to-edge option header:



IOAM edge-to-edge option data MUST be 4-octet aligned:



Namespace-ID: 16-bit identifier of an IOAM namespace. The Namespace-ID value of 0x0000 is defined as the default value and MUST be known to all the nodes implementing IOAM. For any other Namespace-ID value that does not match any Namespace-ID the node is configured to operate on, then the node MUST NOT change the contents of the IOAM data fields.

IOAM-E2E-Type: A 16-bit identifier which specifies which data types are used in the E2E option data. The IOAM-E2E-Type value is a bit field. The order of packing the E2E option data field elements follows the bit order of the IOAM-E2E-Type field, as follows:

Bit 0 (Most significant bit) When set indicates presence of a 64-bit sequence number added to a specific tube which is used to detect packet loss, packet reordering, or packet duplication for that tube. Each tube leverages a dedicated namespace for its sequence numbers.

Bit 1 When set indicates presence of a 32-bit sequence number added to a specific tube which is used to detect packet loss, packet reordering, or packet duplication for that tube. Each tube leverages a dedicated namespace for its sequence numbers.

Bit 2 When set indicates presence of timestamp seconds for the transmission of the frame. This 4-octet field has three possible formats; based on either PTP [IEEE1588v2], NTP [RFC5905], or POSIX [POSIX]. The three timestamp formats are specified in Section 5. In all three cases, the

Timestamp Seconds field contains the 32 most significant bits of the timestamp format that is specified in Section 5. If a node is not capable of populating this field, it assigns the value 0xFFFFFFFF. Note that this is a legitimate value that is valid for 1 second in approximately 136 years; the analyzer should correlate several packets or compare the timestamp value to its own time-of-day in order to detect the error indication.

Bit 3 When set indicates presence of timestamp subseconds for the transmission of the frame. This 4-octet field has three possible formats; based on either PTP [IEEE1588v2], NTP [RFC5905], or POSIX [POSIX]. The three timestamp formats are specified in Section 5. In all three cases, the Timestamp Subseconds field contains the 32 least significant bits of the timestamp format that is specified in Section 5. If a node is not capable of populating this field, it assigns the value 0xFFFFFFFF. Note that this is a legitimate value in the NTP format, valid for approximately 233 picoseconds in every second. If the NTP format is used the analyzer should correlate several packets in order to detect the error indication.

Bit 4-15 Undefined. An IOAM encapsulating node Must set the value of these bits to zero upon transmission and ignore upon receipt.

E2E Option data: Variable-length field. The type of which is determined by the IOAM-E2E-Type.

5. Timestamp Formats

The IOAM data fields include a timestamp field which is represented in one of three possible timestamp formats. It is assumed that the management plane is responsible for determining which timestamp format is used.

5.1. PTP Truncated Timestamp Format

The Precision Time Protocol (PTP) [IEEE1588v2] uses an 80-bit timestamp format. The truncated timestamp format is a 64-bit field, which is the 64 least significant bits of the 80-bit PTP timestamp. The PTP truncated format is specified in Section 4.3 of [I-D.ietf-ntp-packet-timestamps], and the details are presented below for the sake of completeness.

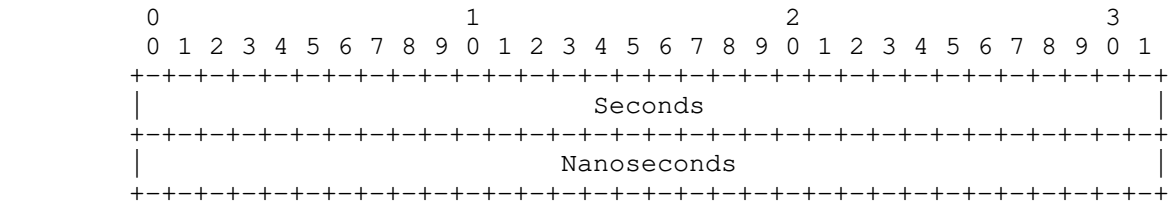


Figure 1: PTP [IEEE1588v2] Truncated Timestamp Format

Timestamp field format:

Seconds: specifies the integer portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: seconds.

Nanoseconds: specifies the fractional portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: nanoseconds. The value of this field is in the range 0 to (10^9)-1.

Epoch:

The PTP [IEEE1588v2] epoch is 1 January 1970 00:00:00 TAI, which is 31 December 1969 23:59:51.999918 UTC.

Resolution:

The resolution is 1 nanosecond.

Wraparound:

This time format wraps around every 2^32 seconds, which is roughly 136 years. The next wraparound will occur in the year 2106.

Synchronization Aspects:

It is assumed that nodes that run this protocol are synchronized among themselves. Nodes may be synchronized to a global reference time. Note that if PTP [IEEE1588v2] is used for synchronization, the timestamp may be derived from the PTP-synchronized clock,

allowing the timestamp to be measured with respect to the clock of an PTP Grandmaster clock.

The PTP truncated timestamp format is not affected by leap seconds.

5.2. NTP 64-bit Timestamp Format

The Network Time Protocol (NTP) [RFC5905] timestamp format is 64 bits long. This format is specified in Section 4.2.1 of [I-D.ietf-ntp-packet-timestamps], and the details are presented below for the sake of completeness.

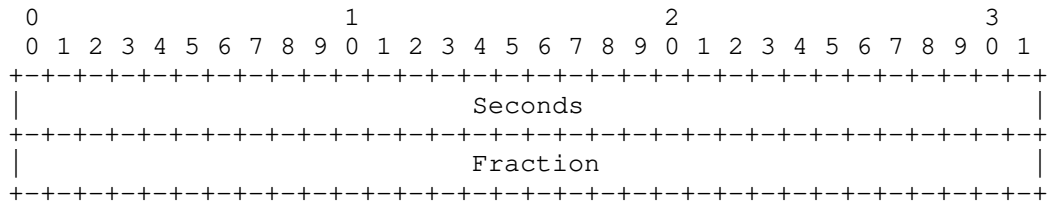


Figure 2: NTP [RFC5905] 64-bit Timestamp Format

Timestamp field format:

Seconds: specifies the integer portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: seconds.

Fraction: specifies the fractional portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: the unit is 2^(-32) seconds, which is roughly equal to 233 picoseconds.

Epoch:

The epoch is 1 January 1900 at 00:00 UTC.

Resolution:

The resolution is 2^(-32) seconds.

Wraparound:

This time format wraps around every 2^32 seconds, which is roughly 136 years. The next wraparound will occur in the year 2036.

Synchronization Aspects:

Nodes that use this timestamp format will typically be synchronized to UTC using NTP [RFC5905]. Thus, the timestamp may be derived from the NTP-synchronized clock, allowing the timestamp to be measured with respect to the clock of an NTP server.

The NTP timestamp format is affected by leap seconds; it represents the number of seconds since the epoch minus the number of leap seconds that have occurred since the epoch. The value of a timestamp during or slightly after a leap second may be temporarily inaccurate.

5.3. POSIX-based Timestamp Format

This timestamp format is based on the POSIX time format [POSIX]. The detailed specification of the timestamp format used in this document is presented below.

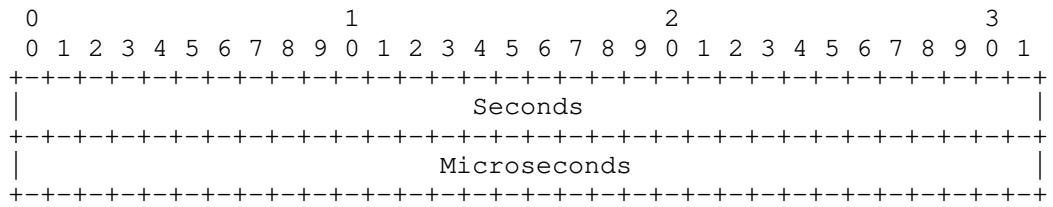


Figure 3: POSIX-based Timestamp Format

Timestamp field format:

Seconds: specifies the integer portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: seconds.

Microseconds: specifies the fractional portion of the number of seconds since the epoch.

+ Size: 32 bits.

+ Units: the unit is microseconds. The value of this field is in the range 0 to (10^6)-1.

Epoch:

The epoch is 1 January 1970 00:00:00 TAI, which is 31 December 1969 23:59:51.999918 UTC.

Resolution:

The resolution is 1 microsecond.

Wraparound:

This time format wraps around every 2^32 seconds, which is roughly 136 years. The next wraparound will occur in the year 2106.

Synchronization Aspects:

It is assumed that nodes that use this timestamp format run Linux operating system, and hence use the POSIX time. In some cases nodes may be synchronized to UTC using a synchronization mechanism that is outside the scope of this document, such as NTP [RFC5905]. Thus, the timestamp may be derived from the NTP-synchronized clock, allowing the timestamp to be measured with respect to the clock of an NTP server.

The POSIX-based timestamp format is affected by leap seconds; it represents the number of seconds since the epoch minus the number of leap seconds that have occurred since the epoch. The value of a timestamp during or slightly after a leap second may be temporarily inaccurate.

6. IOAM Data Export

IOAM nodes collect information for packets traversing a domain that supports IOAM. IOAM decapsulating nodes as well as IOAM transit nodes can choose to retrieve IOAM information from the packet, process the information further and export the information using e.g., IPFIX.

Raw data export of IOAM data using IPFIX is discussed in [I-D.spiegel-ippm-ioam-rawexport].

7. IANA Considerations

This document requests the following IANA Actions.

7.1. Creation of a new In-Situ OAM Protocol Parameters Registry (IOAM) Protocol Parameters IANA registry

IANA is requested to create a new protocol registry for "In-Situ OAM (IOAM) Protocol Parameters". This is the common registry that will include registrations for all IOAM namespaces. Each Registry, whose names are listed below:

IOAM Type

IOAM Trace Type

IOAM Trace flags

IOAM POT Type

IOAM POT flags

IOAM E2E Type

IOAM Namespace-ID

will contain the current set of possibilities defined in this document. New registries in this name space are created via RFC Required process as per [RFC8126].

The subsequent sub-sections detail the registries herein contained.

7.2. IOAM Type Registry

This registry defines 128 code points for the IOAM-Type field for identifying IOAM options as explained in Section 4. The following code points are defined in this draft:

0 IOAM Pre-allocated Trace Option Type

1 IOAM Incremental Trace Option Type

2 IOAM POT Option Type

3 IOAM E2E Option Type

4 - 127 are available for assignment via RFC Required process as per [RFC8126].

7.3. IOAM Trace Type Registry

This registry defines code point for each bit in the 16-bit IOAM-Trace-Type field for Pre-allocated trace option and Incremental trace option defined in Section 4.2. The meaning of Bits 0 - 11 for trace type are defined in this document in Paragraph 5 of Section 4.2.1:

Bit 0 hop_Lim and node_id in short format

Bit 1 ingress_if_id and egress_if_id in short format

Bit 2 timestamp seconds

Bit 3 timestamp subseconds

Bit 4 transit delay

Bit 5 namespace specific data in short format

Bit 6 queue depth

Bit 7 variable length Opaque State Snapshot

Bit 8 hop_Lim and node_id in wide format

Bit 9 ingress_if_id and egress_if_id in wide format

Bit 10 namespace specific data in wide format

Bit 11 buffer occupancy

Bit 23 checksum complement

The meaning for Bits 12 - 22 are available for assignment via RFC Required process as per [RFC8126].

7.4. IOAM Trace Flags Registry

This registry defines code points for each bit in the 4 bit flags for Pre-allocated trace option and Incremental trace option defined in Section 4.2. The meaning of Bit 0 - 1 for trace flags are defined in this document in Paragraph 3 of Section 4.2.1:

Bit 0 "Overflow" (O-bit)

Bit 1 "Loopback" (L-bit)

The meaning for Bits 2 - 3 are available for assignment via RFC Required process as per [RFC8126].

7.5. IOAM POT Type Registry

This registry defines 256 code points to define IOAM POT Type for IOAM proof of transit option Section 4.3. The code point value 0 is defined in this document:

0: 16 Octet POT data

1 - 255 are available for assignment via RFC Required process as per [RFC8126].

7.6. IOAM POT Flags Registry

This registry defines code points for each bit in the 8 bit flags for IOAM POT option defined in Section 4.3. The meaning of Bit 0 for IOAM POT flags is defined in this document in Section 4.3:

Bit 0 "Profile-to-use" (P-bit)

The meaning for Bits 1 - 7 are available for assignment via RFC Required process as per [RFC8126].

7.7. IOAM E2E Type Registry

This registry defines code points for each bit in the 16 bit IOAM-E2E-Type field for IOAM E2E option Section 4.4. The meaning of Bit 0 - 3 are defined in this document:

Bit 0 64-bit sequence number

Bit 1 32-bit sequence number

Bit 2 timestamp seconds

Bit 3 timestamp subseconds

The meaning of Bits 4 - 15 are available for assignment via RFC Required process as per [RFC8126].

7.8. IOAM Namespace-ID Registry

IANA is requested to set up an "IOAM Namespace-ID Registry", containing 16-bit values. The meaning of Bit 0 is defined in this document. IANA is requested to reserve the values 0x0001 to 0x7FFF for private use (managed by operators), as specified in Section 4.1

of the current document. Registry entries for the values 0x8000 to 0xFFFF are to be assigned via the "Expert Review" policy defined in [RFC8126].

0: default namespace (known to all IOAM nodes)

0x0001 - 0x7FFF: reserved for private use

0x8000 - 0xFFFF: unassigned

8. Security Considerations

As discussed in [RFC7276], a successful attack on an OAM protocol in general, and specifically on IOAM, can prevent the detection of failures or anomalies, or create a false illusion of nonexistent ones.

The Proof of Transit option (Section Section 4.3) is used for verifying the path of data packets. The security considerations of POT are further discussed in [I-D.brockners-proof-of-transit].

The data elements of IOAM can be used for network reconnaissance, allowing attackers to collect information about network paths, performance, queue states, buffer occupancy and other information.

IOAM can be used as a means for implementing Denial of Service (DoS) attacks, or for amplifying them. For example, a malicious attacker can add an IOAM header to packets in order to consume the resources of network devices that take part in IOAM or collectors that analyze the IOAM data. Another example is a packet length attack, in which an attacker pushes IOAM headers into data packets, causing these packets to be increased beyond the MTU size, resulting in fragmentation or in packet drops.

Since IOAM options may include timestamps, if network devices use synchronization protocols then any attack on the time protocol [RFC7384] can compromise the integrity of the timestamp-related data fields.

At the management plane, attacks may be implemented by misconfiguring or by maliciously configuring IOAM-enabled nodes in a way that enables other attacks. Thus, IOAM configuration should be secured in a way that authenticates authorized users and verifies the integrity of configuration procedures.

Notably, IOAM is expected to be deployed in specific network domains, thus confining the potential attack vectors to within the network domain. Indeed, in order to limit the scope of threats to within the

current network domain the network operator is expected to enforce policies that prevent IOAM traffic from leaking outside of the IOAM domain, and prevent IOAM data from outside the domain to be processed and used within the domain.

9. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, LJ Wobker, Erik Nordmark, Vengada Prasad Govindan, and Andrew Yourtchenko for the comments and advice.

This document leverages and builds on top of several concepts described in [I-D.kitamura-ipv6-record-route]. The authors would like to acknowledge the work done by the author Hiroshi Kitamura and people involved in writing it.

The authors would like to gracefully acknowledge useful review and insightful comments received from Joe Clarke, Al Morton, and Mickey Spiegel.

10. References

10.1. Normative References

[IEEE1588v2] Institute of Electrical and Electronics Engineers, "IEEE Std 1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008, <http://standards.ieee.org/findstds/standard/1588-2008.html>.

[POSIX] Institute of Electrical and Electronics Engineers, "IEEE Std 1003.1-2008 (Revision of IEEE Std 1003.1-2004) - IEEE Standard for Information Technology - Portable Operating System Interface (POSIX(R))", IEEE Std 1003.1-2008, 2008, <https://standards.ieee.org/findstds/standard/1003.1-2008.html>.

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

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

10.2. Informative References

[I-D.brockners-proof-of-transit]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Leddy, J., Youell, S., Mozes, D., and T. Mizrahi, "Proof of Transit", draft-brockners-proof-of-transit-05 (work in progress), May 2018.

[I-D.ietf-ntp-packet-timestamps]
Mizrahi, T., Fabini, J., and A. Morton, "Guidelines for Defining Packet Timestamps", draft-ietf-ntp-packet-timestamps-04 (work in progress), October 2018.

[I-D.ietf-nvo3-geneve]
Gross, J., Ganga, I., and T. Sridhar, "Geneve: Generic Network Virtualization Encapsulation", draft-ietf-nvo3-geneve-08 (work in progress), October 2018.

[I-D.ietf-nvo3-vxlan-gpe]
Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-06 (work in progress), April 2018.

[I-D.kitamura-ipv6-record-route]
Kitamura, H., "Record Route for IPv6 (PR6) Hop-by-Hop Option Extension", draft-kitamura-ipv6-record-route-00 (work in progress), November 2000.

[I-D.lapukhov-dataplane-probe]
Lapukhov, P. and r. remy@barefootnetworks.com, "Data-plane probe for in-band telemetry collection", draft-lapukhov-dataplane-probe-01 (work in progress), June 2016.

[I-D.spiegel-ippm-ioam-rawexport]
Spiegel, M., Brockners, F., Bhandari, S., and R. Sivakolundu, "In-situ OAM raw data export with IPFIX", draft-spiegel-ippm-ioam-rawexport-00 (work in progress), March 2018.

[RFC7276] Mizrahi, T., Sprecher, N., Bellagamba, E., and Y. Weingarten, "An Overview of Operations, Administration, and Maintenance (OAM) Tools", RFC 7276, DOI 10.17487/RFC7276, June 2014, <https://www.rfc-editor.org/info/rfc7276>.

[RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <https://www.rfc-editor.org/info/rfc7384>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <https://www.rfc-editor.org/info/rfc7799>.

[RFC7820] Mizrahi, T., "UDP Checksum Complement in the One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP)", RFC 7820, DOI 10.17487/RFC7820, March 2016, <https://www.rfc-editor.org/info/rfc7820>.

[RFC7821] Mizrahi, T., "UDP Checksum Complement in the Network Time Protocol (NTP)", RFC 7821, DOI 10.17487/RFC7821, March 2016, <https://www.rfc-editor.org/info/rfc7821>.

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

Authors' Addresses

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN 40549
Germany

Email: fbrockne@cisco.com

Shwetha Bhandari
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Email: shwethab@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com

Hannes Gredler
RtBrick Inc.

Email: hannes@rtbrick.com

John Leddy
Comcast
United States

Email: John_Leddy@cable.comcast.com

Stephen Youell
JP Morgan Chase
25 Bank Street
London E14 5JP
United Kingdom

Email: stephen.youell@jpmorgan.com

Tal Mizrahi
Huawei Network.IO Innovation Lab
Israel

Email: tal.mizrahi.phd@gmail.com

David Mozes

Email: mosesster@gmail.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com

Remy Chang
Barefoot Networks
4750 Patrick Henry Drive
Santa Clara, CA 95054
US

Daniel Bernier
Bell Canada
Canada

Email: daniel.bernier@bell.ca

John Lemon
Broadcom
270 Innovation Drive
San Jose, CA 95134
US

Email: john.lemon@broadcom.com

Network Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: June 10, 2019

M. Bagnulo
UC3M
B. Claise
Cisco Systems, Inc.
P. Eardley
BT
A. Morton
AT&T Labs
A. Akhter
Consultant

December 7, 2018

Registry for Performance Metrics
draft-ietf-ippm-metric-registry-17

Abstract

This document defines the format for the Performance Metrics registry and defines the IANA Registry for Performance Metrics. This document also gives a set of guidelines for Registered Performance Metric requesters and reviewers.

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 https://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 June 10, 2019.

Copyright Notice

Table of Contents

1. Introduction

The IETF specifies and uses Performance Metrics of protocols and applications transported over its protocols. Performance metrics are such an important part of the operations of IETF protocols that [RFC6390] specifies guidelines for their development.

The definition and use of Performance Metrics in the IETF happens in various working groups (WG), most notably:

The "IP Performance Metrics" (IPPM) WG is the WG primarily focusing on Performance Metrics definition at the IETF.

The "Metric Blocks for use with RTCP's Extended Report Framework" (XRBLOCK) WG recently specified many Performance Metrics related to "RTP Control Protocol Extended Reports (RTCP XR)" [RFC3611], which establishes a framework to allow new information to be conveyed in RTCP, supplementing the original report blocks defined in "RTP: A Transport Protocol for Real-Time Applications", [RFC3550].

The "Benchmarking Methodology" WG (BMWG) defined many Performance Metrics for use in laboratory benchmarking of inter-networking technologies.

The "IP Flow Information eXport" (IPFIX) concluded WG specified an IANA process for new Information Elements. Some Performance Metrics related Information Elements are proposed on regular basis.

The "Performance Metrics for Other Layers" (PMOL) concluded WG, defined some Performance Metrics related to Session Initiation Protocol (SIP) voice quality [RFC6035].

It is expected that more Performance Metrics will be defined in the future, not only IP-based metrics, but also metrics which are protocol-specific and application-specific.

However, despite the importance of Performance Metrics, there are two related problems for the industry. First, how to ensure that when one party requests another party to measure (or report or in some way act on) a particular Performance Metric, then both parties have exactly the same understanding of what Performance Metric is being referred to. Second, how to discover which Performance Metrics have been specified, so as to avoid developing new Performance Metric that is very similar, but not quite inter-operable. The problems can be addressed by creating a registry of performance metrics. The usual way in which IETF organizes namespaces is with Internet Assigned Numbers Authority (IANA) registries, and there is currently no Performance Metrics Registry maintained by the IANA.

This document therefore requests that IANA create and maintain a Performance Metrics Registry, according to the maintenance procedures and the Performance Metrics Registry format defined in this memo. Although the Registry format is primarily for use by IANA, any other organization that wishes to create a Performance Metrics Registry MAY use the same format for their purposes. The authors make no guarantee of the format's applicability to any possible set of Performance Metrics envisaged by other organizations, but encourage others to apply it. In the remainder of this document, unless we explicitly say so, we will refer to the IANA-maintained Performance Metrics Registry as simply the Performance Metrics Registry.

2. Terminology

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

Performance Metric: A Performance Metric is a quantitative measure of performance, targeted to an IETF-specified protocol or targeted to an application transported over an IETF-specified protocol. Examples of Performance Metrics are the FTP response time for a complete file download, the DNS response time to resolve the IP address, a database logging time, etc. This definition is

consistent with the definition of metric in [RFC2330] and broader than the definition of performance metric in [RFC6390].

Registered Performance Metric: A Registered Performance Metric is a Performance Metric expressed as an entry in the Performance Metric Registry, administered by IANA. Such a performance metric has met all the registry review criteria defined in this document in order to included in the registry.

Performance Metrics Registry: The IANA registry containing Registered Performance Metrics.

Proprietary Registry: A set of metrics that are registered in a proprietary registry, as opposed to Performance Metrics Registry.

Performance Metrics Experts: The Performance Metrics Experts is a group of designated experts [RFC8126] selected by the IESG to validate the Performance Metrics before updating the Performance Metrics Registry. The Performance Metrics Experts work closely with IANA.

Parameter: An input factor defined as a variable in the definition of a Performance Metric. A numerical or other specified factor forming one of a set that defines a metric or sets the conditions of its operation. All Parameters must be known to measure using a metric and interpret the results. There are two types of Parameters, Fixed and Run-time parameters. For the Fixed Parameters, the value of the variable is specified in the Performance Metrics Registry entry and different Fixed Parameter values results in different Registered Performance Metrics. For the Run-time Parameters, the value of the variable is defined when the metric measurement method is executed and a given Registered Performance Metric supports multiple values for the parameter. Although Run-time Parameters do not change the fundamental nature of the Performance Metric's definition, some have substantial influence on the network property being assessed and interpretation of the results.

Note: Consider the case of packet loss in the following two Active Measurement Method cases. The first case is packet loss as background loss where the Run-time Parameter set includes a very sparse Poisson stream, and only characterizes the times when packets were lost. Actual user streams likely see much higher loss at these times, due to tail drop or radio errors. The second case is packet loss as inverse of throughput where the Run-time Parameter set includes a very dense, bursty stream, and characterizes the loss experienced by a stream that approximates a user stream. These are both "loss metrics", but

the difference in interpretation of the results is highly dependent on the Run-time Parameters (at least), to the extreme where we are actually using loss to infer its compliment: delivered throughput.

Active Measurement Method: Methods of Measurement conducted on traffic which serves only the purpose of measurement and is generated for that reason alone, and whose traffic characteristics are known a priori. The complete definition of Active Methods is specified in section 3.4 of[RFC7799]. Examples of Active Measurement Methods are the measurement methods for the One way delay metric defined in [RFC7679] and the one for round trip delay defined in [RFC2681].

Passive Measurement Method: Methods of Measurement conducted on network traffic, generated either from the end users or from network elements that would exist regardless whether the measurement was being conducted or not. The complete definition of Passive Methods is specified in section 3.6 of [RFC7799]. One characteristic of Passive Measurement Methods is that sensitive information may be observed, and as a consequence, stored in the measurement system.

Hybrid Measurement Method: Hybrid Methods are Methods of Measurement that use a combination of Active Methods and Passive Methods, to assess Active Metrics, Passive Metrics, or new metrics derived from the a priori knowledge and observations of the stream of interest. The complete definition of Hybrid Methods is specified in section 3.8 of [RFC7799].

3. Scope

This document is meant mainly for two different audiences. For those defining new Registered Performance Metrics, it provides specifications and best practices to be used in deciding which Registered Performance Metrics are useful for a measurement study, instructions for writing the text for each column of the Registered Performance Metrics, and information on the supporting documentation required for the new Performance Metrics Registry entry (up to and including the publication of one or more RFCs or I-Ds describing it). For the appointed Performance Metrics Experts and for IANA personnel administering the new IANA Performance Metric Registry, it defines a set of acceptance criteria against which these proposed Registered Performance Metrics should be evaluated. In addition, this document may be useful for other organization who are defining a Performance Metric registry of its own, who can rely on the Performance Metric registry defined in this document.

This Performance Metric Registry is applicable to Performance Metrics issued from Active Measurement, Passive Measurement, and any other form of Performance Metric. This registry is designed to encompass Performance Metrics developed throughout the IETF and especially for the technologies specified in the following working groups: IPPM, XRBLOCK, IPFIX, and BMWG. This document analyzes an prior attempt to set up a Performance Metric Registry, and the reasons why this design was inadequate [RFC6248]. Finally, this document gives a set of guidelines for requesters and expert reviewers of candidate Registered Performance Metrics.

This document makes no attempt to populate the Performance Metrics Registry with initial entries. It does provides a few examples that are merely illustrations and should not be included in the registry at this point in time.

Based on [RFC8126] Section 4.3, this document is processed as Best Current Practice (BCP) [RFC2026].

4. Motivation for a Performance Metrics Registry

In this section, we detail several motivations for the Performance Metric Registry.

4.1. Interoperability

As any IETF registry, the primary use for a registry is to manage a namespace for its use within one or more protocols. In the particular case of the Performance Metric Registry, there are two types of protocols that will use the Performance Metrics in the Performance Metrics Registry during their operation (by referring to the Index values):

- Control protocol: this type of protocols is used to allow one entity to request another entity to perform a measurement using a specific metric defined by the Performance Metrics Registry. One particular example is the LMAP framework [RFC7594]. Using the LMAP terminology, the Performance Metrics Registry is used in the LMAP Control protocol to allow a Controller to request a measurement task to one or more Measurement Agents. In order to enable this use case, the entries of the Performance Metric Registry must be well enough defined to allow a Measurement Agent implementation to trigger a specific measurement task upon the reception of a control protocol message. This requirement heavily constrains the type of entries that are acceptable for the Performance Metric Registry.

- Report protocol: This type of protocols is used to allow an entity to report measurement results to another entity. By referencing to a specific Performance Metric Registry, it is possible to properly characterize the measurement result data being reported. Using the LMAP terminology, the Performance Metrics Registry is used in the Report protocol to allow a Measurement Agent to report measurement results to a Collector.

It should be noted that the LMAP framework explicitly allows for using not only the IANA-maintained Performance Metrics Registry but also other registries containing Performance Metrics, either defined by other organizations or private ones. However, others who are creating Registries to be used in the context of an LMAP framework are encouraged to use the Registry format defined in this document, because this makes it easier for developers of LMAP Measurement Agents (MAs) to programmatically use information found in those other Registries' entries.

4.2. Single point of reference for Performance Metrics

A Performance Metrics Registry serves as a single point of reference for Performance Metrics defined in different working groups in the IETF. As we mentioned earlier, there are several WGs that define Performance Metrics in the IETF and it is hard to keep track of all them. This results in multiple definitions of similar Performance Metrics that attempt to measure the same phenomena but in slightly different (and incompatible) ways. Having a registry would allow both the IETF community and external people to have a single list of relevant Performance Metrics defined by the IETF (and others, where appropriate). The single list is also an essential aspect of communication about Performance Metrics, where different entities that request measurements, execute measurements, and report the results can benefit from a common understanding of the referenced Performance Metric.

4.3. Side benefits

There are a couple of side benefits of having such a registry. First, the Performance Metrics Registry could serve as an inventory of useful and used Performance Metrics, that are normally supported by different implementations of measurement agents. Second, the results of measurements using the Performance Metrics would be comparable even if they are performed by different implementations and in different networks, as the Performance Metric is properly defined. BCP 176 [RFC6576] examines whether the results produced by independent implementations are equivalent in the context of evaluating the completeness and clarity of metric specifications. This BCP defines the standards track advancement testing for (active)

IPPM metrics, and the same process will likely suffice to determine whether Registered Performance Metrics are sufficiently well specified to result in comparable (or equivalent) results. Registered Performance Metrics which have undergone such testing SHOULD be noted, with a reference to the test results.

5. Criteria for Performance Metrics Registration

It is neither possible nor desirable to populate the Performance Metrics Registry with all combinations of Parameters of all Performance Metrics. The Registered Performance Metrics should be:

1. interpretable by the user.

2. implementable by the software designer,

3. deployable by network operators,

4. accurate, for interoperability and deployment across vendors,

5. Operationally useful, so that it has significant industry interest and/or has seen deployment,

6. Sufficiently tightly defined, so that different values for the Run-time Parameters does not change the fundamental nature of the measurement, nor change the practicality of its implementation.

In essence, there needs to be evidence that a candidate Registered Performance Metric has significant industry interest, or has seen deployment, and there is agreement that the candidate Registered Performance Metric serves its intended purpose.

6. Performance Metric Registry: Prior attempt

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".

A couple of interesting additional quotes from RFC 6248 might help understand the issues related to that registry.

1. "It is not believed to be feasible or even useful to register every possible combination of Type P, metric parameters, and Stream parameters using the current structure of the IPPM Metrics Registry."

2. "The registry structure has been found to be insufficiently detailed to uniquely identify IPPM metrics."

3. "Despite apparent efforts to find current or even future users, no one responded to the call for interest in the RFC 4148 registry during the second half of 2010."

The current approach learns from this by tightly defining each Registered Performance Metric with only a few variable (Run-time) Parameters to be specified by the measurement designer, if any. The idea is that entries in the Performance Metrics Registry stem from different measurement methods which require input (Run-time) parameters to set factors like source and destination addresses (which do not change the fundamental nature of the measurement). The downside of this approach is that it could result in a large number of entries in the Performance Metrics Registry. There is agreement 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, some with with questionable usefulness.

6.1. Why this Attempt Will Succeed

As mentioned in the previous section, one of the main issues with the previous registry was that the metrics contained in the registry were too generic to be useful. This document specifies stricter criteria for performance metric registration (see section 6), and imposes a group of Performance Metrics Experts that will provide guidelines to assess if a Performance Metric is properly specified.

Another key difference between this attempt and the previous one is that in this case there is at least one clear user for the Performance Metrics Registry: the LMAP framework and protocol. Because the LMAP protocol will use the Performance Metrics Registry values in its operation, this actually helps to determine if a metric is properly defined. In particular, since we expect that the LMAP control protocol will enable a controller to request a measurement agent to perform a measurement using a given metric by embedding the Performance Metric Registry value in the protocol, a metric is properly specified if it is defined well-enough so that it is possible (and practical) to implement the metric in the measurement agent. This was the failure of the previous attempt: a registry entry with an undefined Type-P (section 13 of RFC 2330 [RFC2330]) allows implementation to be ambiguous.

7. Definition of the Performance Metric Registry

This Performance Metric Registry is applicable to Performance Metrics used for Active Measurement, Passive Measurement, and any other form of Performance Metric. Each category of measurement has unique properties, so some of the columns defined below are not applicable for a given metric category. In this case, the column(s) SHOULD be populated with the "NA" value (Non Applicable). However, the "NA" value MUST NOT be used by any metric in the following columns: Identifier, Name, URI, Status, Requester, Revision, Revision Date, Description. In the future, a new category of metrics could require additional columns, and adding new columns is a recognized form of registry extension. The specification defining the new column(s) MUST give guidelines to populate the new column(s) for existing entries (in general).

The columns of the Performance Metric Registry are defined below. The columns are grouped into "Categories" to facilitate the use of the registry. Categories are described at the 7.x heading level, and columns are at the 7.x.y heading level. The Figure below illustrates this organization. An entry (row) therefore gives a complete description of a Registered Performance Metric.

Each column serves as a check-list item and helps to avoid omissions during registration and expert review.

Registry Categories and Columns, shown as

```
Category
------------------
Column |  Column |

Summary
-------------------------------------------------------------------------
Identifier | Name | URIs | Desc. | Reference | Change Controller | Ver |

Metric Definition
-----------------------------------------
Reference Definition | Fixed Parameters |

Method of Measurement
-----------------------------------------------------------------------
Reference | Packet     | Traffic | Sampling     | Run-time   | Role |
Method    | Stream     | Filter  | Distribution | Parameters |      |
          | Generation |
Output
-----------------------------------------
Type | Reference  | Units | Calibration |
     | Definition |       |             |

Administrative Information
----------------------------------
Status |Request | Rev | Rev.Date |

Comments and Remarks
--------------------
```

7.1. Summary Category

7.1.1. Identifier

A numeric identifier for the Registered Performance Metric. This identifier MUST be unique within the Performance Metric Registry.

The Registered Performance Metric unique identifier is a 16-bit integer (range 0 to 65535).

The Identifier 0 should be Reserved. The Identifier values from 64512 to 65536 are reserved for private use.

When adding newly Registered Performance Metrics to the Performance Metric Registry, IANA should assign the lowest available identifier to the next Registered Performance Metric.

7.1.2. Name

As the name of a Registered Performance Metric is the first thing a potential human implementor will use when determining whether it is suitable for their measurement study, it is important to be as precise and descriptive as possible. In future, users will review the names to determine if the metric they want to measure has already been registered, or if a similar entry is available as a basis for creating a new entry.

Names are composed of the following elements, separated by an underscore character "_":

MetricType_Method_SubTypeMethod_... Spec_Units_Output

- MetricType: a combination of the directional properties and the metric measured, such as:

  RTDelay (Round Trip Delay)

  RTDNS (Response Time Domain Name Service)

  RLDNS (Response Loss Domain Name Service)

  OWDelay (One Way Delay)

  RTLoss (Round Trip Loss)

  OWLoss (One Way Loss)

  OWPDV (One Way Packet Delay Variation)

  OWIPDV (One Way Inter-Packet Delay Variation)

  OWReorder (One Way Packet Reordering)

  OWDuplic (One Way Packet Duplication)

  OWBTC (One Way Bulk Transport Capacity)

  OWMBM (One Way Model Based Metric)

  SPMonitor (Single Point Monitor)

  MPMonitor (Multi-Point Monitor)

- Method: One of the methods defined in [RFC7799], such as:

Active (depends on a dedicated measurement packet stream and observations of the stream)

Passive (depends *solely* on observation of one or more existing packet streams)

HybridType1 (obervations on one stream that combine both active and passive methods)

HybridType2 (obervations on two or more streams that combine both active and passive methods)

Spatial (Spatial Metric of RFC5644)

- SubTypeMethod: One or more sub-types to further describe the features of the entry, such as:

  ICMP (Internet Control Message Protocol)

  IP (Internet Protocol)

  DSCPxx (where xx is replaced by a Diffserv code point)

  UDP (User Datagram Protocol)

  TCP (Transport Control Protocol)

  QUIC (QUIC transport protocol)

  HS (Hand-Shake, such as TCP's 3-way HS)

  Poisson (Packet generation using Poisson distribution)

  Periodic (Periodic packet generation)

  SendOnRcv (Sender keeps one packet in-transit by sending when previous packet arrives)

  PayloadxxxxB (where xxxx is replaced by an integer, the number of octets in the Payload))

  SustainedBurst (Capacity test, worst case)

  StandingQueue (test of bottleneck queue behavior)

  @@@@<add others from MBM draft?>

SubTypeMethod values are separated by a hyphen "-" character, which indicates that they belong to this element, and that their order is unimportant when considering name uniqueness.

- Spec: RFC that specifies this entry in the form RFCXXXXsecY, such as RFC7799sec3. Note: this is not the Primary Reference specification for the metric definition; it will contain the placeholder "RFCXXXXsecY" until the RFC number is assigned to the specifying document, and would remain blank in private registry entries without a corresponding RFC.

- Units: The units of measurement for the output, such as:

  Seconds

  Ratio (unitless)

  Percent (value multiplied by 100)

  Logical (1 or 0)

  Packets

  BPS (Bits per Second)

  PPS (Packets per Second)

  EventTotal (for unit-less counts)

  Multiple (more than one type of unit)

  Enumerated (a list of outcomes)

  Unitless

- Output: The type of output resulting from measurement, such as:

  Singleton

  Raw (multiple Singletons)

  Count

  Minimum

  Maximum

  Median

Mean

95Percentile (95th Percentile)

99Percentile (99th Percentile)

StdDev (Standard Deviation)

Variance

PFI (Pass, Fail, Inconclusive)

FlowRecords (descriptions of flows observed)

LossRatio (lost packets to total packets, <=1)

An example is:

RTDelay_Active_IP-UDP-Periodic_RFCXXXXsecY_Seconds_95Percentile

as described in section 4 of [I-D.ietf-ippm-initial-registry].

Note that private registries following the format described here SHOULD use the prefix "Priv_" on any name to avoid unintended conflicts (further considerations are described in section 10). Private registry entries usually have no specifying RFC, thus the Spec: element has no clear interpretation.

7.1.3. URIs

The URIs column MUST contain a URI [RFC3986] that uniquely identifies the metric. This URI is a URN [RFC2141]. The URI is automatically generated by prepending the prefix

urn:ietf:metrics:perf:

to the metric name. The resulting URI is globally unique.

The URIs column MUST contain a second URI which is a URL [RFC3986] and uniquely identifies and locates the metric entry so it is accessible through the Internet. The URL points to a file containing the human-readable information of exactly one registry entry. Ideally, the file will be HTML-formated and contain URLs to referenced sections of HTML-ized RFCs. The separate files for different entries can be more easily edited and re-used when preparing new entries. The exact composition of each metric URL will be determined by IANA and reside on "iana.org", but there will be some overlap with the URN described above. The major sections of

[I-D.ietf-ippm-initial-registry] provide an example in HTML form (sections 4 and higher).

7.1.4. Description

A Registered Performance Metric description is a written representation of a particular Performance Metrics Registry entry. It supplements the Registered Performance Metric name to help Performance Metrics Registry users select relevant Registered Performance Metrics.

7.1.5. Reference

This entry gives the specification containing the candidate registry entry which was reviewed and agreed, if such an RFC or other specification exists.

7.1.6. Change Controller

This entry names the entity responsible for approving revsions to the regsitry entry, and provides contact information.

7.1.7. Version (of Registry Format)

This entry gives the version number for the registry format used. Formats complying with this memo MUST use 1.0.

7.2. Metric Definition Category

This category includes columns to prompt all necessary details related to the metric definition, including the RFC reference and values of input factors, called fixed parameters, which are left open in the RFC but have a particular value defined by the performance metric.

7.2.1. Reference Definition

This entry provides a reference (or references) to the relevant section(s) of the document(s) that define the metric, as well as any supplemental information needed to ensure an unambiguous definition for implementations. The reference needs to be an immutable document, such as an RFC; for other standards bodies, it is likely to be necessary to reference a specific, dated version of a specification.

7.2.2. Fixed Parameters

Fixed Parameters are Parameters whose value must be specified in the Performance Metrics Registry. The measurement system uses these values.

Where referenced metrics supply a list of Parameters as part of their descriptive template, a sub-set of the Parameters will be designated as Fixed Parameters. As an example for active metrics, Fixed Parameters determine most or all of the IPPM Framework convention "packets of Type-P" as described in [RFC2330], such as transport protocol, payload length, TTL, etc. An example for passive metrics is for RTP packet loss calculation that relies on the validation of a packet as RTP which is a multi-packet validation controlled by MIN_SEQUENTIAL as defined by [RFC3550]. Varying MIN_SEQUENTIAL values can alter the loss report and this value could be set as a Fixed Parameter.

Parameters MUST have well-defined names. For human readers, the hanging indent style is preferred, and any Parameter names and definitions that do not appear in the Reference Method Specification MUST appear in this column (or Run-time Parameters column).

Parameters MUST have a well-specified data format.

A Parameter which is a Fixed Parameter for one Performance Metrics Registry entry may be designated as a Run-time Parameter for another Performance Metrics Registry entry.

7.3. Method of Measurement Category

This category includes columns for references to relevant sections of the RFC(s) and any supplemental information needed to ensure an unambiguous method for implementations.

7.3.1. Reference Method

This entry provides references to relevant sections of the RFC(s) describing the method of measurement, as well as any supplemental information needed to ensure unambiguous interpretation for implementations referring to the RFC text.

Specifically, this section should include pointers to pseudocode or actual code that could be used for an unambigious implementation.

7.3.2. Packet Stream Generation

This column applies to Performance Metrics that generate traffic for a part of their Measurement Method purposes including but not necessarily limited to Active metrics. The generated traffic is referred as stream and this columns describe its characteristics.

Each entry for this column contains the following information:

- Value: The name of the packet stream scheduling discipline
- Reference: the specification where the stream is defined

The packet generation stream may require parameters such as the the average packet rate and distribution truncation value for streams with Poisson-distributed inter-packet sending times. In case such parameters are needed, they should be included either in the Fixed parameter column or in the run time parameter column, depending on wether they will be fixed or will be an input for the metric.

The simplest example of stream specification is Singleton scheduling (see [RFC2330]), where a single atomic measurement is conducted. Each atomic measurement could consist of sending a single packet (such as a DNS request) or sending several packets (for example, to request a webpage). Other streams support a series of atomic measurements in a "sample", with a schedule defining the timing between each transmitted packet and subsequent measurement. Principally, two different streams are used in IPPM metrics, Poisson distributed as described in [RFC2330] and Periodic as described in [RFC3432]. Both Poisson and Periodic have their own unique parameters, and the relevant set of parameters names and values should be included either in the Fixed Parameters column or in the Run-time parameter column.

7.3.3. Traffic Filter

This column applies to Performance Metrics that observe packets flowing through (the device with) the measurement agent i.e. that is not necessarily addressed to the measurement agent. This includes but is not limited to Passive Metrics. The filter specifies the traffic that is measured. This includes protocol field values/ ranges, such as address ranges, and flow or session identifiers.

The traffic filter itself depends on needs of the metric itself and a balance of operators measurement needs and user’s need for privacy. Mechanics for conveying the filter criteria might be the BPF (Berkley Packet Filter) or PSAMP [RFC5475] Property Match Filtering which reuses IPFIX [RFC7012]. An example BPF string for matching TCP/80

traffic to remote destination net 192.0.2.0/24 would be "dst net 192.0.2.0/24 and tcp dst port 80". More complex filter engines might be supported by the implementation that might allow for matching using Deep Packet Inspection (DPI) technology.

The traffic filter includes the following information:

Type: the type of traffic filter used, e.g. BPF, PSAMP, OpenFlow rule, etc. as defined by a normative reference

Value: the actual set of rules expressed

7.3.4. Sampling Distribution

The sampling distribution defines out of all the packets that match the traffic filter, which one of those are actually used for the measurement. One possibility is "all" which implies that all packets matching the Traffic filter are considered, but there may be other sampling strategies. It includes the following information:

Value: the name of the sampling distribution

Reference definition: pointer to the specification where the sampling distribution is properly defined.

The sampling distribution may require parameters. In case such parameters are needed, they should be included either in the Fixed parameter column or in the run time parameter column, depending on wether they will be fixed or will be an input for the metric.

Sampling and Filtering Techniques for IP Packet Selection are documented in the PSAMP (Packet Sampling) [RFC5475], while the Framework for Packet Selection and Reporting, [RFC5474] provides more background information. The sampling distribution parameters might be expressed in terms of the Information Model for Packet Sampling Exports, [RFC5477], and the Flow Selection Techniques, [RFC7014].

7.3.5. Run-time Parameters

Run-Time Parameters are Parameters that must be determined, configured into the measurement system, and reported with the results for the context to be complete. However, the values of these parameters is not specified in the Performance Metrics Registry (like the Fixed Parameters), rather these parameters are listed as an aid to the measurement system implementer or user (they must be left as variables, and supplied on execution).

Where metrics supply a list of Parameters as part of their descriptive template, a sub-set of the Parameters will be designated as Run-Time Parameters.

Parameters MUST have well defined names. For human readers, the hanging indent style is preferred, and the names and definitions that do not appear in the Reference Method Specification MUST appear in this column.

A Data Format for each Run-time Parameter MUST be specified in this column, to simplify the control and implementation of measurement devices. For example, parameters that include an IPv4 address can be encoded as a 32 bit integer (i.e. binary base64 encoded value) or ip-address as defined in [RFC6991]. The actual encoding(s) used must be explicitly defined for each Run-time parameter. IPv6 addresses and options MUST be accomodated, allowing Registered Metrics to be used in either address family.

Examples of Run-time Parameters include IP addresses, measurement point designations, start times and end times for measurement, and other information essential to the method of measurement.

7.3.6. Role

In some method of measurements, there may be several roles defined e.g. on a one-way packet delay active measurement, there is one measurement agent that generates the packets and the other one that receives the packets. This column contains the name of the role for this particular entry. In the previous example, there should be two entries in the registry, one for each role, so that when a measurement agent is instructed to perform the one way delay source metric know that it is supposed to generate packets. The values for this field are defined in the reference method of measurement.

7.4. Output Category

For entries which involve a stream and many singleton measurements, a statistic may be specified in this column to summarize the results to a single value. If the complete set of measured singletons is output, this will be specified here.

Some metrics embed one specific statistic in the reference metric definition, while others allow several output types or statistics.

7.4.1. Type

This column contains the name of the output type. The output type defines a single type of result that the metric produces. It can be the raw results (packet send times and singleton metrics), or it can be a summary statistic. The specification of the output type MUST define the format of the output. In some systems, format specifications will simplify both measurement implementation and collection/storage tasks. Note that if two different statistics are required from a single measurement (for example, both "Xth percentile mean" and "Raw"), then a new output type must be defined ("Xth percentile mean AND Raw"). See the Naming section above for a list of Output Types.

7.4.2. Reference Definition

This column contains a pointer to the specification(s) where the output type and format are defined.

7.4.3. Metric Units

The measured results must be expressed using some standard dimension or units of measure. This column provides the units.

When a sample of singletons (see Section 11 of[RFC2330] for definitions of these terms) is collected, this entry will specify the units for each measured value.

7.4.4. Calibration

Some specifications for Methods of Measurement include the possibility to perform an error calibration. Section 3.7.3 of [RFC7679] is one example. In the registry entry, this field will identify a method of calibration for the metric, and when available, the measurement system SHOULD perform the calibration when requested and produce the output with an indication that it is the restult of a calbration method. In-situ calibration could be enabled with an internal loopback that includes as much of the measurement system as possible, performs address manipulation as needed, and provides some form of isolation (e.g., deterministic delay) to avoid send-receive interface contention. Some portion of the random and systematic error can be characterized this way.

For one-way delay measurements, the error calibration must include an assessment of the internal clock synchronization with its external reference (this internal clock is supplying timestamps for measurement). In practice, the time offsets of clocks at both the source and destination are needed to estimate the systematic error

due to imperfect clock synchronization (the time offsets are smoothed, thus the random variation is not usually represented in the results).

Both internal loopback calibration and clock synchronization can be used to estimate the *available accuracy* of the Output Metric Units. For example, repeated loopback delay measurements will reveal the portion of the Output result resolution which is the result of system noise, and thus inaccurate.

7.5. Administrative information

7.5.1. Status

The status of the specification of this Registered Performance Metric. Allowed values are 'current' and 'deprecated'. All newly defined Information Elements have 'current' status.

7.5.2. Requester

The requester for the Registered Performance Metric. The requester MAY be a document, such as RFC, or person.

7.5.3. Revision

The revision number of a Registered Performance Metric, starting at 0 for Registered Performance Metrics at time of definition and incremented by one for each revision.

7.5.4. Revision Date

The date of acceptance or the most recent revision for the Registered Performance Metric.

7.6. Comments and Remarks

Besides providing additional details which do not appear in other categories, this open Category (single column) allows for unforeseen issues to be addressed by simply updating this informational entry.

8. The Life-Cycle of Registered Performance Metrics

Once a Performance Metric or set of Performance Metrics has been identified for a given application, candidate Performance Metrics Registry entry specifications prepared in accordance with Section 7 should be submitted to IANA to follow the process for review by the Performance Metric Experts, as defined below. This process is also

used for other changes to the Performance Metric Registry, such as deprecation or revision, as described later in this section.

It is also desirable that the author(s) of a candidate Performance Metrics Registry entry seek review in the relevant IETF working group, or offer the opportunity for review on the working group mailing list.

8.1. Adding new Performance Metrics to the Performance Metrics Registry

Requests to add Registered Performance Metrics in the Performance Metric Registry are submitted to IANA, which forwards the request to a designated group of experts (Performance Metric Experts) appointed by the IESG; these are the reviewers called for by the Expert Review [RFC8126]policy defined for the Performance Metric Registry. The Performance Metric Experts review the request for such things as compliance with this document, compliance with other applicable Performance Metric-related RFCs, and consistency with the currently defined set of Registered Performance Metrics.

Authors are expected to review compliance with the specifications in this document to check their submissions before sending them to IANA.

The Performance Metric Experts should endeavor to complete referred reviews in a timely manner. If the request is acceptable, the Performance Metric Experts signify their approval to IANA, which updates the Performance Metric Registry. If the request is not acceptable, the Performance Metric Experts can coordinate with the requester to change the request to be compliant. The Performance Metric Experts may also choose in exceptional circumstances to reject clearly frivolous or inappropriate change requests outright.

This process should not in any way be construed as allowing the Performance Metric Experts to overrule IETF consensus. Specifically, any Registered Performance Metrics that were added with IETF consensus require IETF consensus for revision or deprecation.

Decisions by the Performance Metric Experts may be appealed as in Section 7 of [RFC8126].

8.2. Revising Registered Performance Metrics

A request for Revision is only permissible when the changes maintain backward-compatibility with implementations of the prior Performance Metrics Registry entry describing a Registered Performance Metric (entries with lower revision numbers, but the same Identifier and Name).

The purpose of the Status field in the Performance Metric Registry is to indicate whether the entry for a Registered Performance Metric is 'current' or 'deprecated'.

In addition, no policy is defined for revising the Performance Metric entries in the IANA Regsirty or addressing errors therein. To be certain, changes and deprecations within the Performance Metric Registry are not encouraged, and should be avoided to the extent possible. However, in recognition that change is inevitable, the provisions of this section address the need for revisions.

Revisions are initiated by sending a candidate Registered Performance Metric definition to IANA, as in Section 8, identifying the existing Performance Metrics Registry entry.

The primary requirement in the definition of a policy for managing changes to existing Registered Performance Metrics is avoidance of interoperability problems; Performance Metric Experts must work to maintain interoperability above all else. Changes to Registered Performance Metrics may only be done in an inter-operable way; necessary changes that cannot be done in a way to allow interoperability with unchanged implementations must result in the creation of a new Registered Performance Metric and possibly the deprecation of the earlier metric.

A change to a Registered Performance Metric SHALL be determined to be backward-compatible only when:

1. it involves the correction of an error that is obviously only editorial; or

2. it corrects an ambiguity in the Registered Performance Metric's definition, which itself leads to issues severe enough to prevent the Registered Performance Metric's usage as originally defined; or

3. it corrects missing information in the metric definition without changing its meaning (e.g., the explicit definition of 'quantity' semantics for numeric fields without a Data Type Semantics value); or

4. it harmonizes with an external reference that was itself corrected.

If an Performance Metric revision is deemed permissible by the Performance Metric Experts, according to the rules in this document, IANA makes the change in the Performance Metric Registry. The

requester of the change is appended to the requester in the Performance Metrics Registry.

Each Registered Performance Metric in the Performance Metrics Registry has a revision number, starting at zero. Each change to a Registered Performance Metric following this process increments the revision number by one.

When a revised Registered Performance Metric is accepted into the Performance Metric Registry, the date of acceptance of the most recent revision is placed into the revision Date column of the registry for that Registered Performance Metric.

Where applicable, additions to Registered Performance Metrics in the form of text Comments or Remarks should include the date, but such additions may not constitute a revision according to this process.

Older version(s) of the updated metric entries are kept in the registry for archival purposes. The older entries are kept with all fields unmodified (version, revision date) except for the status field that SHALL be changed to "Deprecated".

8.3. Deprecating Registered Performance Metrics

Changes that are not permissible by the above criteria for Registered Performance Metric's revision may only be handled by deprecation. A Registered Performance Metric MAY be deprecated and replaced when:

1. the Registered Performance Metric definition has an error or shortcoming that cannot be permissibly changed as in Section 8.2 Revising Registered Performance Metrics; or

2. the deprecation harmonizes with an external reference that was itself deprecated through that reference's accepted deprecation method.

A request for deprecation is sent to IANA, which passes it to the Performance Metric Experts for review. When deprecating an Performance Metric, the Performance Metric description in the Performance Metric Registry must be updated to explain the deprecation, as well as to refer to any new Performance Metrics created to replace the deprecated Performance Metric.

The revision number of a Registered Performance Metric is incremented upon deprecation, and the revision Date updated, as with any revision.

The use of deprecated Registered Performance Metrics should result in a log entry or human-readable warning by the respective application.

Names and Metric IDs of deprecated Registered Performance Metrics must not be reused.

The deprecated entries are kept with all fields unmodified, except the version, revision date, and the status field (changed to "Deprecated").

9. Security considerations

This draft defines a registry structure, and does not itself introduce any new security considerations for the Internet. The definition of Performance Metrics for this registry may introduce some security concerns, but the mandatory references should have their own considerations for secuity, and such definitions should be reviewed with security in mind if the security considerations are not covered by one or more reference standards.

10. IANA Considerations

This document requests the following IANA Actions.

10.1. New Namespace Assignments

This document requests the allocation of the URI prefix urn:ietf:metrics for the purpose of generating URIs for metrics in general. The registration procedure for the new "metrics" URN sub-namespace is IETF Review.

This document requests the allocation of the URI prefix urn:ietf:metrics:perf for the purpose of generating URIs for Registered Performance Metrics. The registration procedures for the new "perf" URN sub-namespace are Expert Review or IETF Standards Action, and coordinated with the entries added to the New Performance Metrics Registry (see below).

10.2. Performance Metric Name Elements

This document specifies the procedure for Performance Metrics Name Element Registry setup. IANA is requested to create a new set of registries for Performance Metric Name Elements called "IETF URN Sub-namespace for Registered Performance Metric Name Elements" (urn:ietf:metrics:perf). Each Registry, whose names are listed below:

MetricType:

Method:

SubTypeMethod:

Spec:

Units:

Output:

will contain the current set of possibilities for Performance Metric Registry Entry Names.

To populate the IETF URN Sub-namespace for Registered Performance Metric Name Elements at creation, the IANA is asked to use the lists of values for each name element listed in Section 7.1.2. The Name Elements in each registry are case-sensitive.

When preparing a Metric entry for Registration, the developer SHOULD choose Name elements from among the registered elements. However, if the proposed metric is unique in a significant way, it may be necessary to propose a new Name element to properly describe the metric, as described below.

A candidate Metric Entry RFC or document for Expert Review would propose one or more new element values required to describe the unique entry, and the new name element(s) would be reviewed along with the metric entry. New assignments for IETF URN Sub-namespace for Registered Performance Metric Name Elements will be administered by IANA through Expert Review [RFC8126], i.e., review by one of a group of experts, the Performance Metric Experts, who are appointed by the IESG upon recommendation of the Transport Area Directors.

10.3. New Performance Metrics Registry

This document specifies the procedure for Performance Metrics Registry setup. IANA is requested to create a new registry for Performance Metrics called "Registered Performance Metrics". This Registry will contain the following Summary columns:

Identifier:

Name:

URIs:

Description:

Reference:

Change Controller:

Version:

Descriptions of these columns and additional information found in the template for registry entries (categories and columns) are further defined in section Section 7.

The "Identifier" 0 should be Reserved. "The Identifier" values from 64512 to 65536 are reserved for private use.

Names starting with the prefix Priv_ are reserved for private use, and are not considered for registration. The "Name" column entries are further defined in section Section 7.

The "URIs" column will have a URL to the full template of each registry entry, and the linked text may be the URN itself. The template shall be HTML-ized to aid the reader, with links to reference RFCs (similar to the way that Internet Drafts are HTML-ized, the same tool can perform the function).

The "Reference" column will include an RFC, an approved specification from another standards body, or the contact person.

New assignments for Performance Metric Registry will be administered by IANA through Expert Review [RFC8126], i.e., review by one of a group of experts, the Performance Metric Experts, who are appointed by the IESG upon recommendation of the Transport Area Directors. The experts can be initially drawn from the Working Group Chairs, document editors, and members of the Performance Metrics Directorate, among other sources of experts.

Extensions of the Performance Metric Registry require IETF Standards Action. Only one form of registry extension is envisaged:

1. Adding columns, or both categories and columns, to accommodate unanticipated aspects of new measurements and metric categories.

If the Performance Metrics Registry is extended in this way, the Version number of future entries complying with the extension SHALL be incremented (either in the unit or tenths digit, depending on the degree of extension.

11. Acknowledgments

Thanks to Brian Trammell and Bill Cerveny, IPPM chairs, for leading some brainstorming sessions on this topic. Thanks to Barbara Stark and Juergen Schoenwaelder for the detailed feedback and suggestions. Thanks to Andrew McGregor for suggestions on metric naming. Thanks to Michelle Cotton for her early IANA review, and to Amanda Barber for answering questions related to the presentation of the registry and accessibility of the complete template via URL.

12. References

12.1. Normative References

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, DOI 10.17487/RFC2026, October 1996, <https://www.rfc-editor.org/info/rfc2026>.

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

[RFC2141] Moats, R., "URN Syntax", RFC 2141, DOI 10.17487/RFC2141, May 1997, <https://www.rfc-editor.org/info/rfc2141>.

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <https://www.rfc-editor.org/info/rfc2330>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4148] Stephan, E., "IP Performance Metrics (IPPM) Metrics Registry", BCP 108, RFC 4148, DOI 10.17487/RFC4148, August 2005, <https://www.rfc-editor.org/info/rfc4148>.

[RFC6248] Morton, A., "RFC 4148 and the IP Performance Metrics (IPPM) Registry of Metrics Are Obsolete", RFC 6248, DOI 10.17487/RFC6248, April 2011, <https://www.rfc-editor.org/info/rfc6248>.

[RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, DOI 10.17487/RFC6390, October 2011, <https://www.rfc-editor.org/info/rfc6390>.

[RFC6576] Geib, R., Ed., Morton, A., Fardid, R., and A. Steinmitz, "IP Performance Metrics (IPPM) Standard Advancement Testing", BCP 176, RFC 6576, DOI 10.17487/RFC6576, March 2012, <https://www.rfc-editor.org/info/rfc6576>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

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

12.2. Informative References

[I-D.ietf-ippm-initial-registry] Morton, A., Bagnulo, M., Eardley, P., and K. D'Souza, "Initial Performance Metric Registry Entries", draft-ietf-ippm-initial-registry-08 (work in progress), October 2018.

[RFC2679] Almes, G., Kalidindi, S., and M. Zekauskas, "A One-way Delay Metric for IPPM", RFC 2679, DOI 10.17487/RFC2679, September 1999, <https://www.rfc-editor.org/info/rfc2679>.

[RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <https://www.rfc-editor.org/info/rfc2681>.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <https://www.rfc-editor.org/info/rfc3393>.

[RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, DOI 10.17487/RFC3432, November 2002, <https://www.rfc-editor.org/info/rfc3432>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <https://www.rfc-editor.org/info/rfc3550>.

[RFC3611] Friedman, T., Ed., Caceres, R., Ed., and A. Clark, Ed., "RTP Control Protocol Extended Reports (RTCP XR)", RFC 3611, DOI 10.17487/RFC3611, November 2003, <https://www.rfc-editor.org/info/rfc3611>.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, DOI 10.17487/RFC4566, July 2006, <https://www.rfc-editor.org/info/rfc4566>.

[RFC5474] Duffield, N., Ed., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, DOI 10.17487/RFC5474, March 2009, <https://www.rfc-editor.org/info/rfc5474>.

[RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, DOI 10.17487/RFC5475, March 2009, <https://www.rfc-editor.org/info/rfc5475>.

[RFC5477] Dietz, T., Claise, B., Aitken, P., Dressler, F., and G. Carle, "Information Model for Packet Sampling Exports", RFC 5477, DOI 10.17487/RFC5477, March 2009, <https://www.rfc-editor.org/info/rfc5477>.

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <https://www.rfc-editor.org/info/rfc5481>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC6035] Pendleton, A., Clark, A., Johnston, A., and H. Sinnreich, "Session Initiation Protocol Event Package for Voice Quality Reporting", RFC 6035, DOI 10.17487/RFC6035, November 2010, <https://www.rfc-editor.org/info/rfc6035>.

[RFC6776] Clark, A. and Q. Wu, "Measurement Identity and Information Reporting Using a Source Description (SDES) Item and an RTCP Extended Report (XR) Block", RFC 6776, DOI 10.17487/RFC6776, October 2012, <https://www.rfc-editor.org/info/rfc6776>.

[RFC6792] Wu, Q., Ed., Hunt, G., and P. Arden, "Guidelines for Use of the RTP Monitoring Framework", RFC 6792, DOI 10.17487/RFC6792, November 2012, <https://www.rfc-editor.org/info/rfc6792>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7003] Clark, A., Huang, R., and Q. Wu, Ed., "RTP Control Protocol (RTCP) Extended Report (XR) Block for Burst/Gap Discard Metric Reporting", RFC 7003, DOI 10.17487/RFC7003, September 2013, <https://www.rfc-editor.org/info/rfc7003>.

[RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <https://www.rfc-editor.org/info/rfc7012>.

[RFC7014] D'Antonio, S., Zseby, T., Henke, C., and L. Peluso, "Flow Selection Techniques", RFC 7014, DOI 10.17487/RFC7014, September 2013, <https://www.rfc-editor.org/info/rfc7014>.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/info/rfc7594>.

[RFC7679] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <https://www.rfc-editor.org/info/rfc7679>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <https://www.rfc-editor.org/info/rfc7799>.

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

Benoit Claise
Cisco Systems, Inc.
De Kleetlaan 6a b1
1831 Diegem
Belgium

Email: bclaise@cisco.com

Philip Eardley
BT
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

Aamer Akhter
Consultant
118 Timber Hitch
Cary, NC
USA

Email: aakhter@gmail.com

IPPM WG
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2019

R. Civil
Ciena Corporation
A. Morton
AT&T Labs
R. Rahman
Cisco Systems
M. Jethanandani
Xoriant Corporation
K. Pentikousis, Ed.
Travelping
July 2, 2018

Two-Way Active Measurement Protocol (TWAMP) Data Model
draft-ietf-ippm-twamp-yang-13

Abstract

This document specifies a data model for client and server implementations of the Two-Way Active Measurement Protocol (TWAMP). The document defines the TWAMP data model through Unified Modeling Language (UML) class diagrams and formally specifies it using a NDMA-compliant YANG model.

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 https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on January 3, 2019.

Table of Contents

1. Introduction

The Two-Way Active Measurement Protocol (TWAMP) [RFC5357] is used to measure network performance parameters such as latency, bandwidth, and packet loss by sending probe packets and measuring their experience in the network. To date, TWAMP implementations do not come with a standard management framework, and, as such, implementers have no choice except to provide a proprietary mechanism. This document addresses this gap by defining the model using UML [UML] class diagrams, and formally specifying a NMDA-complaint [RFC8342] TWAMP data model using YANG 1.1 [RFC7950].

1.1. Motivation

In current TWAMP deployments the lack of a standardized data model limits the flexibility to dynamically instantiate TWAMP-based measurements across equipment from different vendors. In large, virtualized, and dynamically instantiated infrastructures where network functions are placed according to orchestration algorithms, proprietary mechanisms for managing TWAMP measurements pose severe limitations with respect to programmability.

Two major trends call for standardizing TWAMP management aspects. First, it is expected that in the coming years large-scale and multi-vendor TWAMP deployments will become the norm. From an operations perspective, using several vendor-specific TWAMP configuration mechanisms when one standard mechanism could provide an alternative is expensive and inefficient. Second, the increasingly software-defined and virtualized nature of network infrastructures, based on dynamic service chains [NSC] and programmable control and management planes Software-Defined Networking (SDN): Layers and Architecture Terminology [RFC7426] requires a well-defined data model for TWAMP implementations. This document defines such a TWAMP data model and specifies it formally using the YANG 1.1 [RFC7950] data modeling language.

Note to RFC Editor:

Please replace the date 2018-07-02 in Section 5.2 of the draft with the date of publication of this draft as a RFC. Also, replace reference to RFC XXXX, and draft-ietf-ippm-port-twamp-test with the RFC numbers assigned to the drafts.

1.2. Terminology

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

14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.3. Document Organization

The rest of this document is organized as follows. Section 2 presents the scope and applicability of this document. Section 3 provides a high-level overview of the TWAMP data model. Section 4 details the configuration parameters of the data model and Section 5 specifies in YANG the TWAMP data model. Section 6 lists illustrative examples which conform to the YANG data model specified in this document. Appendix A elaborates these examples further.

2. Scope, Model, and Applicability

The purpose of this document is the specification of a vendor-independent data model for TWAMP implementations.

Figure 1 illustrates a redrawn version of the TWAMP logical model found in Section 1.2 of TWAMP [RFC5357]. The figure is annotated with pointers to the UML [UML] diagrams provided in this document and associated with the data model of the four logical entities in a TWAMP deployment, namely the TWAMP Control-Client, Server, Session-Sender and Session-Reflector. A UML [UML] Notation Guide is available in Section 5 of the said document.

As per TWAMP [RFC5357], unlabeled links in Figure 1 are left unspecified and may be proprietary protocols.

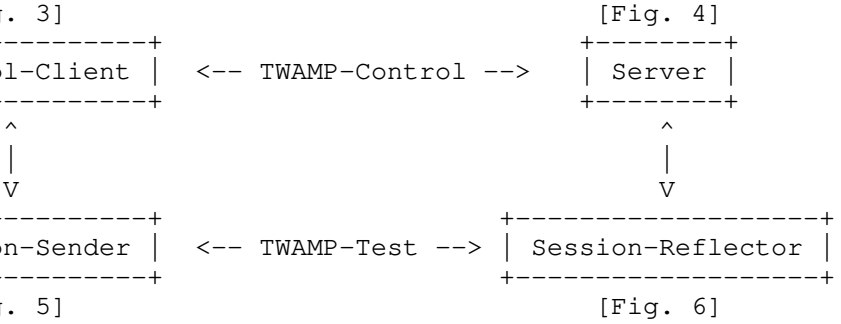


Figure 1: Annotated TWAMP logical model

As per TWAMP [RFC5357], a TWAMP implementation may follow a simplified logical model, in which the same node acts both as Control-Client and Session-Sender, while another node acts at the same time as TWAMP Server and Session-Reflector. Figure 2 illustrates this simplified logical model and indicates the

interaction between the TWAMP configuration client and server using, for instance, NETCONF [RFC6241] or RESTCONF [RFC8040].

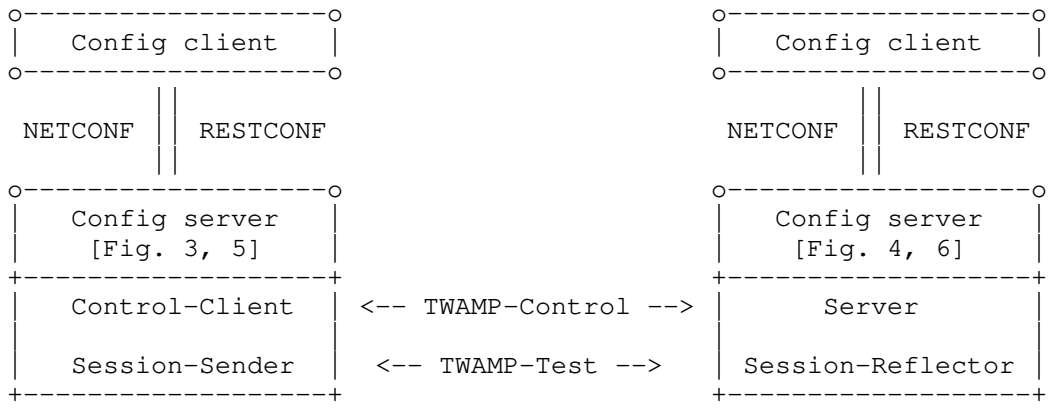


Figure 2: Simplified TWAMP model and protocols

The data model defined in this document is orthogonal to the specific protocol used between the Config client and Config server to communicate the TWAMP configuration parameters.

Operational actions such as how TWAMP-Test sessions are started and stopped, how performance measurement results are retrieved, or how stored results are cleared, and so on, are not addressed by the configuration model defined in this document. As noted above, such operational actions are not part of the TWAMP specification TWAMP [RFC5357] and hence are out of scope of this document. See also Appendix B. In addition, for operational state, current work in Registry for Performance Metrics [I-D.ietf-ippm-metric-registry], can be used to develop an independent model for the performance metrics that need to be captured and retrieved.

3. Data Model Overview

The TWAMP data model includes four categories of configuration items.

First, global configuration items relate to parameters that are set on a per device level. For example, the administrative status of the device with respect to whether it allows TWAMP sessions and, if so, in what capacity (e.g. Control-Client, Server or both), is a typical instance of a global configuration item.

A second category includes attributes that can be configured on a per TWAMP-Control connection basis, such as the Server IP address.

A third category includes attributes related to per TWAMP-Test session attributes, for instance setting different values in the Differentiated Services Code Point (DSCP) field.

Finally, the data model includes attributes that relate to the operational state of the TWAMP implementation.

As the TWAMP data model is described in the remaining sections of this document, readers should keep in mind the functional entity grouping illustrated in Figure 1.

3.1. Control-Client

A TWAMP Control-Client has an administrative status field set at the device level that indicates whether the node is enabled to function as such.

Each TWAMP Control-Client is associated with zero or more TWAMP-Control connections. The main configuration parameters of each control connection are:

- A name which can be used to uniquely identify at the Control-Client a particular control connection. This name is necessary for programmability reasons because at the time of creation of a TWAMP-Control connection not all IP and TCP port number information needed to uniquely identify the connection is available.
- The IP address of the interface the Control-Client will use for connections.
- The IP address of the remote TWAMP Server.
- Authentication and encryption attributes such as KeyID, Token and the Client Initialization Vector (Client-IV); see also Section 3.1 in OWAMP [RFC4656] and Randomness Requirements for Security [RFC4086].

Each TWAMP-Control connection, in turn, is associated with zero or more TWAMP-Test sessions. For each test session, the following configuration items should be noted:

- The test session name uniquely identifies a particular test session at the Control-Client and Session-Sender. Similar to the control connections above, this unique test session name is needed because at the time of creation of a TWAMP-Test session, for example, the source UDP port number is not known to uniquely identify the test session.

- The IP address and UDP port number of the Session-Sender on the path under test by TWAMP.

- The IP address and UDP port number of the Session-Reflector on said path.

- Information pertaining to the test packet stream, such as the test starting time, which performance metric is to be used, as defined in Registry for Performance Metrics [I-D.ietf-ippm-metric-registry], or whether the test should be repeated.

3.2. Server

Each TWAMP Server has an administrative status field set at the device level to indicate whether the node is enabled to function as a TWAMP Server.

Each Server is associated with zero or more TWAMP-Control connections. Each control connection is uniquely identified by the 4-tuple {Control-Client IP address, Control-Client TCP port number, Server IP address, Server TCP port}. Control connection configuration items on a TWAMP Server are read-only.

3.3. Session-Sender

A TWAMP Session-Sender has an administrative status field set at the device level that indicates whether the node is enabled to function as such.

There is one Session-Sender instance for each TWAMP-Test session that is initiated from the sending device. Primary configuration fields include:

- The test session name MUST be identical to the corresponding test session name on the TWAMP Control-Client (Section 3.1).

- The control connection name, which along with the test session name uniquely identify the TWAMP Session-Sender instance.

- Information pertaining to the test packet stream, such as, the number of test packets and the packet distribution to be employed; see also Network performance measurement with periodic streams [RFC3432].

3.4. Session-Reflector

Each TWAMP Session-Reflector has an administrative status field set at the device level to indicate whether the node is enabled to function as such.

Each Session-Reflector is associated with zero or more TWAMP-Test sessions. For each test session, the REFWAIT timeout parameter, which determines whether to discontinue the session if no packets have been received (TWAMP [RFC5357], Section 4.2), can be configured.

Read-only access to other data model parameters, such as the Sender IP address, is foreseen. Each test session can be uniquely identified by the 4-tuple mentioned in Section 3.2.

4. Data Model Parameters

This section defines the TWAMP data model using UML [UML] and introduces selected parameters associated with the four TWAMP logical entities. The complete TWAMP data model specification is provided in the YANG module presented in Section 5.2.

4.1. Control-Client

The client container (see Figure 3) holds items that are related to the configuration of the TWAMP Control-Client logical entity (recall Figure 1).

The client container includes an administrative configuration parameter (client/admin-state) that indicates whether the device is allowed to initiate TWAMP-Control connections.

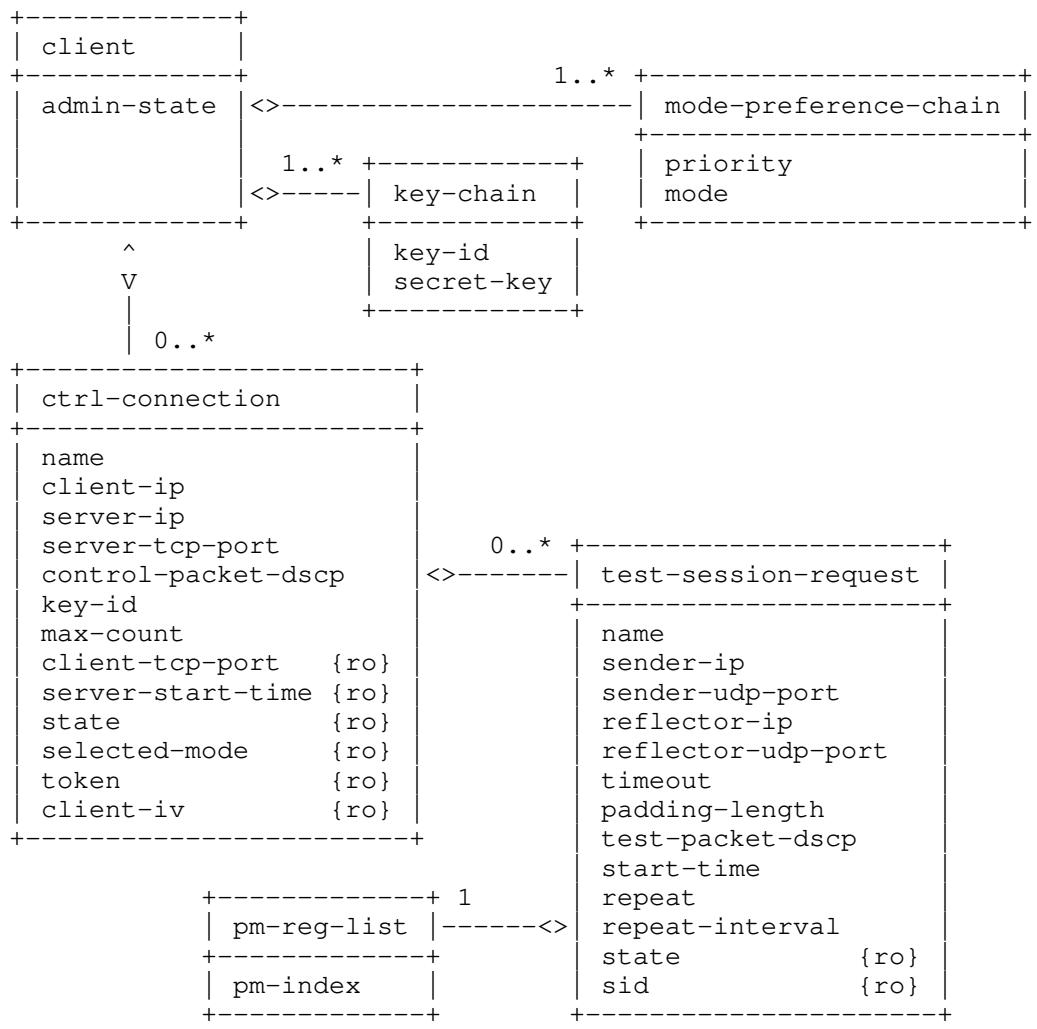


Figure 3: TWAMP Control-Client UML class diagram

The client container holds a list (mode-preference-chain) which specifies the Mode values according to their preferred order of use by the operator of this Control-Client, including the authentication and encryption Modes. Specifically, mode-preference-chain lists the mode and its corresponding priority, as a 16-bit unsigned integer. Values for the priority start with zero, the highest priority, and decreasing priority value is indicated by every increase in value by one.

Depending on the Modes available in the Server Greeting, the Control-Client MUST choose the highest priority Mode from the configured mode-preference-chain list.

Note that the list of preferred Modes may set multiple bit positions independently, such as when referring to the extended TWAMP features in Mixed Security Mode for TWAMP [RFC5618], Individual Session Control Feature for TWAMP [RFC5938], TWAMP Reflect Octets and Symmetrical Size Features [RFC6038], and IKEv2-Derived Shared Secret Key for OWAMP and TWAMP [RFC7717]. If the Control-Client cannot determine an acceptable Mode, or when the bit combinations do not make sense, e.g., both authenticated and unauthenticated bit are set, it MUST respond with zero Mode bits set in the Set-up Response message, indicating it will not continue with the control connection.

In addition, the client container holds a list named key-chain which relates key-id with the respective secret-key. Both the Server and the Control-Client use the same mappings from key-id to secret-key (in Figure 3); in order for this to work properly, key-id must be unique across all systems in the administrative domain. The Server, being prepared to conduct sessions with more than one Control-Client, uses key-id to choose the appropriate secret-key; a Control-Client would typically have different secret keys for different Servers. The secret-key is the shared secret, of type binary and the length SHOULD contain at least 128 bits of entropy. The key-id and secret-key encoding SHOULD follow Section 9.8 of YANG [RFC7950]. The derived key length (dkLen in PKCS #5: Password-Based Cryptography Specification Version 2.1 [RFC8018]) MUST be 16 octets for the AES Session-key used for encryption and 32 octets for the HMAC-SHA1 Session-key used for authentication; see also Section 6.10 of OWAMP [RFC4656].

Each client container also holds a list of control connections, where each item in the list describes a TWAMP control connection initiated by this Control-Client. There SHALL be one ctrl-connection per TWAMP-Control (TCP) connection that is to be initiated from this device.

In turn, each ctrl-connection holds a test-session-request list. Each test-session-request holds information associated with the Control-Client for this test session. This includes information associated with the Request-TW-Session/Accept-Session message exchange (see Section 3.5 of TWAMP [RFC5357]).

There SHALL be one instance of test-session-request for each TWAMP-Test session that is to be negotiated by this TWAMP-Control connection via a Request-TW-Session/Accept-Session exchange.

The Control-Client is also responsible for scheduling TWAMP-Test sessions, therefore test-session-request holds information related to these actions (e.g. pm-index, repeat-interval).

4.2. Server

The server container (see Figure 4) holds items that are related to the configuration of the TWAMP Server logical entity (recall Figure 1).

The server container includes an administrative configuration parameter (server/admin-state) that indicates whether the device is allowed to receive TWAMP-Control connections.

A device operating in the Server role cannot configure attributes on a per TWAMP-Control connection basis, as it has no foreknowledge of the incoming TWAMP-Control connections to be received. Consequently, any parameter that the Server might want to apply to an incoming control connection must be configured at the overall Server level and applied to all incoming TWAMP-Control connections.

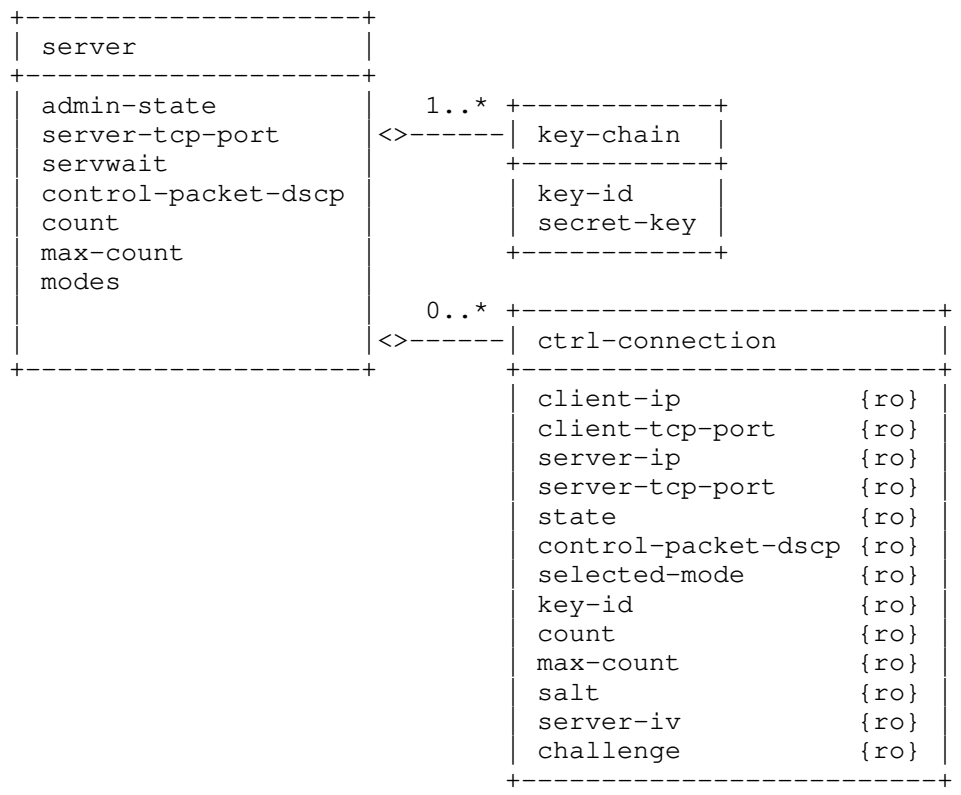


Figure 4: TWAMP Server UML class diagram

Each server container holds a list named key-chain which relates key-id with the respective secret-key. As mentioned in Section 4.1, both the Server and the Control-Client use the same mapping from key-id to shared secret-key; in order for this to work properly, key-id must be unique across all the systems in the administrative domain. The Server, being prepared to conduct sessions with more than one Control-Client, uses key-id to choose the appropriate secret-key; a Control-Client would typically have different secret keys for different Servers. The key-id tells the Server which shared secret-key the Control-Client wishes to use for authentication or encryption.

Each incoming control connection active on the Server is represented by a ctrl-connection. There SHALL be one ctrl-connection per incoming TWAMP-Control (TCP) connection that is received and active on the Server. Each ctrl-connection can be uniquely identified by the 4-tuple {client-ip, client-tcp-port, server-ip, server-tcp-port}. All items in the ctrl-connection list are read-only.

4.3. Session-Sender

The session-sender container, illustrated in Figure 5, holds items that are related to the configuration of the TWAMP Session-Sender logical entity.

The session-sender container includes an administrative parameter (session-sender/admin-state) that controls whether the device is allowed to initiate TWAMP-Test sessions.

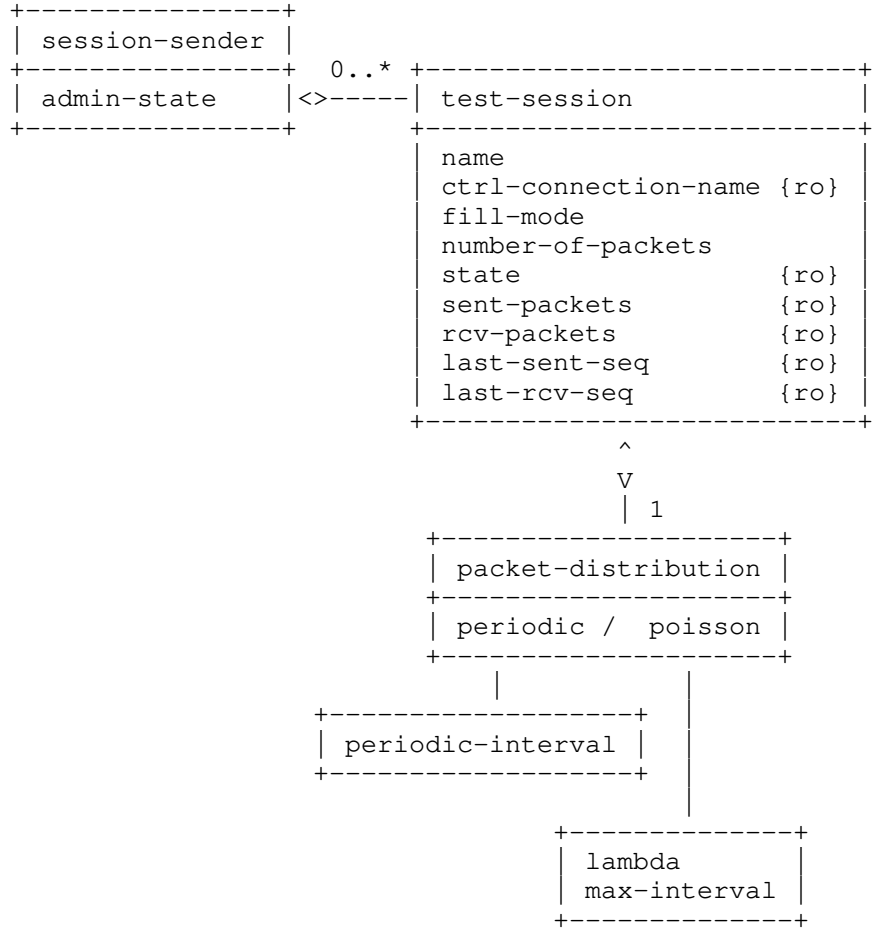


Figure 5: TWAMP Session-Sender UML class diagram

Each TWAMP-Test session initiated by the Session-Sender will be represented by an instance of a test-session object. There SHALL be

one instance of test-session for each TWAMP-Test session for which packets are being sent.

4.4. Session-Reflector

The session-reflector container, illustrated in Figure 6, holds items that are related to the configuration of the TWAMP Session-Reflector logical entity.

The session-reflector container includes an administrative parameter (session-reflector/admin-state) that controls whether the device is allowed to respond to incoming TWAMP-Test sessions.

A device operating in the Session-Reflector role cannot configure attributes on a per-session basis, as it has no foreknowledge of what incoming sessions it will receive. As such, any parameter that the Session-Reflector might want to apply to an incoming TWAMP-Test session must be configured at the overall Session-Reflector level and are applied to all incoming sessions.

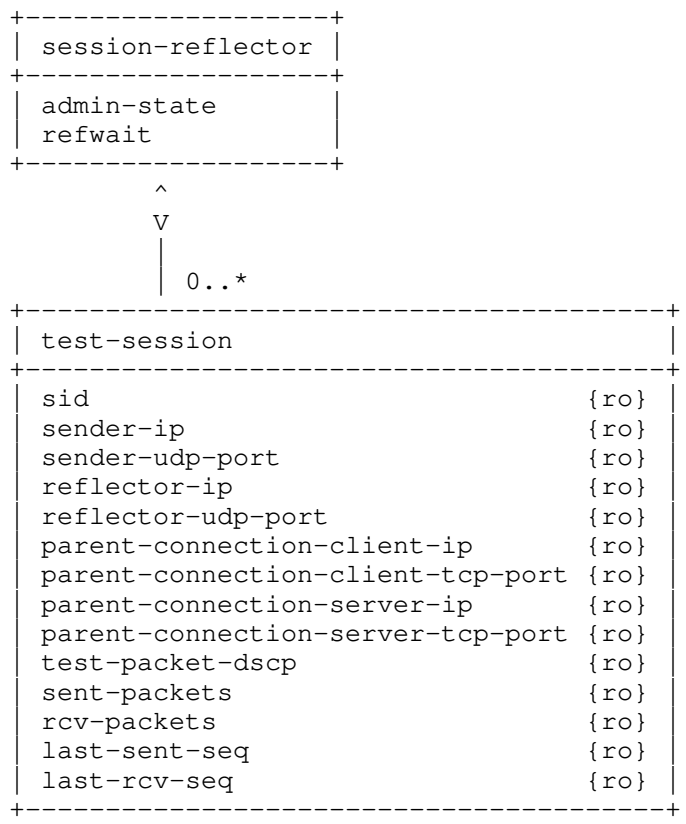


Figure 6: TWAMP Session-Reflector UML class diagram

Each incoming TWAMP-Test session that is active on the Session-Reflector SHALL be represented by an instance of a test-session object. All items in the test-session object are read-only.

Instances of test-session are indexed by a session identifier (sid). This value is auto-allocated by the TWAMP Server as test session requests are received, and communicated back to the Control-Client in the SID field of the Accept-Session message; see Section 4.3 of TWAMP Reflect Octets and Symmetrical Size Features [RFC6038].

When attempting to retrieve operational data for active test sessions from a Session-Reflector device, the user will not know what sessions are currently active on that device, or what SIDs have been auto-allocated for these test sessions. If the user has network access to the Control-Client device, then it is possible to read the data for this session under client/ctrl-connection/test-session-request/sid and obtain the SID (see Figure 3). The user may then use this SID

value as an index to retrieve an individual session-reflector/test-session instance on the Session-Reflector device.

If the user has no network access to the Control-Client device, then the only option is to retrieve all test-session instances from the Session-Reflector device, and then pick out specific test-session instances of interest to the user. This could be problematic if a large number of test sessions are currently active on that device.

Each Session-Reflector TWAMP-Test session contains the following 4-tuple: {parent-connection-client-ip, parent-connection-client-tcp-port, parent-connection-server-ip, parent-connection-server-tcp-port}. This 4-tuple MUST correspond to the equivalent 4-tuple {client-ip, client-tcp-port, server-ip, server-tcp-port} in server/ctrl-connection. This 4-tuple allows the user to trace back from the TWAMP-Test session to the (parent) TWAMP-Control connection that negotiated this test session.

5. Data Model

This section formally specifies the TWAMP data model using YANG.

5.1. YANG Tree Diagram

This section presents a simplified graphical representation of the TWAMP data model using a YANG tree diagram. Readers should keep in mind that the limit of 72 characters per line forces us to introduce artificial line breaks in some tree diagram nodes. Tree diagrams used in this document follow the notation defined in YANG Tree Diagrams [RFC8340].

```
module: ietf-twamp
    +--rw twamp
       +--rw client {control-client}?
       |  +--rw admin-state?             boolean
       |  +--rw mode-preference-chain* [priority]
       |  |  +--rw priority    uint16
       |  |  +--rw mode?       twamp-modes
       |  +--rw key-chain* [key-id]
       |  |  +--rw key-id        string
       |  |  +--rw secret-key?   binary
       |  +--rw ctrl-connection* [name]
       |     +--rw name                    string
       |     +--rw client-ip?              inet:ip-address
       |     +--rw server-ip               inet:ip-address
       |     +--rw server-tcp-port?        inet:port-number
       |     +--rw control-packet-dscp?    inet:dscp
       |     +--rw key-id?                 string
```

```
|      +--rw max-count-exponent?     uint8
|      +--ro client-tcp-port?        inet:port-number
|      +--ro server-start-time?      uint64
|      +--ro repeat-count?           uint64
|      +--ro state?
|      |       control-client-connection-state
|      +--ro selected-mode?          twamp-modes
|      +--ro token?                  binary
|      +--ro client-iv?              binary
|      +--rw test-session-request* [name]
|         +--rw name                  string
|         +--rw sender-ip?            inet:ip-address
|         +--rw sender-udp-port?      union
|         +--rw reflector-ip          inet:ip-address
|         +--rw reflector-udp-port?   inet:port-number
|         +--rw timeout?              uint64
|         +--rw padding-length?       uint32
|         +--rw test-packet-dscp?     inet:dscp
|         +--rw start-time?           uint64
|         +--rw repeat?               uint32
|         +--rw repeat-interval?      uint32
|         +--rw pm-reg-list* [pm-index]
|         |  +--rw pm-index    uint16
|         +--ro state?                test-session-state
|         +--ro sid?                  string
|
+--rw server {server}?
|  +--rw admin-state?           boolean
|  +--rw server-tcp-port?       inet:port-number
|  +--rw servwait?              uint32
|  +--rw control-packet-dscp?   inet:dscp
|  +--rw count?                 uint8
|  +--rw max-count-exponent?    uint8
|  +--rw modes?                 twamp-modes
|  +--rw key-chain* [key-id]
|  |  +--rw key-id        string
|  |  +--rw secret-key?   binary
|  +--ro ctrl-connection*
|           [client-ip client-tcp-port server-ip server-tcp-port]
|     +--ro client-ip              inet:ip-address
|     +--ro client-tcp-port        inet:port-number
|     +--ro server-ip              inet:ip-address
|     +--ro server-tcp-port        inet:port-number
|     +--ro state?                 server-ctrl-connection-state
|     +--ro control-packet-dscp?   inet:dscp
|     +--ro selected-mode?         twamp-modes
|     +--ro key-id?                string
|     +--ro count?                 uint8
|     +--ro max-count-exponent?    uint8
|
```

```
        |      +--ro salt?                  binary
        |      +--ro server-iv?             binary
        |      +--ro challenge?             binary
        +--rw session-sender {session-sender}?
        |  +--rw admin-state?    boolean
        |  +--rw test-session* [name]
        |     +--rw name                    string
        |     +--ro ctrl-connection-name?   string
        |     +--rw fill-mode?              padding-fill-mode
        |     +--rw number-of-packets       uint32
        |     +--rw (packet-distribution)?
        |     |  +--:(periodic)
        |     |  |  +--rw periodic-interval       decimal64
        |     |  +--:(poisson)
        |     |     +--rw lambda                 decimal64
        |     |     +--rw max-interval?          decimal64
        |     +--ro state?                  sender-session-state
        |     +--ro sent-packets?           uint32
        |     +--ro rcv-packets?            uint32
        |     +--ro last-sent-seq?          uint32
        |     +--ro last-rcv-seq?           uint32
        +--rw session-reflector {session-reflector}?
           +--rw admin-state?    boolean
           +--rw refwait?        uint32
           +--ro test-session*
                   [sender-ip sender-udp-port reflector-ip reflector-udp
-port]
              +--ro sid?                                  string
              +--ro sender-ip                             inet:ip-address
              +--ro sender-udp-port
              |       dynamic-port-number
              +--ro reflector-ip                          inet:ip-address
              +--ro reflector-udp-port                    inet:port-numbe
r
              +--ro parent-connection-client-ip?          inet:ip-address
              +--ro parent-connection-client-tcp-port?    inet:port-numbe
r
              +--ro parent-connection-server-ip?          inet:ip-address
              +--ro parent-connection-server-tcp-port?    inet:port-numbe
r
              +--ro test-packet-dscp?                     inet:dscp
              +--ro sent-packets?                         uint32
              +--ro rcv-packets?                          uint32
              +--ro last-sent-seq?                        uint32
              +--ro last-rcv-seq?                         uint32
```

Figure 7: YANG Tree Diagram.

5.2. YANG Module

This section presents the YANG module for the TWAMP data model defined in this document. The module imports definitions from Common YANG Data Types [RFC6991], and references NTPv4 Specification [RFC5905], Framework for IP Performance Metrics [RFC2330], Randomness Requirements for Security [RFC4086], OWAMP [RFC4656], TWAMP [RFC5357], More Features for TWAMP [RFC5618], Individual Session Control Feature [RFC5938], TWAMP Reflect Octets and Symmetrical Size Features [RFC6038], Advances Stream and Sampling Framework [RFC7312], IKEv2-Derived Shared Secret Key for OWAMP and TWAMP [RFC7717], and OWAMP and TWAMP Well-Known Port Assignments [I-D.ietf-ippm-port-twamp-test].

<CODE BEGINS> file "ietf-twamp@2018-07-02.yang"

```
module ietf-twamp {
  yang-version 1.1;
  namespace urn:ietf:params:xml:ns:yang:ietf-twamp;
  prefix ietf-twamp;

  import ietf-inet-types {
    prefix inet;
    reference
      "RFC 6991: Common YANG Types.";
  }

  organization
    "IETF IPPM (IP Performance Metrics) Working Group";

  contact
    "WG Web: http://tools.ietf.org/wg/ippm/
     WG List: ippm@ietf.org

     Editor: Ruth Civil
             gcivil@ciena.com
     Editor: Al Morton
             acmorton@att.com
     Editor: Reshad Rehman
             rrahman@cisco.com
     Editor: Mahesh Jethanandani
             mjethanandani@gmail.com
     Editor: Kostas Pentikousis
             k.pentikousis@travelping.com";


  description
    "This YANG module specifies a vendor-independent data
```

```
        model for the Two-Way Active Measurement Protocol (TWAMP).

        The data model covers four TWAMP logical entities, namely,
        Control-Client, Server, Session-Sender, and Session-Reflector,
        as illustrated in the annotated TWAMP logical model (Fig. 1
        of RFC XXXX).

        This YANG module uses features to indicate which of the four
        logical entities are supported by a TWAMP implementation.

        Copyright (c) 2018 IETF Trust and the persons identified as
        the document authors.  All rights reserved.
        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject
        to the license terms contained in, the Simplified BSD
        License set forth in Section 4.c of the IETF Trust's Legal
        Provisions Relating to IETF Documents
        (http://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX; see
        the RFC itself for full legal notices.";

      revision 2018-07-02 {
        description
          "Initial Revision.

          Covers RFC 5357, RFC 5618, RFC 5938, RFC 6038, RFC 7717, and
          draft-ietf-ippm-metric-registry";

        reference
          "RFC XXXX: TWAMP YANG Data Model.";
      }


      /*
       * Typedefs
       */

      typedef twamp-modes {
        type bits {
          bit unauthenticated {
            position 0;
            description
              "Unauthenticated mode, in which no encryption or
              authentication is applied in TWAMP-Control and
              TWAMP-Test. KeyID, Token, and Client-IV are not used in
              the Set-Up-Response message. See Section 3.1 of
              RFC 4656.";
```

```
          reference
            "RFC 4656: A One-way Active Measurement Protocol
             (OWAMP)";
        }
        bit authenticated {
          position 1;
          description
            "Authenticated mode, in which the Control-Client and
             Server possess a shared secret thus prohibiting
             'theft of service'. As per Section 6 of RFC 4656,
             in 'authenticated mode, the timestamp is in the clear
             and is not protected cryptographically in any way,
             while the rest of the message has the same protection
             as in encrypted mode. This mode allows one to trade off
             cryptographic protection against accuracy of
             timestamps.'";
          reference
            "RFC 4656: A One-way Active Measurement Protocol
             (OWAMP)";
        }
        bit encrypted {
          position 2;
          description
            "Encrypted mode 'makes it impossible to alter
             timestamps undetectably' [Section 6 of RFC 4656].
             See also Section 4 of RFC 7717.";
          reference
            "RFC 4656: A One-way Active Measurement Protocol
             (OWAMP)";
        }
        bit unauth-test-encrpyt-control {
          position 3;
          description
            "When using the Mixed Security Mode, the TWAMP-Test
             protocol follows the Unauthenticated mode and the
             TWAMP-Control protocol the Encrypted mode.";
          reference
            "RFC 5618: Mixed Security Mode for the Two-Way Active
             Measurement Protocol (TWAMP)";
        }
        bit individual-session-control {
          position 4;
          description
            "This mode enables individual test sessions using
             Session Identifiers.";
          reference
            "RFC 5938: Individual Session Control Feature
             for the Two-Way Active Measurement Protocol (TWAMP)";
```

```
        }
        bit reflect-octets {
          position 5;
          description
            "This mode indicates the reflect octets capability.";
          reference
            "RFC 6038: Two-Way Active Measurement Protocol (TWAMP)
             Reflect Octets and Symmetrical Size Features";
        }
        bit symmetrical-size {
          position 6;
          description
            "This mode indicates support for the symmetrical size
             sender test packet format.";
          reference
            "RFC 6038: Two-Way Active Measurement Protocol (TWAMP)
             Reflect Octets and Symmetrical Size Features";
        }
        bit IKEv2Derived {
          position 7;
          description
            "In this mode the the shared key is derived
             from an IKEv2 security association (SA).";
          reference
            "RFC 7717: IKEv2-Derived Shared Secret Key for
             the One-Way Active Measurement Protocol (OWAMP)
             and Two-Way Active Measurement Protocol (TWAMP)";
        }
      }
      description
        "Specifies the configurable TWAMP-Modes supported during a
         TWAMP-Control Connection setup between a Control-Client
         and a Server. Section 7 of RFC 7717 summarizes the
         TWAMP-Modes registry and points to their formal
         specification.";
    }

    typedef control-client-connection-state {
      type enumeration {
        enum active {
          description
            "Indicates an active TWAMP-Control connection to
             Server.";
        }
        enum idle {
          description
            "Indicates an idle TWAMP-Control connection to Server.";
        }
```

```
      }
      description
        "Indicates the Control-Client TWAMP-Control connection
         state.";
    }

    typedef test-session-state {
      type enumeration {
        enum accepted {
          value 0;
          description
            "Indicates an accepted TWAMP-Test session request.";
        }
        enum failed {
          value 1;
          description
            "Indicates a TWAMP-Test session failure due to
             some unspecified reason (catch-all).";
        }
        enum internal-error {
          value 2;
          description
            "Indicates a TWAMP-Test session failure due to
             an internal error.";
        }
        enum not-supported {
          value 3;
          description
            "Indicates a TWAMP-Test session failure because
             some aspect of the TWAMP-Test session request
             is not supported.";
        }
        enum permanent-resource-limit {
          value 4;
          description
            "Indicates a TWAMP-Test session failure due to
             permanent resource limitations.";
        }
        enum temp-resource-limit {
          value 5;
          description
            "Indicates a TWAMP-Test session failure due to
             temporary resource limitations.";
        }
      }
      description
        "Indicates the Control-Client TWAMP-Test session state.";
    }
```

```
    typedef server-ctrl-connection-state {
      type enumeration {
        enum active {
          description
            "Indicates an active TWAMP-Control connection
             to the Control-Client.";
        }
        enum servwait {
          description
            "Indicates that the TWAMP-Control connection to the
             Control-Client is in SERVWAIT as per the definition of
             Section 3.1 of RFC 5357.";
        }
      }
      description
        "Indicates the Server TWAMP-Control connection state.";
    }

    typedef sender-session-state {
      type enumeration {
        enum active {
          description
            "Indicates that the TWAMP-Test session is active.";
        }
        enum failure {
          description
            "Indicates that the TWAMP-Test session has failed.";
        }
      }
      description
        "Indicates the Session-Sender TWAMP-Test session state.";
    }

    typedef padding-fill-mode {
      type enumeration {
        enum zero {
          description
            "TWAMP-Test packets are padded with all zeros.";
        }
        enum random {
          description
            "TWAMP-Test packets are padded with pseudo-random
            numbers.";
        }
      }
      description
        "Indicates what type of packet padding is used in the
         TWAMP-Test packets.";
```

```
  }

  typedef dynamic-port-number {
    type inet:port-number {
      range 49152..65535;
    }
    description "Dynamic range for port numbers.";
  }

  /*
   * Features
   */

  feature control-client {
    description
      "Indicates that the device supports configuration of the
       TWAMP Control-Client logical entity.";
  }

  feature server {
    description
      "Indicates that the device supports configuration of the
       TWAMP Server logical entity.";
  }

  feature session-sender {
    description
      "Indicates that the device supports configuration of the
       TWAMP Session-Sender logical entity.";
  }

  feature session-reflector {
    description
      "Indicates that the device supports configuration of the
       TWAMP Session-Reflector logical entity.";
  }


  /*
   * Reusable node groups
   */

  grouping key-management {
    list key-chain {
      key key-id;
      leaf key-id {
        type string {
          length 1..80;
```

```
          }
          description
            "KeyID used for a TWAMP-Control connection. As per
             Section 3.1 of RFC 4656, KeyID is 'a UTF-8 string, up to
             80 octets in length' and is used to select which 'shared
             shared secret the [Control-Client] wishes to use to
             authenticate or encrypt'.";
          }
          leaf secret-key {
            type binary;
            description
              "The secret key corresponding to the KeyID for this
               TWAMP-Control connection.";
          }
          description
            "Relates KeyIDs with their respective secret keys
             in a TWAMP-Control connection.";
      }
      description
          "Used by the Control-Client and Server for TWAMP-Control
           key management.";
    }

    grouping maintenance-statistics {
      leaf sent-packets {
        type uint32;
        config false;
        description
          "Indicates the number of packets sent.";
      }

      leaf rcv-packets {
        type uint32;
        config false;
        description
          "Indicates the number of packets received.";
      }

      leaf last-sent-seq {
        type uint32;
        config false;
        description
          "Indicates the last sent sequence number.";
      }

      leaf last-rcv-seq {
        type uint32;
        config false;
```

```
        description
          "Indicates the last received sequence number.";
      }
      description
        "Used for TWAMP-Test maintenance statistics.";
    }

    grouping count {
      leaf count {
        type uint8 {
          range "10..31";
        }
        default 15;
        description
          "Parameter communicated to the Control-Client as part of
           the Server Greeting message and used for deriving a key
           from a shared secret as per Section 3.1 of  RFC 4656:
           MUST be a power of 2 and at least 1024. It is configured
           by providing said power. For example, configuring 20 here
           means count 2^20 = 1048576. The default is 15,
           meaning 2^15 = 32768.";
        }
     description
       "Reusable data structure for count, which is used both in the
        Server and the Control-Client.";
    }

    grouping max-count-exponent {
      leaf max-count-exponent {
        type uint8 {
          range 10..31;
        }
        default 20;
        description
          "This parameter limits the maximum Count value, which MUST
           be a power of 2 and at least 1024 as per RFC 5357. It is
           configured by providing said power. For example,
           configuring 10 here means max count 2^10 = 1024.
           The default is 20, meaning 2^20 = 1048576.

           A TWAMP Server uses this configured value in the
           Server-Greeting message sent to the Control-Client.

           A TWAMP Control-Client uses this configured value to
           prevent denial-of-service (DOS) attacks by closing the
           control connection to the Server if it 'receives a
           Server-Greeting message with Count greater that its
           maximum configured value', as per Section 6 of RFC 5357.
```

```
            Further, note that according to Section 6 of RFC 5357:

            'If an attacking system sets the maximum value in
            Count (2**32), then the system under attack would stall
            for a significant period of time while it attempts to
            generate keys.

            TWAMP-compliant systems SHOULD have a configuration
            control to limit the maximum count value. The default
            max-count-exponent value SHOULD be 15 which corresponds
            to a maximum value of 2**15 or 32768.'

            RFC 5357 does not qualify 'significant period' in terms of
            time, but it is clear that this depends on the processing
            capacity available and operators need to pay attention to
            this security consideration.";
        }
      description
        "Reusable data structure for max-count which is used both at
         the Control-Client and the Server containers.";
    }


    /*
     * Configuration data nodes
     */

    container twamp {
      description
        "TWAMP logical entity configuration grouping of four models
        which correspond to the four TWAMP logical entities
        Control-Client, Server, Session-Sender, and Session-Reflector
        as illustrated in Fig. 1 of RFC XXXX.";

      container client {
        if-feature control-client;
        description
          "Configuration of the TWAMP Control-Client logical
           entity.";

        leaf admin-state {
          type boolean;
          default true;
          description
            "Indicates whether the device is allowed to operate as a
             TWAMP Control-Client.";
        }
```

```
        list mode-preference-chain {
          key priority;
          unique mode;
          leaf priority {
            type uint16;
            description
              "Indicates the Control-Client Mode preference priority
               expressed as a 16-bit unsigned integer. Values for the
               priority start with zero, the highest priority, and
               decreasing priority value is indicated by every increase
               in value by one.";
          }
          leaf mode {
            type twamp-modes;
            description
              "The supported TWAMP Mode matching the corresponding
               priority.";

          }
          description
            "Indicates the Control-Client preferred order of use of
             the supported TWAMP Modes.

             Depending on the Modes available in the TWAMP Server
             Greeting message (see Fig. 2 of RFC 7717), the
             Control-Client MUST choose the highest priority
             Mode from the configured mode-preference-chain list.";
        }

        uses key-management;

        list ctrl-connection {
          key name;
          description
            "List of TWAMP Control-Client control connections.
             Each item in the list describes a control connection
             that will be initiated by this Control-Client";

          leaf name {
            type string;
            description
              "A unique name used as a key to identify this
               individual TWAMP-Control connection on the
               Control-Client device.";
          }
          leaf client-ip {
            type inet:ip-address;
            description
```

```
            "The IP address of the local Control-Client device,
             to be placed in the source IP address field of the
             IP header in TWAMP-Control (TCP) packets belonging
             to this control connection. If not configured, the
             device SHALL choose its own source IP address.";
          }
          leaf server-ip {
            type inet:ip-address;
            mandatory true;
            description
              "The IP address of the remote Server device, which the
               TWAMP-Control connection will be initiated to.";
          }

          leaf server-tcp-port {
            type inet:port-number;
            default 862;
            description
              "This parameter defines the TCP port number that is
               to be used by this outgoing TWAMP-Control connection.
               Typically, this is the well-known TWAMP-Control
               port number (862) as per RFC 5357 However, there are
               known realizations of TWAMP in the field that were
               implemented before this well-known port number was
               allocated. These early implementations allowed the
               port number to be configured. This parameter is
               therefore provided for backward compatibility
               reasons.";
          }

          leaf control-packet-dscp {
            type inet:dscp;
            default 0;
            description
              "The DSCP value to be placed in the IP header of
               TWAMP-Control (TCP) packets generated by this
               Control-Client.";
          }

          leaf key-id {
            type string {
              length 1..80;
            }
            description
             "Indicates the KeyID value selected for this
              TWAMP-Control connection.";
          }
```

```
          uses max-count-exponent;

          leaf client-tcp-port {
            type inet:port-number;
            config false;
            description
              "Indicates the source TCP port number used in the
               TWAMP-Control packets belonging to this control
               connection.";
          }

          leaf server-start-time {
            type uint64;
            config false;
            description
              "Indicates the Start-Time advertised by the Server in
               the Server-Start message (RFC 4656, Section 3.1),
               representing the time when the current
               instantiation of the Server started operating.
               The timestamp format follows RFC 5905
               according to Section 4.1.2 of RFC 4656.";
            reference
              "RFC 4656: OWAMP, Section 3.1 and 4.1.2,
               RFC 5905: NTPv4 Specification.";
          }

          leaf repeat-count {
            type uint64;
            config false;
            description
              "Indicates how many times the test session has been
               repeated. When a test is running, this value will be
               greater than 0. If the repeat parameter is non-zero,
               this value is smaller than or equal to the repeat
               parameter.";
          }
          leaf state {
            type control-client-connection-state;
            config false;
            description
              "Indicates the current state of the TWAMP-Control
               connection state.";
          }

          leaf selected-mode {
            type twamp-modes;
            config false;
            description
```

```
            "The TWAMP Mode that the Control-Client has chosen for
             this control connection as set in the Mode field of
             the Set-Up-Response message";
          reference
            "RFC 4656, Section 3.1.";
        }

        leaf token {
            type binary {
            length 64;
          }
          config false;
          description
            "This parameter holds the 64 octets containing 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; see
             also the last paragraph of Section 6 in RFC 4656.

             If the Mode defined in RFC 7717 is selected
             (selected-mode), Token is limited to 16 octets.";
          reference
            "RFC 4086: Randomness Requirements for Security

             RFC 7717: IKEv2-Derived Shared Secret Key for the
             One-Way Active Measurement Protocol (OWAMP) and
             Two-Way Active Measurement Protocol (TWAMP)";
        }

        leaf client-iv {
          type binary {
            length 16;
          }
          config false;
          description
            "Indicates the Control-Client Initialization Vector
             (Client-IV), that is generated randomly by the
             Control-Client. As per RFC 4656:

              Client-IV merely needs to be unique (i.e., it MUST
              never be repeated for different sessions using the
              same secret key; a simple way to achieve that without
              the use of cumbersome state is to generate the
              Client-IV values using a cryptographically secure
              pseudo-random number source.

              If the Mode defined in RFC 7717 is selected
              (selected-mode), Client-IV is limited to 12 octets.";
```

```
            reference
              "RFC 4656: A One-way Active Measurement Protocol
               (OWAMP).

               RFC 7717: IKEv2-Derived Shared Secret Key for the
               One-Way Active Measurement Protocol (OWAMP) and
               Two-Way Active Measurement Protocol (TWAMP)";
          }

          list test-session-request {
            key name;
            description
              "Information associated with the Control-Client
               for this test session";

            leaf name {
              type string;
              description
                "A unique name to be used for identification of
                 this TWAMP-Test session on the Control-Client.";
            }

            leaf sender-ip {
              type inet:ip-address;
              description
                "The IP address of the Session-Sender device,
                 which is to be placed in the source IP address
                 field of the IP header in TWAMP-Test (UDP) packets
                 belonging to this test session. This value will be
                 used to populate the sender address field of the
                 Request-TW-Session message.

                 If not configured, the device SHALL choose its own
                 source IP address.";
            }

            leaf sender-udp-port {
              type union {
                type dynamic-port-number;
                type enumeration {
                  enum autoallocate {
                    description
                      "Indicates that the Contol-Client will
                       auto-allocate the TWAMP-Test (UDP) port number
                       from the dynamic port range.";
                  }
                }
              }
```

```
        default autoallocate;
        description
          "The UDP port number that is to be used by
           the Session-Sender for this TWAMP-Test session.
           The number is restricted to the dynamic port range.

           By default the Control-Client SHALL auto-allocate a
           UDP port number for this TWAMP-Test session.

           The configured (or auto-allocated) value is
           advertised in the Sender Port field of the
           Request-TW-session message (see Section 3.5 of
           RFC 5357). Note that in the scenario where a device
           auto-allocates a UDP port number for a session, and
           the repeat parameter for that session indicates that
           it should be repeated, the device is free to
           auto-allocate a different UDP port number when it
           negotiates the next (repeated) iteration of this
           session.";
      }

      leaf reflector-ip {
        type inet:ip-address;
        mandatory true;
        description
          "The IP address belonging to the remote
           Session-Reflector device to which the TWAMP-Test
           session will be initiated. This value will be
           used to populate the receiver address field of
           the Request-TW-Session message.";
      }

      leaf reflector-udp-port {
        type inet:port-number {
          range "862 | 49152..65535";
        }
        description
          "This parameter defines the UDP port number that
           will be used by the Session-Reflector for
           this TWAMP-Test session. The default number is
           within the dynamic port range and is to be placed
           in the Receiver Port field of the Request-TW-Session
           message. The well-known port (862) MAY be
           used.";
        reference
          "draft-ietf-ippm-port-twamp-test: OWAMP and TWAMP
           Well-Known Port Assignments.";
      }
```

```
            leaf timeout {
              type uint64;
              units seconds;
              default 2;
              description
                "The length of time (in seconds) that the
                 Session-Reflector should continue to respond to
                 packets belonging to this TWAMP-Test session after
                 a Stop-Sessions TWAMP-Control message has been
                 received.

                 This value will be placed in the Timeout field of
                 the Request-TW-Session message.";
              reference
                "RFC 5357: TWAMP, Section 3.5.";
            }

            leaf padding-length {
              type uint32 {
                range 64..4096;
              }
              description
                  "The number of padding bytes to be added to the
                   TWAMP-Test (UDP) packets generated by the
                   Session-Sender.

                   This value will be placed in the Padding Length
                   field of the Request-TW-Session message.";
              reference
                "RFC 4656, Section 3.5.";
            }

            leaf test-packet-dscp {
              type inet:dscp;
              default 0;
              description
                "The DSCP value to be placed in the IP header
                 of TWAMP-Test packets generated by the
                 Session-Sender, and in the UDP header of the
                 TWAMP-Test response packets generated by the
                 Session-Reflector for this test session.

                 This value will be placed in the Type-P Descriptor
                 field of the Request-TW-Session message";
              reference
                "RFC 5357.";
            }
```

```
            leaf start-time {
              type uint64;
              default 0;
              description
                "Time when the session is to be started
                 (but not before the TWAMP Start-Sessions command
                 is issued; see Section 3.4 of RFC 5357).

                 The start-time value is placed in the Start Time
                 field of the Request-TW-Session message.

                 The timestamp format follows RFC 5905 as per
                 Section 3.5 of RFC 4656.

                 The default value of 0 indicates that the session
                 will be started as soon as the Start-Sessions
                 message is received.";
            }

            leaf repeat {
              type uint32 {
                range 0..4294967295;
              }
              default 0;
              description
                "This value determines if the TWAMP-Test session must
                 be repeated. When a test session has completed, the
                 repeat parameter is checked.

                 The default value of 0 indicates that the session
                 MUST NOT be repeated.

                 If the repeat value is 1 through 4,294,967,294
                 then the test session SHALL be repeated using the
                 information in repeat-interval parameter, and the
                 parent TWAMP-Control connection for this test
                 session is restarted to negotiate a new instance
                 of this TWAMP-Test session.

                 A value of 4,294,967,295 indicates that the test
                 session SHALL be repeated *forever* using the
                 information in repeat-interval parameter, and SHALL
                 NOT decrement the value.";
            }

            leaf repeat-interval  {
              when "../repeat!='0'" {
                description
```

```
                "This parameter determines the timing of repeated
                 TWAMP-Test sessions when repeat is more than 0.

                 When the value of repeat-interval is 0, the
                 negotiation of a new test session SHALL begin
                 immediately after the previous test session
                 completes. Otherwise, the Control-Client will
                 wait for the number of seconds specified in the
                 repeat-interval parameter before negotiating the
                 new instance of this TWAMP-Test session.";
            }
            type uint32;
            units seconds;
            default 0;
            description
              "Repeat interval (in seconds).";
          }

          list pm-reg-list {
            key pm-index;
            leaf pm-index {
              type uint16;
              description
                "Numerical index value of a Registered Metric
                 in the Performance Metric Registry
                 (see ietf-ippm-metric-registry). Output statistics
                 are specified in the corresponding Registry
                 entry.";
            }
            description
              "A list of one or more Performance Metric Registry
               Index values, which communicate packet stream
               characteristics along with one or more metrics
               to be measured.

               All members of the pm-reg-list MUST have the same
               stream characteristics, such that they combine
               to specify all metrics that shall be measured on
               a single stream.";
            reference
              "ietf-ippm-metric-registry: Registry for
               Performance Metrics";
          }

          leaf state {
            type test-session-state;
            config false;
            description
```

```
                "Indicates the TWAMP-Test session state, accepted or
                 indication of an error.";
              reference
                "Section 3.5 of RFC 5357.";
            }
            leaf sid {
              type string;
              config false;
              description
                "The SID allocated by the Server for this TWAMP-Test
                 session, and communicated back to the Control-Client
                 in the SID field of the Accept-Session message";
              reference
                "Section 4.3 of RFC 6038.";
            }
          }
        }
      }

      container server {
        if-feature server;
        description
          "Configuration of the TWAMP Server logical entity.";

        leaf admin-state {
          type boolean;
          default true;
          description
            "Indicates whether the device is allowed to operate
             as a TWAMP Server.";
        }

        leaf server-tcp-port {
          type inet:port-number;
          default 862;
          description
            "This parameter defines the well known TCP port number
             that is used by TWAMP-Control. The Server will listen
             on this port number for incoming TWAMP-Control
             connections. Although this is defined as a fixed value
             (862) in RFC 5357, there are several realizations of
             TWAMP in the field that were implemented before this
             well-known port number was allocated. These early
             implementations allowed the port number to be
             configured. This parameter is therefore provided for
             backward compatibility reasons.";
        }
```

```
        leaf servwait {
          type uint32 {
            range 1..604800;
          }
          units seconds;
          default 900;
          description
            "TWAMP-Control (TCP) session timeout, in seconds.
             According to Section 3.1 of RFC 5357,

             Server MAY discontinue any established control
             connection when no packet associated with that
             connection has been received within SERVWAIT seconds.";
        }

        leaf control-packet-dscp {
          type inet:dscp;
          description
            "The DSCP value to be placed in the IP header of
             TWAMP-Control (TCP) packets generated by the Server.

             Section 3.1 of  RFC 5357 specifies that the server
             SHOULD use the DSCP value from the Control-Clients
             TCP SYN. However, for practical purposes TWAMP will
             typically be implemented using a general purpose TCP
             stack provided by the underlying operating system,
             and such a stack may not provide this information to the
             user. Consequently, it is not always possible to
             implement the behavior described in RFC 5357 in an
             OS-portable version of TWAMP.

             The default behavior if this item is not set is to use
             the DSCP value from the Control-Clients TCP SYN.";
           reference
             "Section 3.1 of RFC 5357.";
        }

        uses count;

        uses max-count-exponent;

        leaf modes {
          type twamp-modes;
          description
            "The bit mask of TWAMP Modes this Server instance
             is willing to support; see IANA TWAMP Modes Registry.";
        }
```

```
        uses key-management;

        list ctrl-connection {
          key "client-ip client-tcp-port server-ip server-tcp-port";
          config false;
          description
            "List of all incoming TWAMP-Control (TCP) connections.";

          leaf client-ip {
            type inet:ip-address;
            description
              "The IP address on the remote Control-Client device,
               which is the source IP address used in the
               TWAMP-Control (TCP) packets belonging to this control
               connection.";
          }

          leaf client-tcp-port {
            type inet:port-number;
            description
              "The source TCP port number used in the TWAMP-Control
               (TCP) packets belonging to this control connection.";
          }

          leaf server-ip {
            type inet:ip-address;
            description
              "The IP address of the local Server device, which is
               the destination IP address used in the
               TWAMP-Control (TCP) packets belonging to this control
               connection.";
          }

          leaf server-tcp-port {
            type inet:port-number;
            description
              "The destination TCP port number used in the
               TWAMP-Control (TCP) packets belonging to this
               control connection. This will usually be the
               same value as the server-tcp-port configured
               under twamp/server. However, in the event that
               the user re-configured server/server-tcp-port
               after this control connection was initiated, this
               value will indicate the server-tcp-port that is
               actually in use for this control connection.";
          }

          leaf state {
```

```
            type server-ctrl-connection-state;
            description
              "Indicates the Server TWAMP-Control connection state.";
          }

          leaf control-packet-dscp {
            type inet:dscp;
            description
              "The DSCP value used in the IP header of the
               TWAMP-Control (TCP) packets sent by the Server
               for this control connection. This will usually
               be the same value as is configured in the
               control-packet-dscp parameter under the twamp/server
               container.  However, in the event that the user
               re-configures server/dscp after this control
               connection is already in progress, this read-only
               value will show the actual dscp value in use by this
               TWAMP-Control connection.";
          }

          leaf selected-mode {
            type twamp-modes;
            description
              "The Mode that was chosen for this TWAMP-Control
               connection as set in the Mode field of the
               Set-Up-Response message.";
          }

          leaf key-id {
            type string {
              length 1..80;
            }
            description
              "The KeyID value that is in use by this TWAMP-Control
               connection as selected by Control-Client.";
          }

          uses count {
            description
              "The count value that is in use by this TWAMP-Control
               connection. This will usually be the same value
               as is configured under twamp/server. However, in the
               event that the user re-configured server/count
               after this control connection is already in progress,
               this read-only value will show the actual count that
               is in use for this TWAMP-Control connection.";
          }
```

```
          uses max-count-exponent {
            description
              "This read-only value indicates the actual max-count in
               use for this control connection. Usually this would be
               the same value as configured under twamp/server.";
          }

          leaf salt {
            type binary {
              length 16;
            }
            description
              "A parameter used in deriving a key from a
               shared secret as described in Section 3.1 of RFC 4656.
               It is communicated to the Control-Client as part of
               the Server Greeting message.";
          }

          leaf server-iv {
            type binary {
              length 16;
            }
            description
              "The Server Initialization Vector
               (IV) generated randomly by the Server.";
          }

          leaf challenge {
            type binary {
              length 16;
            }
            description
              "A random sequence of octets generated by the Server.
               As described in client/token, Challenge is used
               by the Control-Client to prove possession of a
               shared secret.";
          }
        }
      }

      container session-sender {
        if-feature session-sender;
        description
          "Configuration of the TWAMP Session-Sender logical entity";
        leaf admin-state {
          type boolean;
          default true;
          description
```

```
            "Indicates whether the device is allowed to operate
             as a TWAMP Session-Sender.";
          }

          list test-session{
            key name;
            description
              "List of TWAMP Session-Sender test sessions.";

            leaf name {
              type string;
              description
                "A unique name for this TWAMP-Test session to be used
                 for identifying this test session by the
                 Session-Sender logical entity.";
            }

            leaf ctrl-connection-name {
              type string;
              config false;
              description
                "The name of the parent TWAMP-Control connection that
                 is responsible for negotiating this TWAMP-Test
                 session.";
            }

            leaf fill-mode {
              type padding-fill-mode;
              default zero;
              description
                "Indicates whether the padding added to the
                 TWAMP-Test (UDP) packets will contain pseudo-random
                 numbers, or whether it should consist of all zeroes,
                 as per Section 4.2.1 of RFC 5357.";
            }

            leaf number-of-packets {
              type uint32;
              mandatory true;
              description
                "The overall number of TWAMP-Test (UDP) packets to be
                 transmitted by the Session-Sender for this test
                 session.";
            }

            choice packet-distribution {
              description
                "Indicates the distribution to be used for transmitting
```

```
               the TWAMP-Test (UDP) packets.";
             case periodic {
               leaf periodic-interval {
                 type decimal64 {
                   fraction-digits 5;
                 }
                 units seconds;
                 mandatory true;
                 description
                   "Indicates the time to wait (in seconds) between
                    the first bits of TWAMP-Test (UDP) packet
                    transmissions for this test session.";
                 reference
                   "RFC 3432: Network performance measurement
                              with periodic streams";
               }
             }
             case poisson {
               leaf lambda {
                 type decimal64 {
                   fraction-digits 5;
                 }
                 units seconds;
                 mandatory true;
                 description
                   "Indicates the average time interval (in seconds)
                    between packets in the Poisson distribution.
                    The packet is calculated using the reciprocal of
                    lambda and the TWAMP-Test packet size (which
                    depends on the selected Mode and the packet
                    padding).";
                reference
                  "RFC 2330: Framework for IP Performance Metrics";
               }
               leaf max-interval {
                 type decimal64 {
                   fraction-digits 5;
                 }
                 units seconds;
                 description
                   "Indicates the maximum time (in seconds)
                    between packet transmissions.";
                 reference
                   "RFC 7312: Advanced Stream and Sampling Framework
                              for IP Performance Metrics (IPPM)";
               }
             }
           }
```

```
          leaf state {
            type sender-session-state;
            config false;
            description
              "Indicates the Session-Sender test session state.";
          }

          uses maintenance-statistics;
        }
      }

      container session-reflector {
        if-feature session-reflector;
        description
          "Configuration of the TWAMP Session-Reflector logical
           entity";

        leaf admin-state {
          type boolean;
          default true;
          description
            "Indicates whether the device is allowed to operate
             as a TWAMP Session-Reflector.";
        }

        leaf refwait {
          type uint32 {
            range 1..604800;
          }
          units seconds;
          default 900;
          description
            "The Session-Reflector MAY discontinue any session that
             has been started when no packet associated with that
             session has been received for REFWAIT seconds. As per
             Section 3.1 of RFC 5357, this timeout allows a
             Session-Reflector to free up resources in case of
             failure.";
        }

        list test-session {
          key
            "sender-ip sender-udp-port
             reflector-ip reflector-udp-port";
          config false;
          description
            "TWAMP Session-Reflectortest sessions.";
```

```
          leaf sid {
            type string;
            description
              "An auto-allocated identifier for this TWAMP-Test
               session that is unique within the context of this
               Server/Session-Reflector device only. This value
               is communicated to the Control-Client that
               requested the test session in the SID field of the
               Accept-Session message.";
          }

          leaf sender-ip {
            type inet:ip-address;
            description
              "The IP address on the remote device, which is the
               source IP address used in the TWAMP-Test (UDP) packets
               belonging to this test session.";
          }

          leaf sender-udp-port {
            type dynamic-port-number;
            description
              "The source UDP port used in the TWAMP-Test packets
               belonging to this test session.";
          }

          leaf reflector-ip {
            type inet:ip-address;
            description
              "The IP address of the local Session-Reflector
               device, which is the destination IP address used
               in the TWAMP-Test (UDP) packets belonging to this test
               session.";
          }

          leaf reflector-udp-port {
            type inet:port-number {
              range "862 | 49152..65535";
              }
            description
              "The destination UDP port number used in the
               TWAMP-Test (UDP) test packets belonging to this
               test session.";
          }

          leaf parent-connection-client-ip {
            type inet:ip-address;
            description
```

```
              "The IP address on the Control-Client device, which
               is the source IP address used in the TWAMP-Control
               (TCP) packets belonging to the parent control
               connection that negotiated this test session.";
            }

            leaf parent-connection-client-tcp-port {
              type inet:port-number;
              description
                "The source TCP port number used in the TWAMP-Control
                 (TCP) packets belonging to the parent control
                 connection that negotiated this test session.";
            }

            leaf parent-connection-server-ip {
              type inet:ip-address;
              description
                "The IP address of the Server device, which is the
                 destination IP address used in the TWAMP-Control
                 (TCP) packets belonging to the parent control
                 connection that negotiated this test session.";
            }

            leaf parent-connection-server-tcp-port {
              type inet:port-number;
              description
                "The destination TCP port number used in the
                 TWAMP-Control (TCP) packets belonging to the parent
                 control connection that negotiated this test
                 session.";
            }

            leaf test-packet-dscp {
              type inet:dscp;
              description
                "The DSCP value present in the IP header of
                 TWAMP-Test (UDP) packets belonging to this session.";
            }

            uses maintenance-statistics;
          }
        }
      }
    }

    <CODE ENDS>
```

6. Data Model Examples

This section presents a simple but complete example of configuring all four entities in Figure 1, based on the YANG module specified in Section 5. The example is illustrative in nature, but aims to be self-contained, i.e. were it to be executed in a real TWAMP implementation it would lead to a correctly configured test session. For completeness, examples are provided for both IPv4 and IPv6.

A more elaborated example, which also includes authentication parameters, is provided in Appendix A.

6.1. Control-Client

Figure 8 shows a configuration example for a Control-Client with client/admin-state enabled. In a real implementation following Figure 2 this would permit the initiation of TWAMP-Control connections and TWAMP-Test sessions.

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <client>
      <admin-state>true</admin-state>
    </client>
  </twamp>
</config>
```

Figure 8: XML instance enabling Control-Client operation.

The following example shows a Control-Client with two instances of client/ctrl-connection, one called "RouterA" and another called "RouterB". Each TWAMP-Control connection is to a different Server. The control connection named "RouterA" has two test session requests. The TWAMP-Control connection named "RouterB" has no TWAMP-Test session requests.

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <client>
      <admin-state>true</admin-state>
      <ctrl-connection>
        <name>RouterA</name>
        <client-ip>203.0.113.1</client-ip>
        <server-ip>203.0.113.2</server-ip>
        <test-session-request>
```

```
            <name>Test1</name>
            <sender-ip>203.0.113.3</sender-ip>
            <sender-udp-port>54001</sender-udp-port>
            <reflector-ip>203.0.113.4</reflector-ip>
            <reflector-udp-port>50001</reflector-udp-port>
            <start-time>0</start-time>
          </test-session-request>
          <test-session-request>
            <name>Test2</name>
            <sender-ip>203.0.113.1</sender-ip>
            <sender-udp-port>54001</sender-udp-port>
            <reflector-ip>203.0.113.2</reflector-ip>
            <reflector-udp-port>50001</reflector-udp-port>
            <start-time>0</start-time>
          </test-session-request>
        </ctrl-connection>
        <ctrl-connection>
          <name>RouterB</name>
          <client-ip>203.0.113.1</client-ip>
          <server-ip>203.0.113.3</server-ip>
        </ctrl-connection>
      </client>
    </twamp>
  </config>

  <?xml version="1.0" encoding="utf-8"?>
  <config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
      <client>
        <admin-state>true</admin-state>
        <ctrl-connection>
          <name>RouterA</name>
          <client-ip>2001:DB8:203:0:113::1</client-ip>
          <server-ip>2001:DB8:203:0:113::2</server-ip>
          <test-session-request>
            <name>Test1</name>
            <sender-ip>2001:DB8:203:1:113::3</sender-ip>
            <sender-udp-port>54000</sender-udp-port>
            <reflector-ip>2001:DB8:203:1:113::4</reflector-ip>
            <reflector-udp-port>55000</reflector-udp-port>
            <start-time>0</start-time>
          </test-session-request>
          <test-session-request>
            <name>Test2</name>
            <sender-ip>2001:DB8:203:0:113::1</sender-ip>
            <sender-udp-port>54001</sender-udp-port>
            <reflector-ip>2001:DB8:203:0:113::2</reflector-ip>
            <reflector-udp-port>55001</reflector-udp-port>
```

```
            <start-time>0</start-time>
          </test-session-request>
        </ctrl-connection>
        <ctrl-connection>
          <name>RouterB</name>
          <client-ip>2001:DB8:203:0:113::1</client-ip>
          <server-ip>2001:DB8:203:0:113::3</server-ip>
        </ctrl-connection>
      </client>
    </twamp>
  </config>
```

6.2. Server

Figure 9 shows a configuration example for a Server with server/admin-state enabled, which permits a device following Figure 2 to respond to TWAMP-Control connections and TWAMP-Test sessions.

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <server>
      <admin-state>true</admin-state>
    </server>
  </twamp>
</config>
```

Figure 9: XML instance enabling Server operation.

The following example presents a Server with the TWAMP-Control connection corresponding to the control connection name (client/ctrl-connection/name) "RouterA" presented in Section 6.1.

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <server>
      <admin-state>true</admin-state>
      <ctrl-connection>
        <client-ip>203.0.113.1</client-ip>
        <client-tcp-port>16341</client-tcp-port>
        <server-ip>203.0.113.2</server-ip>
        <server-tcp-port>862</server-tcp-port>
        <state>active</state>
      </ctrl-connection>
    </server>
  </twamp>
</data>

<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <server>
      <admin-state>true</admin-state>
      <ctrl-connection>
        <client-ip>2001:DB8:203:0:113::1</client-ip>
        <client-tcp-port>16341</client-tcp-port>
        <server-ip>2001:DB8:203:0:113::2</server-ip>
        <server-tcp-port>862</server-tcp-port>
        <state>active</state>
      </ctrl-connection>
    </server>
  </twamp>
</data>
```

6.3. Session-Sender

Figure 10 shows a configuration example for a Session-Sender with session-sender/admin-state enabled, which permits a device following Figure 2 to initiate TWAMP-Test sessions.

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <session-sender>
      <admin-state>true</admin-state>
    </session-sender>
  </twamp>
</config>
```

Figure 10: XML instance enabling Session-Sender operation.

The following configuration example shows a Session-Sender with the two TWAMP-Test sessions presented in Section 6.1.

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <session-sender>
      <admin-state>true</admin-state>
      <test-session>
        <name>Test1</name>
        <ctrl-connection-name>RouterA</ctrl-connection-name>
        <number-of-packets>900</number-of-packets>
        <periodic-interval>1</periodic-interval>
      </test-session>
      <test-session>
        <name>Test2</name>
        <ctrl-connection-name>RouterA</ctrl-connection-name>
        <number-of-packets>900</number-of-packets>
        <lambda>1</lambda>
        <max-interval>2</max-interval>
      </test-session>
    </session-sender>
  </twamp>
</data>
```

6.4. Session-Reflector

This configuration example shows a Session-Reflector with session-reflector/admin-state enabled, which permits a device following Figure 2 to respond to TWAMP-Test sessions.

```
<?xml version="1.0" encoding="utf-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <session-reflector>
      <admin-state>true</admin-state>
    </session-reflector>
  </twamp>
</config>
```

Figure 11: XML instance enabling Session-Reflector operation.

The following example shows the two Session-Reflector TWAMP-Test sessions corresponding to the test sessions presented in Section 6.3.

[note: '\' line wrapping is for formatting only]

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <session-reflector>
      <admin-state>true</admin-state>
      <test-session>
        <sender-ip>203.0.113.3</sender-ip>
        <sender-udp-port>54000</sender-udp-port>
        <reflector-ip>203.0.113.4</reflector-ip>
        <reflector-udp-port>50001</reflector-udp-port>
        <sid>1232</sid>
        <parent-connection-client-ip>203.0.113.1</parent-connection-\
client-ip>
        <parent-connection-client-tcp-port>16341</parent-connection-\
client-tcp-port>
        <parent-connection-server-ip>203.0.113.2</parent-connection-\
server-ip>
        <parent-connection-server-tcp-port>862</parent-connection-se\
rver-tcp-port>
        <sent-packets>2</sent-packets>
        <rcv-packets>2</rcv-packets>
        <last-sent-seq>1</last-sent-seq>
        <last-rcv-seq>1</last-rcv-seq>
      </test-session>
      <test-session>
        <sender-ip>203.0.113.1</sender-ip>
        <sender-udp-port>54001</sender-udp-port>
        <reflector-ip>192.0.2.2</reflector-ip>
        <reflector-udp-port>50001</reflector-udp-port>
        <sid>178943</sid>
        <parent-connection-client-ip>203.0.113.1</parent-connection-\
```

```
   client-ip>
           <parent-connection-client-tcp-port>16341</parent-connection-\
   client-tcp-port>
           <parent-connection-server-ip>203.0.113.2</parent-connection-\
   server-ip>
           <parent-connection-server-tcp-port>862</parent-connection-se\
   rver-tcp-port>
           <sent-packets>21</sent-packets>
           <rcv-packets>21</rcv-packets>
           <last-sent-seq>20</last-sent-seq>
           <last-rcv-seq>20</last-rcv-seq>
         </test-session>
       </session-reflector>
     </twamp>
   </data>

   [note: '\' line wrapping is for formatting only]

   <?xml version="1.0" encoding="utf-8"?>
   <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
     <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
       <session-reflector>
         <admin-state>true</admin-state>
         <test-session>
           <sender-ip>203.0.113.3</sender-ip>
           <sender-udp-port>54000</sender-udp-port>
           <reflector-ip>203.0.113.4</reflector-ip>
           <reflector-udp-port>54001</reflector-udp-port>
           <sid>1232</sid>
           <parent-connection-client-ip>203.0.113.1</parent-connection-\
   client-ip>
           <parent-connection-client-tcp-port>16341</parent-connection-\
   client-tcp-port>
           <parent-connection-server-ip>203.0.113.2</parent-connection-\
   server-ip>
           <parent-connection-server-tcp-port>862</parent-connection-se\
   rver-tcp-port>
           <sent-packets>2</sent-packets>
           <rcv-packets>2</rcv-packets>
           <last-sent-seq>1</last-sent-seq>
           <last-rcv-seq>1</last-rcv-seq>
         </test-session>
         <test-session>
           <sender-ip>203.0.113.1</sender-ip>
           <sender-udp-port>54001</sender-udp-port>
           <reflector-ip>192.0.2.2</reflector-ip>
           <reflector-udp-port>55001</reflector-udp-port>
           <sid>178943</sid>
```

```
        <parent-connection-client-ip>203.0.113.1</parent-connection-\
client-ip>
        <parent-connection-client-tcp-port>16341</parent-connection-\
client-tcp-port>
        <parent-connection-server-ip>203.0.113.2</parent-connection-\
server-ip>
        <parent-connection-server-tcp-port>862</parent-connection-se\
rver-tcp-port>
        <sent-packets>21</sent-packets>
        <rcv-packets>21</rcv-packets>
        <last-sent-seq>20</last-sent-seq>
        <last-rcv-seq>20</last-rcv-seq>
      </test-session>
    </session-reflector>
  </twamp>
</data>
```

7. Security Considerations

Virtually all existing measurement systems using TWAMP [RFC5357] are administered by the same network operator. Attacks on the measurement infrastructure could be launched by third-parties to commandeer the packet generation capability, corrupt the measurements, or other examples of nefarious acts.

The YANG module specified in Section 5 of this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF [RFC6241] layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC5246].

The NETCONF Access Control Module (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of nodes defined in this YANG module which are writeable. These data nodes may be considered sensitive and vulnerable to attacks in some network environments. Ability to write into these nodes without proper protection can have a negative effect on the devices that support this feature.

If written, the 'admin-state' node can cause unintended test sessions to be created. If the node 'number-of-packets' that dictates how many packets are sent in any particular test session is written with

a large value, it can cause a test session to run longer than expected. Nodes that are particularly vulnerable include several timeout values put in the protocol to protect against sessions that are not active but are consuming resources. These are the REFWAIT timeout parameter which determine whether to discontinue the session if no packets are received, and nodes 'count' and 'max-count-exponent' which can cause a long time to be spent on PBKDF2 iterations. In addition, 'dscp' node marked with different DSCP markings, can cause the test traffic on the network to be skewed, and the result manipulated. Finally, nodes within 'mode-preference-chain' which specify the 'mode' and 'priority' values and indicate the preferred order of use by an operator, can be manipulated to send unauthenticated or non-encrypted traffic, enabling a MITM attack. Limiting access to these nodes will limit the ability to launch an attack in network environments.

The 'token' node defined in the model, containing a concatenation of a Challenge, AES Session-key used for encryption, and HMAC-SHA1 Session-key used for authentication, is sensitive from a privacy perspective, and can be used to disrupt a test session. The ability to read the field should be limited to the administrator of the test network.

8. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in IETF XML Registry [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-twamp

Registrant Contact: The IESG.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry YANG [RFC6020].

name: ietf-twamp

namespace: urn:ietf:params:xml:ns:yang:ietf-twamp

prefix: twamp

reference: RFC XXXX

9. Acknowledgements

We thank Fred Baker, Kevin D'Souza, Gregory Mirsky, Brian Trammell, Robert Sherman, and Marius Georgescu for their thorough and constructive reviews, comments and text suggestions.

Haoxing Shen contributed to the definition of the YANG module in Section 5.

Jan Lindblad and Ladislav Lhokta did thorough reviews of the YANG module and the examples in Appendix A.

Kostas Pentikousis was partially supported by FP7 UNIFY (http://fp7-unify.eu), a research project partially funded by the European Community under the Seventh Framework Program (grant agreement no. 619609). The views expressed here are those of the authors only. The European Commission is not liable for any use that may be made of the information in this document.

10. Contributors

Lianshu Zheng.

11. References

11.1. Normative References

[I-D.ietf-ippm-metric-registry]
Bagnulo, M., Claise, B., Eardley, P., Morton, A., and A. Akhter, "Registry for Performance Metrics", draft-ietf-ippm-metric-registry-14 (work in progress), March 2018.

[I-D.ietf-ippm-port-twamp-test]
Morton, A. and G. Mirsky, "OWAMP and TWAMP Well-Known Port Assignments", draft-ietf-ippm-port-twamp-test-01 (work in progress), March 2018.

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

[RFC3432] Raisanen, V., Grotefeld, G., and A. Morton, "Network performance measurement with periodic streams", RFC 3432, DOI 10.17487/RFC3432, November 2002, <https://www.rfc-editor.org/info/rfc3432>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6038] Morton, A. and L. Ciavattone, "Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features", RFC 6038, DOI 10.17487/RFC6038, October 2010, <https://www.rfc-editor.org/info/rfc6038>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7717] Pentikousis, K., Ed., Zhang, E., and Y. Cui, "IKEv2-Derived Shared Secret Key for the One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP)", RFC 7717, DOI 10.17487/RFC7717, December 2015, <https://www.rfc-editor.org/info/rfc7717>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

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

[UML] ISO/IEC, "Information technology - Open Distributed Processing - Unified Modeling Language", April 2005.

11.2. Informative References

[NSC] John, W., Pentikousis, K., et al., "Research directions in network service chaining", Proc. SDN for Future Networks and Services (SDN4FNS), Trento, Italy IEEE, November 2013.

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <https://www.rfc-editor.org/info/rfc2330>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5618] Morton, A. and K. Hedayat, "Mixed Security Mode for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5618, DOI 10.17487/RFC5618, August 2009, <https://www.rfc-editor.org/info/rfc5618>.

[RFC5938] Morton, A. and M. Chiba, "Individual Session Control Feature for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5938, DOI 10.17487/RFC5938, August 2010, <https://www.rfc-editor.org/info/rfc5938>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC7312] Fabini, J. and A. Morton, "Advanced Stream and Sampling Framework for IP Performance Metrics (IPPM)", RFC 7312, DOI 10.17487/RFC7312, August 2014, <https://www.rfc-editor.org/info/rfc7312>.

[RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <https://www.rfc-editor.org/info/rfc7426>.

[RFC8018] Moriarty, K., Ed., Kaliski, B., and A. Rusch, "PKCS #5: Password-Based Cryptography Specification Version 2.1", RFC 8018, DOI 10.17487/RFC8018, January 2017, <https://www.rfc-editor.org/info/rfc8018>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

Appendix A. Detailed Data Model Examples

This appendix extends the example presented in Section 6 by configuring more fields such as authentication parameters, DSCP values and so on.

A.1. Control-Client

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <client>
      <admin-state>true</admin-state>
      <mode-preference-chain>
        <priority>0</priority>
        <mode>authenticated</mode>
      </mode-preference-chain>
      <mode-preference-chain>
        <priority>1</priority>
```

```
          <mode>unauthenticated</mode>
        </mode-preference-chain>
        <key-chain>
          <key-id>KeyClient1ToRouterA</key-id>
          <secret-key>c2VjcmV0MQ==</secret-key>
        </key-chain>
        <key-chain>
          <key-id>KeyForRouterB</key-id>
          <secret-key>c2VjcmV0Mg0K</secret-key>
        </key-chain>
        <ctrl-connection>
          <name>RouterA</name>
          <client-ip>203.0.113.1</client-ip>
          <server-ip>203.0.113.2</server-ip>
          <control-packet-dscp>32</control-packet-dscp>
          <key-id>KeyClient1ToRouterA</key-id>
          <test-session-request>
            <name>Test1</name>
            <sender-ip>203.0.113.3</sender-ip>
            <sender-udp-port>54000</sender-udp-port>
            <reflector-ip>203.0.113.4</reflector-ip>
            <reflector-udp-port>55000</reflector-udp-port>
            <padding-length>64</padding-length>
            <start-time>0</start-time>
          </test-session-request>
          <test-session-request>
            <name>Test2</name>
            <sender-ip>203.0.113.1</sender-ip>
            <sender-udp-port>54001</sender-udp-port>
            <reflector-ip>203.0.113.2</reflector-ip>
            <reflector-udp-port>55001</reflector-udp-port>
            <padding-length>128</padding-length>
            <start-time>0</start-time>
          </test-session-request>
        </ctrl-connection>
      </client>
    </twamp>
  </data>

  <?xml version="1.0" encoding="utf-8"?>
  <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
      <client>
        <admin-state>true</admin-state>
        <mode-preference-chain>
          <priority>0</priority>
          <mode>authenticated</mode>
        </mode-preference-chain>
```

```
        <mode-preference-chain>
          <priority>1</priority>
          <mode>unauthenticated</mode>
        </mode-preference-chain>
        <key-chain>
          <key-id>KeyClient1ToRouterA</key-id>
          <secret-key>c2VjcmV0MQ==</secret-key>
        </key-chain>
        <key-chain>
          <key-id>KeyForRouterB</key-id>
          <secret-key>c2VjcmV0Mg0K</secret-key>
        </key-chain>
        <ctrl-connection>
          <name>RouterA</name>
          <client-ip>2001:DB8:203:0:113::1</client-ip>
          <server-ip>2001:DB8:203:0:113::2</server-ip>
          <control-packet-dscp>32</control-packet-dscp>
          <key-id>KeyClient1ToRouterA</key-id>
          <test-session-request>
            <name>Test1</name>
            <sender-ip>2001:DB8:10:1:1::1</sender-ip>
            <sender-udp-port>54000</sender-udp-port>
            <reflector-ip>2001:DB8:10:1:1::2</reflector-ip>
            <reflector-udp-port>55000</reflector-udp-port>
            <padding-length>64</padding-length>
            <start-time>0</start-time>
          </test-session-request>
          <test-session-request>
            <name>Test2</name>
            <sender-ip>2001:DB8:203:0:113::1</sender-ip>
            <sender-udp-port>54001</sender-udp-port>
            <reflector-ip>2001:DB8:203:0:113::2</reflector-ip>
            <reflector-udp-port>55001</reflector-udp-port>
            <padding-length>128</padding-length>
            <start-time>0</start-time>
          </test-session-request>
        </ctrl-connection>
      </client>
    </twamp>
  </data>
```

A.2. Server

```
  <?xml version="1.0" encoding="utf-8"?>
  <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
      <server>
```

```
        <admin-state>true</admin-state>
        <servwait>1800</servwait>
        <control-packet-dscp>32</control-packet-dscp>
        <modes>authenticated unauthenticated</modes>
        <count>15</count>
        <key-chain>
          <key-id>KeyClient1ToRouterA</key-id>
          <secret-key>c2VjcmV0MQ==</secret-key>
        </key-chain>
        <key-chain>
          <key-id>KeyClient10ToRouterA</key-id>
          <secret-key>c2VjcmV0MTANCg==</secret-key>
        </key-chain>
        <ctrl-connection>
          <client-ip>203.0.113.1</client-ip>
          <client-tcp-port>16341</client-tcp-port>
          <server-ip>203.0.113.2</server-ip>
          <server-tcp-port>862</server-tcp-port>
          <control-packet-dscp>32</control-packet-dscp>
          <selected-mode>unauthenticated</selected-mode>
          <key-id>KeyClient1ToRouterA</key-id>
          <count>15</count>
        </ctrl-connection>
      </server>
    </twamp>
  </data>

  <?xml version="1.0" encoding="utf-8"?>
  <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
      <server>
        <admin-state>true</admin-state>
        <servwait>1800</servwait>
        <control-packet-dscp>32</control-packet-dscp>
        <modes>authenticated unauthenticated</modes>
        <count>15</count>
        <key-chain>
          <key-id>KeyClient1ToRouterA</key-id>
          <secret-key>c2VjcmV0MQ==</secret-key>
        </key-chain>
        <key-chain>
          <key-id>KeyClient10ToRouterA</key-id>
          <secret-key>c2VjcmV0MTANCg==</secret-key>
        </key-chain>
        <ctrl-connection>
          <client-ip>2001:DB8:203:0:113::1</client-ip>
          <client-tcp-port>16341</client-tcp-port>
          <server-ip>2001:DB8:203:0:113::2</server-ip>
```

```
          <server-tcp-port>862</server-tcp-port>
          <control-packet-dscp>32</control-packet-dscp>
          <selected-mode>unauthenticated</selected-mode>
          <key-id>KeyClient1ToRouterA</key-id>
          <count>15</count>
        </ctrl-connection>
      </server>
    </twamp>
  </data>
```

A.3. Session-Sender

```
  <?xml version="1.0" encoding="utf-8"?>
  <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
      <session-sender>
        <admin-state>true</admin-state>
        <test-session>
          <name>Test1</name>
          <ctrl-connection-name>RouterA</ctrl-connection-name>
          <fill-mode>zero</fill-mode>
          <number-of-packets>900</number-of-packets>
          <periodic-interval>1</periodic-interval>
          <sent-packets>2</sent-packets>
          <rcv-packets>2</rcv-packets>
          <last-sent-seq>1</last-sent-seq>
          <last-rcv-seq>1</last-rcv-seq>
        </test-session>
        <test-session>
          <name>Test2</name>
          <ctrl-connection-name>RouterA</ctrl-connection-name>
          <fill-mode>random</fill-mode>
          <number-of-packets>900</number-of-packets>
          <lambda>1</lambda>
          <max-interval>2</max-interval>
          <sent-packets>21</sent-packets>
          <rcv-packets>21</rcv-packets>
          <last-sent-seq>20</last-sent-seq>
          <last-rcv-seq>20</last-rcv-seq>
        </test-session>
      </session-sender>
    </twamp>
  </data>
```

A.4. Session-Reflector

[note: '\' line wrapping is for formatting only]

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
    <session-reflector>
      <admin-state>true</admin-state>
      <test-session>
        <sender-ip>203.0.113.3</sender-ip>
        <sender-udp-port>54000</sender-udp-port>
        <reflector-ip>203.0.113.4</reflector-ip>
        <reflector-udp-port>55000</reflector-udp-port>
        <sid>1232</sid>
        <parent-connection-client-ip>203.0.113.1</parent-connection-\
client-ip>
        <parent-connection-client-tcp-port>16341</parent-connection-\
client-tcp-port>
        <parent-connection-server-ip>203.0.113.2</parent-connection-\
server-ip>
        <parent-connection-server-tcp-port>862</parent-connection-se\
rver-tcp-port>
        <test-packet-dscp>32</test-packet-dscp>
        <sent-packets>2</sent-packets>
        <rcv-packets>2</rcv-packets>
        <last-sent-seq>1</last-sent-seq>
        <last-rcv-seq>1</last-rcv-seq>
      </test-session>
      <test-session>
        <sender-ip>203.0.113.1</sender-ip>
        <sender-udp-port>54001</sender-udp-port>
        <reflector-ip>192.0.2.2</reflector-ip>
        <reflector-udp-port>55001</reflector-udp-port>
        <sid>178943</sid>
        <parent-connection-client-ip>203.0.113.1</parent-connection-\
client-ip>
        <parent-connection-client-tcp-port>16341</parent-connection-\
client-tcp-port>
        <parent-connection-server-ip>203.0.113.2</parent-connection-\
server-ip>
        <parent-connection-server-tcp-port>862</parent-connection-se\
rver-tcp-port>
        <test-packet-dscp>32</test-packet-dscp>
        <sent-packets>21</sent-packets>
        <rcv-packets>21</rcv-packets>
        <last-sent-seq>20</last-sent-seq>
        <last-rcv-seq>20</last-rcv-seq>
```

```
        </test-session>
      </session-reflector>
    </twamp>
  </data>

  [note: '\' line wrapping is for formatting only]

  <?xml version="1.0" encoding="utf-8"?>
  <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <twamp xmlns="urn:ietf:params:xml:ns:yang:ietf-twamp">
      <session-reflector>
        <admin-state>true</admin-state>
        <test-session>
          <sender-ip>2001:DB8:10:1:1::1</sender-ip>
          <sender-udp-port>54000</sender-udp-port>
          <reflector-ip>2001:DB8:10:1:1::2</reflector-ip>
          <reflector-udp-port>55000</reflector-udp-port>
          <sid>1232</sid>
          <parent-connection-client-ip>2001:DB8:203:0:113::1</parent-c\
  onnection-client-ip>
          <parent-connection-client-tcp-port>16341</parent-connection-\
  client-tcp-port>
          <parent-connection-server-ip>2001:DB8:203:0:113::2</parent-c\
  onnection-server-ip>
          <parent-connection-server-tcp-port>862</parent-connection-se\
  rver-tcp-port>
          <test-packet-dscp>32</test-packet-dscp>
          <sent-packets>2</sent-packets>
          <rcv-packets>2</rcv-packets>
          <last-sent-seq>1</last-sent-seq>
          <last-rcv-seq>1</last-rcv-seq>
        </test-session>
        <test-session>
          <sender-ip>2001:DB8:203:0:113::1</sender-ip>
          <sender-udp-port>54001</sender-udp-port>
          <reflector-ip>2001:DB8:192:68::2</reflector-ip>
          <reflector-udp-port>55001</reflector-udp-port>
          <sid>178943</sid>
          <parent-connection-client-ip>2001:DB8:203:0:113::1</parent-c\
  onnection-client-ip>
          <parent-connection-client-tcp-port>16341</parent-connection-\
  client-tcp-port>
          <parent-connection-server-ip>2001:DB8:203:0:113::2</parent-c\
  onnection-server-ip>
          <parent-connection-server-tcp-port>862</parent-connection-se\
  rver-tcp-port>
          <test-packet-dscp>32</test-packet-dscp>
          <sent-packets>21</sent-packets>
```

```
          <rcv-packets>21</rcv-packets>
          <last-sent-seq>20</last-sent-seq>
          <last-rcv-seq>20</last-rcv-seq>
        </test-session>
      </session-reflector>
    </twamp>
  </data>
```

Appendix B. TWAMP Operational Commands

TWAMP operational commands could be performed programmatically or manually, e.g. using a command-line interface (CLI).

With respect to programmability, YANG can be used to define NETCONF Remote Procedure Calls (RPC), therefore it would be, in principle, possible to define TWAMP RPC operations for actions such as starting or stopping control connections or test sessions or groups of sessions; retrieving results; clearing stored results, and so on.

However, TWAMP [RFC5357] does not attempt to describe such operational actions. Refer also to Section 2 and the unlabeled links in Figure 1. In actual deployments different TWAMP implementations may support different sets of operational commands, with different restrictions. Therefore, this document considers it the responsibility of the individual implementation to define its corresponding TWAMP operational commands data model.

Authors' Addresses

Ruth Civil
Ciena Corporation
307 Legget Drive
Kanata, ON K2K 3C8
Canada

Email: gcivil@ciena.com
URI: www.ciena.com

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

Reshad Rahman
Cisco Systems
2000 Innovation Drive
Kanata, ON K2K 3E8
Canada

Email: rrahman@cisco.com

Mahesh Jethanandani
Xoriant Corporation
1248 Reamswood Drive
Sunnyvale, CA 94089
USA

Email: mjethanandani@gmail.com

Kostas Pentikousis (editor)
Travelping
Siemensdamm 50
Berlin 13629
Germany

Email: k.pentikousis@travelping.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 2, 2018

G. Mirsky
ZTE Corp.
G. Jun
ZTE Corporation
H. Nydell
Accedian Networks
October 29, 2017

Simple Two-way Active Measurement Protocol
draft-mirsky-ippm-stamp-01

Abstract

This document describes a Two-way Active Measurement Protocol which enables measurement of both one-way and round-trip performance metrics like delay, delay variation and packet loss.

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 https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on May 2, 2018.

Table of Contents

1. Introduction

2. Conventions used in this document

2.1. Terminology

STAMP - Simple Two-way Active Measurement Protocol

NTP - Network Time Protocol

PTP - Precision Time Protocol

2.2. Requirements Language

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

3. Softwarization of Performance Measurement

Instance of a Simple Two-way Active Measurement Protocol (STAMP) session between a Sender and a Reflector controlled by communication between a Configuration Client as a manager and Configuration Servers as agents of the configuration session that configures STAMP measurement between Sender and Reflector. The Configuration Client also issues queries to obtain operational state information and/or measurement results.

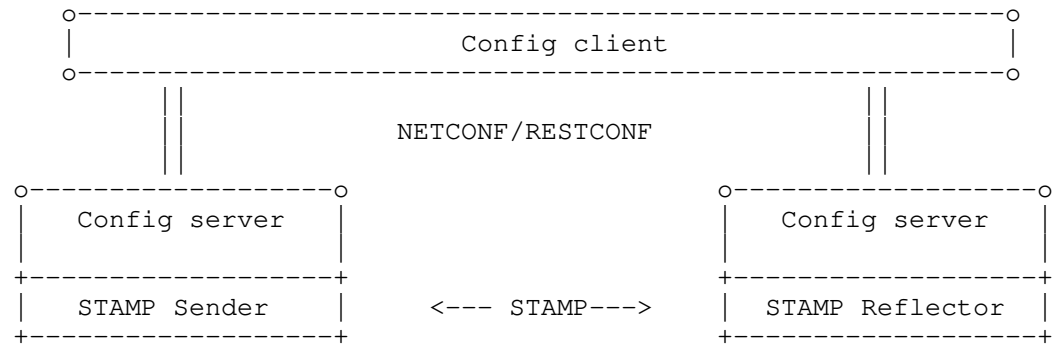


Figure 1: STAMP Reference Model

4. Theory of Operation

STAMP Sender transmits test packets toward STAMP Reflector. STAMP Reflector receives Sender's packet and acts according to the configuration and optional control information communicated in the Sender's test packet. STAMP defines two different test packet formats, one for packets transmitted by the STAMP-Sender and one for packets transmitted by the STAMP-Reflector. STAMP supports three modes: unauthenticated, authenticated, and encrypted. Unauthenticated STAMP test packets are compatible on the wire with unauthenticated TWAMP-Test [RFC5357] packet formats.

By default STAMP uses symmetrical packets, i.e. size of the packet transmitted by Reflector equals to the size of the packet received by the Reflector.

4.1. Sender Behavior and Packet Format

4.1.1. Sender Packet Format in Unauthenticated Mode

Because STAMP supports symmetrical test packets, STAMP Sender packet has minimum size of 44 octets in unauthenticated mode, see Figure 2, and 48 octets in authenticated or encrypted modes , see Figure 4.

For unauthenticated mode:

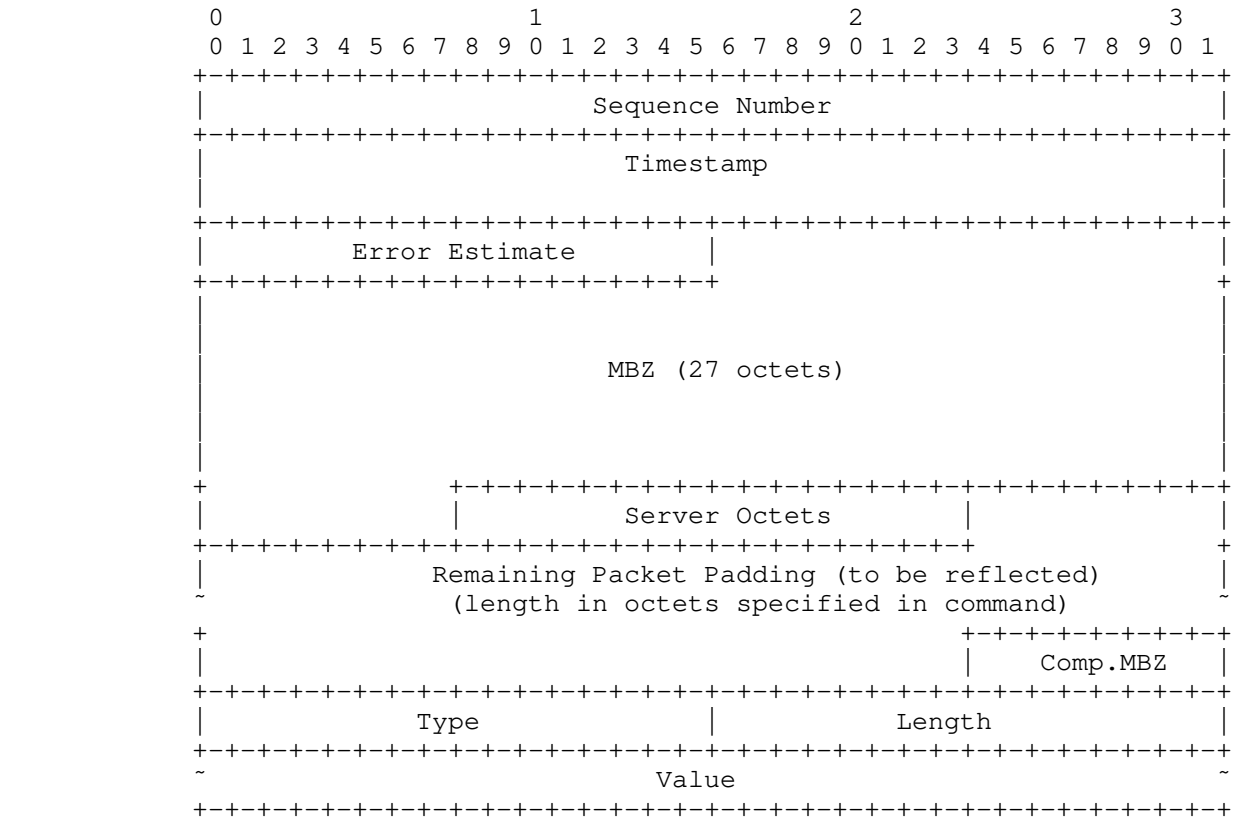


Figure 2: STAMP Sender test packet format in unauthenticated mode

where fields are defined as the following:

- Sequence Number is four octets long field. For each new session its value starts at zero and is incremented with each transmitted packet.

- Timestamp is eight octets long field. STAMP node MUST support Network Time Protocol (NTP) version 4 64-bit timestamp format [RFC5905]. STAMP node MAY support IEEE 1588v2 Precision Time Protocol truncated 64-bit timestamp format [IEEE.1588.2008].

- Error Estimate is two octets long field with format displayed in Figure 3

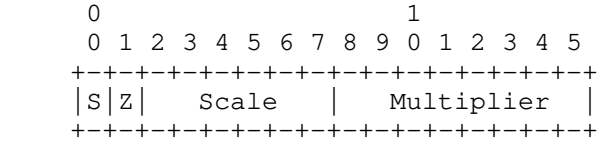


Figure 3: Error Estimate Format

  where S, Scale and Multiplier fields are interpreted as they have been defined in section 4.1.2 [RFC4656]; and Z field - as has been defined in section 2.3 [RFC8186]:

  * 0 - NTP 64 bit format of a timestamp;

  * 1 - PTPv2 truncated format of a timestamp.

- Must-be-Zero (MBZ) field in the sender unauthenticated packet is 27 octets long. It MUST be all zeroed on transmission and ignored on receipt.

- Server Octets field is two octets long field. It MUST follow the 27 octets long MBZ field. The Reflect Octets capability defined in [RFC6038]. The value in the Server Octets field equals to the number of octets the Reflector is expected to copy back to the Sender starting with the Server Octets field. Thus the minimal non-zero value for the Server Octets field is two and value of one is invalid. If none of Payload to be copied the value of the Server Octets field MUST be set to zero on transmit.

- Remaining Packet Padding is optional field of variable length. The number of octets in the Remaining Packet Padding field is the value of the Server Octets field less the length of the Server Octets field.

- Comp.MBZ is variable length field used to achieve alignment on word boundary. Thus the length of Comp.MBZ field may be only 0,

1, 2 or 3 octets. The value of the field MUST be zeroed on transmission and ignored on receipt.

The unauthenticated STAMP Sender packet MAY include Type-Length-Value encodings that immediately follow the Comp. MBZ field.

- Type field is two octets long. The value of the Type field is the codepoint allocated by IANA Section 6 that identifies data in the Value field.

- Length is two octets long field and its value is the length of the Value field in octets.

4.1.2. Sender Packet Format in Authenticated and Encrypted Modes

For authenticated and encrypted modes:

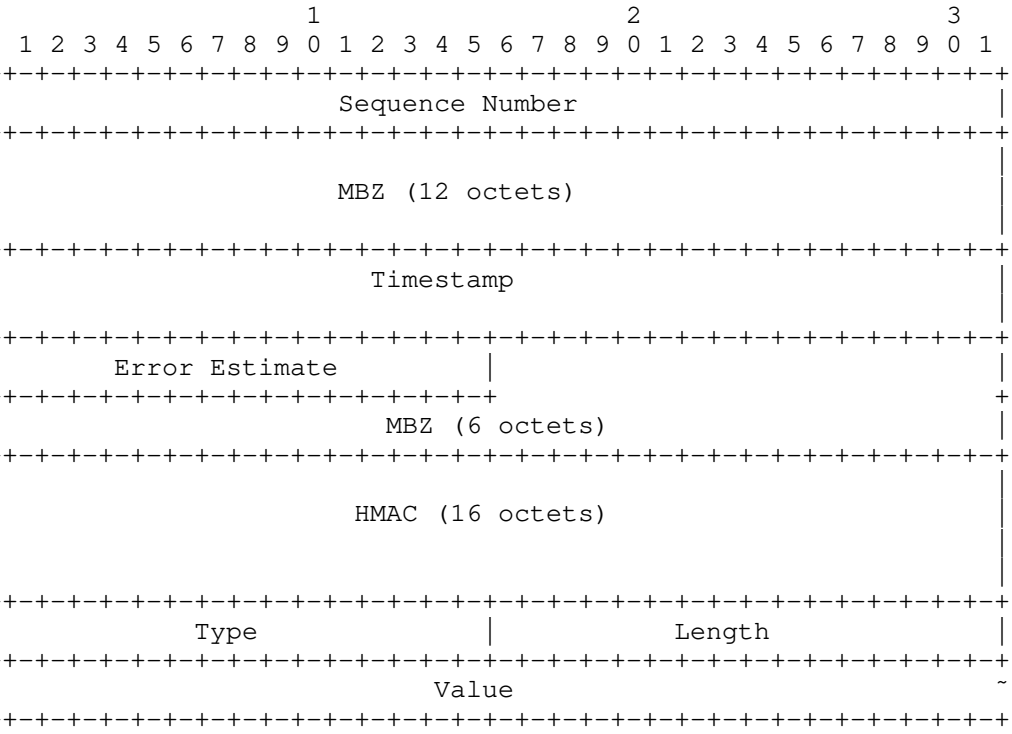


Figure 4: STAMP Sender test packet format in authenticated or encrypted modes

4.2. Reflector Behavior and Packet Format

The Reflector receives the STAMP test packet, verifies it, prepares and transmits the reflected test packet. [Editor note: Verification may include presence and content of TLVs in the STAMP test packet.]

4.2.1. Reflector Packet Format in Unauthenticated Mode

For unauthenticated mode:

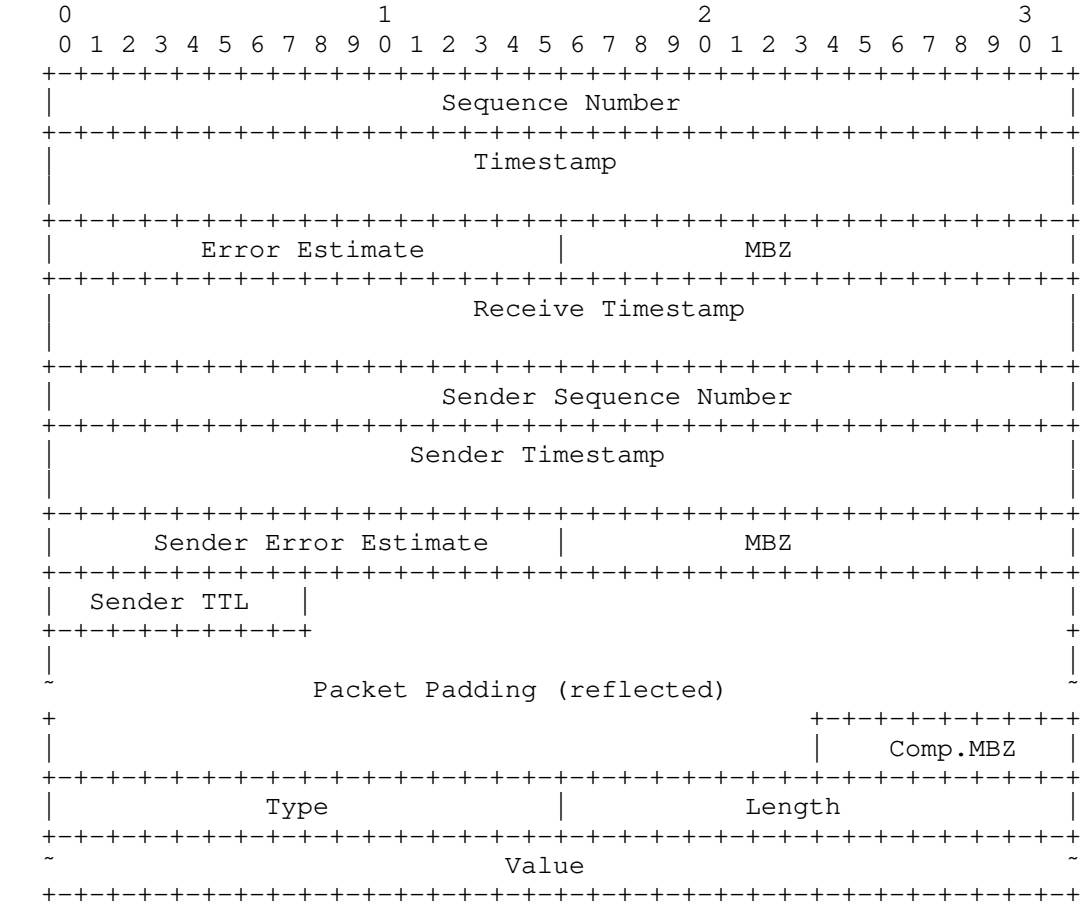


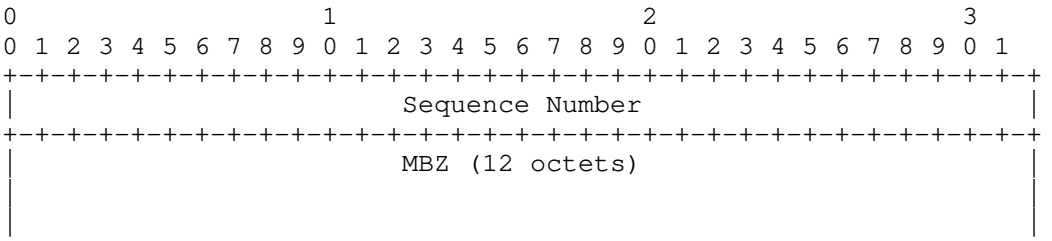
Figure 5: STAMP Reflector test packet format in unauthenticated mode

where fields are defined as the following:

- Sequence Number is four octets long field. The value of the Sequence Number field is set according to the mode of the STAMP Reflector:
  - in the stateless mode the Reflector copies the value from the received STAMP test packet's Sequence Number field;
  - in the stateful mode the Reflector counts the received STAMP test packets in each test session and uses that counter to set value of the Sequence Number field.
- Timestamp and Receiver Timestamp fields are each 8 octets long. The format of these fields, NTP or PTPv2, indicated by the Z flag of the Error Estimate field as described in Section 4.1.
- Error Estimate has the same size and interpretation as described in Section 4.1.
- Sender Sequence Number, Sender Timestamp, and Sender Error Estimate are copies of the corresponding fields in the STAMP test packet send by the Sender.
- Sender TTL is one octet long field and its value is the copy of the TTL field from the received STAMP test packet.
- Packet Padding (reflected) is optional variable length field. The length of the Packet Padding (reflected) field MUST be equal to the value of the Server Octets field (Figure 2). If the value is non-zero, the Reflector copies octets starting with the Server Octets field.
- Comp.MBZ is variable length field used to achieve alignment on word boundary. Thus the length of Comp.MBZ field may be only 0, 1, 2 or 3 octets. The value of the field MUST be zeroed on transmission and ignored on receipt.

4.2.2. Reflector Packet Format in Authenticated and Encrypted Modes

For authenticated and encrypted modes:



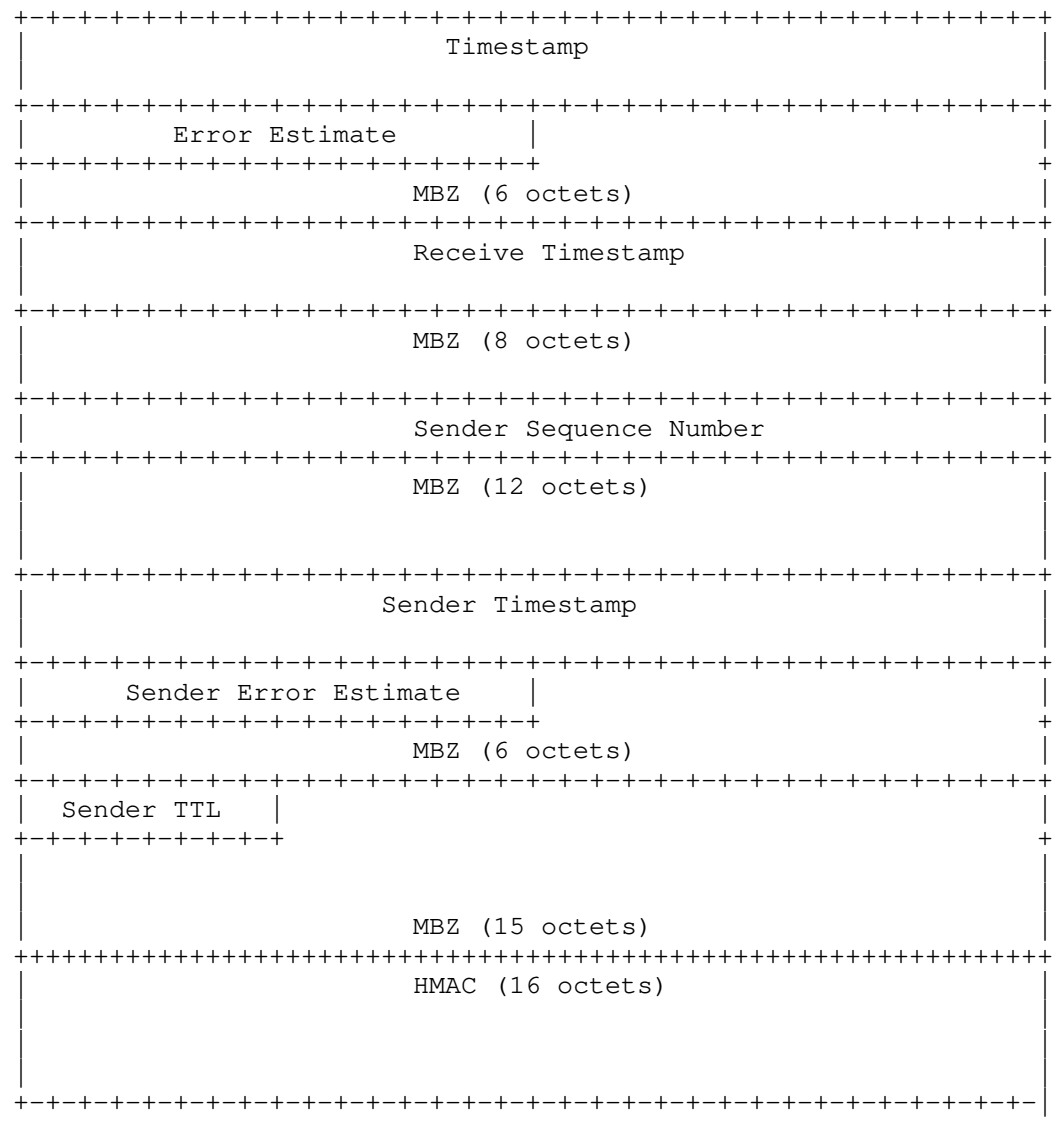


Figure 6: STAMP Reflector test packet format in authenticated or encrypted modes

5. TLV Extensions to STAMP

TBA

5.1. Extra Padding TLV

TBA

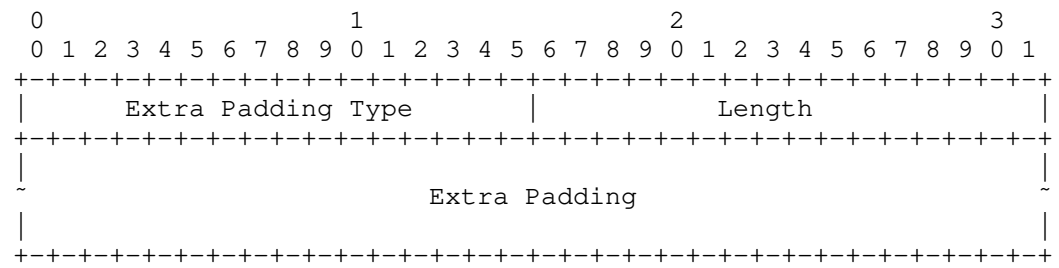


Figure 7: Extra Padding TLV

where fields are defined as the following:

- Extra Padding Type - TBA1 allocated by IANA Section 6.1
- Length - 2 octets long field equals length on the Extra Padding field in octets.
- Extra Padding - pseudo-random sequence of numbers. The field MAY be filled with all zeroes.

5.2. Location TLV

STAMP sender MAY include the Location TLV to request information from the reflector. The sender SHOULD NOT fill any information fields except for Type and Length. The reflector MUST validate the Length value against address family of the transport encapsulating the STAMP test packet. If the value of the Length field is invalid, the reflector MUST zero all fields and MUST NOT return any information to the sender. The reflector MUST ignore all other fields of the received Location TLV.

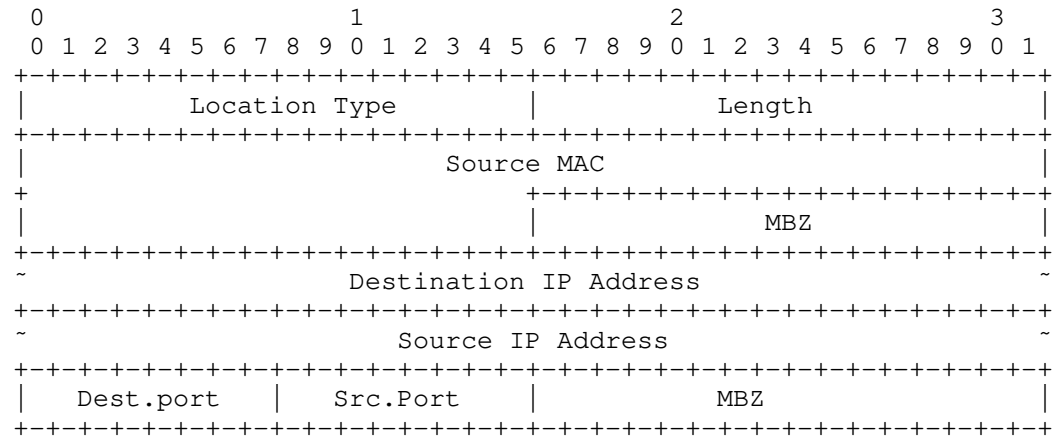


Figure 8: Reflector Location TLV

where fields are defined as the following:

- Location Type - TBA1 allocated by IANA Section 6.1
- Length - 2 octets long field equals length on the Value field in octets. Length field value MUST be 20 octets for IPv4 address family. For IPv6 address family value of the Length field MUST be 44 octets. All other values are invalid
- Source MAC - 6 octets 48 bits long field. The reflector MUST copy Source MAC of received STAMP packet into this field.
- MBZ - two octets long field. MUST be zeroed on transmission and ignored on reception.
- Destination IP Address - IPv4 or IPv6 destination address of the received by the reflector STAMP packet.
- Source IP Address - IPv4 or IPv6 source address of the received by the reflector STAMP packet.
- Dest.port - one octet long UDP destination port number of the received STAMP packet.
- Src.port - one octet long UDP source port number of the received STAMP packet.

5.3. Timestamp Information TLV

STAMP sender MAY include the Timestamp Information TLV to request information from the reflector. The sender SHOULD NOT fill any information fields except for Type and Length. The reflector MUST validate the Length value of the STAMP test packet. If the value of the Length field is invalid, the reflector MUST zero all fields and MUST NOT return any information to the sender.

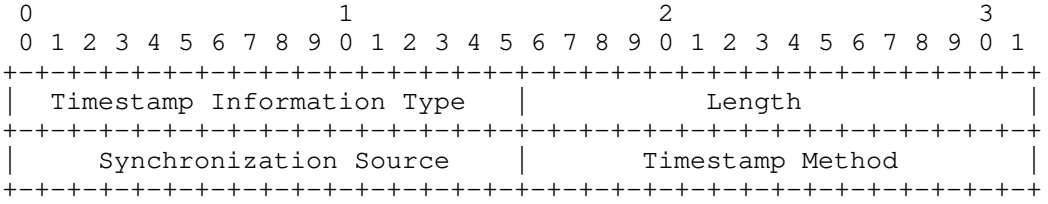


Figure 9: Timestamp Information TLV

where fields are defined as the following:

- Timestamp Information Type - TBA1 allocated by IANA Section 6.1
- Length - 2 octets long field, equals 4 octets.
- Synchronization Source - two octets long field that characterizes the source of clock synchronization at the reflector. The value is one of Section 6.2.
- Timestamp Method - two octets long field that characterizes timestamping method at the reflector. The value is one of Section 6.3. [Ed.note: Should it be split for ingress and egress?]

5.4. Class of Service TLV

The STAMP sender MAY include Class of Service TLV in the STAMP test packet. If the Class of Service TLV is present in the STAMP test packet and the value of the Op field equals Report (TBA5) value Section 6.4, then the STAMP reflector MUST copy DSCP and ECN values from the received STAMP test packet into DSCP and ECN fields of the Class of Service TLV of the reflected STAMP test packet. If the value of the Op field equals Set and Report (TBA6) Section 6.4, then the STAMP reflector MUST use DSCP value from the Class of Service TLV in the received STAMP test packet as DSCP value of STAMP reflected test packet and MUST copy DSCP and ECN values of the received STAMP test packet into DSCP and ECN fields of Class of Service TLV in the STAMP reflected packet.

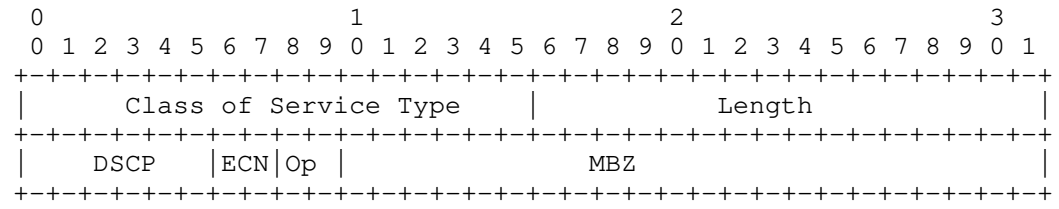


Figure 10: Class of Service TLV

where fields are defined as the following:

o

6. IANA Considerations

6.1. STAMP TLV Registry

IANA is requested to create STAMP TLV Type registry. All code points in the range 1 through 32759 in this registry shall be allocated according to the "IETF Review" procedure as specified in [RFC8126]. Code points in the range 32760 through 65279 in this registry shall be allocated according to the "First Come First Served" procedure as specified in [RFC8126]. Remaining code points are allocated according to the Table 1:

Value	Description	Reference
0	Reserved	This document
1- 32759	Unassigned	IETF Review
32760 - 65279	Unassigned	First Come First Served
65280 - 65519	Experimental	This document
65520 - 65534	Private Use	This document
65535	Reserved	This document

Table 1: STAMP TLV Type Registry

This document defines the following new values in STAMP TLV Type registry:

Value	Description	Reference
TBA1	Extra Padding	This document
TBA2	Location	This document
TBA3	Timestamp Information	This document
TBA4	Class of Service	This document

Table 2: STAMP Types

6.2. Synchronization Source Sub-registry

TBD

6.3. Timestamp Method Sub-registry

TBD

6.4. CoS Operation Sub-registry

TBD

7. Security Considerations

TBD

8. Acknowledgments

TBD

9. Normative References

[IEEE.1588.2008]
"Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Standard 1588, March 2008.

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

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC6038] Morton, A. and L. Ciavattone, "Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features", RFC 6038, DOI 10.17487/RFC6038, October 2010, <https://www.rfc-editor.org/info/rfc6038>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

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

[RFC8186] Mirsky, G. and I. Meilik, "Support of the IEEE 1588 Timestamp Format in a Two-Way Active Measurement Protocol (TWAMP)", RFC 8186, DOI 10.17487/RFC8186, June 2017, <https://www.rfc-editor.org/info/rfc8186>.

Authors' Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Guo Jun
ZTE Corporation
68# Zijinghua Road
Nanjing, Jiangsu 210012
P.R.China

Phone: +86 18105183663
Email: guo.jun2@zte.com.cn

Henrik Nydell
Accedian Networks

Email: hnydell@accedian.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 23, 2018

G. Mirsky
X. Min
ZTE Corp.
W. Luo
Ericsson

October 20, 2017

Simple Two-way Active Measurement Protocol (STAMP) Data Model
draft-mirsky-ippm-stamp-yang-00

Abstract

This document specifies the data model for implementations of Sender and Reflector for Simple Two-way Active Measurement Protocol (STAMP) mode using YANG.

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 https://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 23, 2018.

Copyright Notice

Table of Contents

1. Introduction

The Simple Two-way Active Measurement Protocol (STAMP) [I-D.mirsky-ippm-stamp] can be used to measure performance parameters of IP networks such as latency, jitter, and packet loss by sending test packets and monitoring their experience in the network. The STAMP protocol [Editor:ref to STAMP draft] in unauthenticated mode is on-wire compatible with STAMP Light, mdiscussed in Appendix I [RFC5357]. The STAMP Light is known to have many implementations though no common management framework being defined, thus leaving some aspects of test packet processing to interpretation. As one of goals of STAMP is to support these variations, this document presents their analysis; describes common STAMP and STAMP model while allowing for STAMP extensions in the future. This document defines the STAMP data model and specifies it formally using the YANG data modeling language [RFC6020].

1.1. Conventions used in this document

1.1.1. Requirements Language

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

2. Scope, Model, and Applicability

The scope of this document includes model of the STAMP as defined in [Editor:ref to STAMP draft].

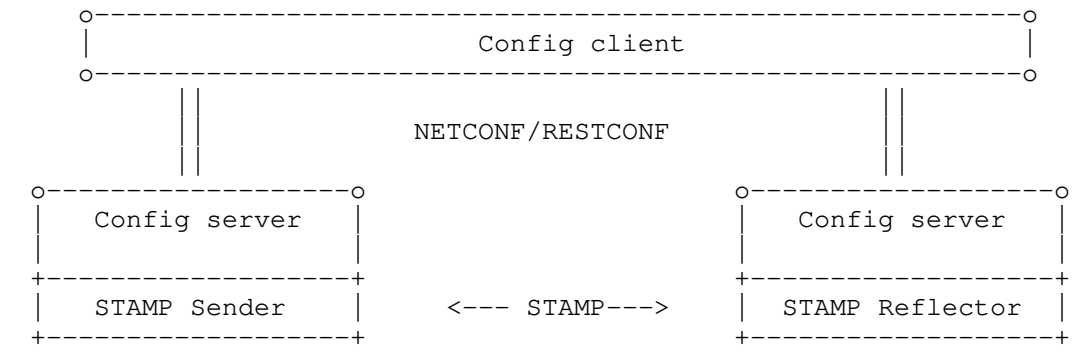


Figure 1: STAMP Reference Model

2.1. Data Model Parameters

This section describes all the parameters of the the stamp data model.

2.1.1. STAMP-Sender

The stamp-session-sender container holds items that are related to the configuration of the stamp Session-Sender logical entity.

The stamp-session-sender-state container holds information about the state of the particular STAMP test session.

RPCs stamp-sender-start and stamp-sender-stop respectively start and stop the referenced by session-id STAMP test session.

2.1.1.1. Controls for Test Session and Preforrmance Metric Calculation

The data model supports several scenarios for a STAMP Sender to execute test sessions and calculate performance metrics:

The test mode in which the test packets are sent unbound in time at defined by the parameter 'interval' in the stamp-session-sender container frequency is referred as continuous mode. Performance metrics in the continuous mode are calculated at period defined by the parameter 'measurement-interval'.

The test mode that has specific number of the test packets configured for the test session in the 'number-of-packets' parameter is referred as periodic mode. The test session may be repeated by the STAMP-Sender with the same parameters. The 'repeat' parameter defines number of tests and the 'repeat-interval' - the interval between the consecuitive tests. The performance metrics are calculated after each test session when the interval defined by the 'session-timeout' expires.

2.1.2. STAMP-Reflector

The stamp-session-reflector container holds items that are related to the configuration of the STAMP Session-Reflector logical entity.

The stamp-session-refl-state container holds Session-Reflector state data for the particular STAMP test session.

3. Data Model

Creating STAMP data model presents number of challenges and among them is identification of a test-session at Session-Reflector. A Session-Reflector MAY require only as little as its IP and UDP port number in received STAMP-Test packet to spawn new test session. More so, to test processing of Class-of-Service along the same route in Equal Cost Multi-Path environment Session-Sender may run STAMP test sessions concurrently using the same source IP address, source UDP port number, destination IP address, and destination UDP port number. Thus the only parameter that can be used to differentiate these test sessions would be DSCP value. The DSCP field may get re-marked along the path and without use of [RFC7750] that will go undetected, but by using five-tuple instead of four-tuple as a key we can ensure that STAMP test packets that are considered as different test sessions follow the same path even in ECMP environments.

3.1. Tree Diagram

```
module: ietf-stamp
    +--rw stamp
    |  +--rw stamp-session-sender {session-sender}?
    |  |  +--rw sender-enable?   enable
    |  |  +--rw test-session* [session-id]
    |  |     +--rw session-id                    uint32
    |  |     +--rw test-session-enable?          enable
    |  |     +--rw number-of-packets?            union
    |  |     +--rw packet-padding-size?          uint32
    |  |     +--rw interval?                     uint32
    |  |     +--rw session-timeout?              uint32
```

```
|  |     +--rw measurement-interval?          uint32
|  |     +--rw repeat?                        union
|  |     +--rw repeat-interval?               uint32
|  |     +--rw dscp-value?                    inet:dscp
|  |     +--rw test-session-reflector-mode?   session-reflector-mode
|  |     +--rw sender-ip                      inet:ip-address
|  |     +--rw sender-udp-port                inet:port-number
|  |     +--rw reflector-ip                   inet:ip-address
|  |     +--rw reflector-udp-port?            inet:port-number
|  |     +--rw authentication-params! {stamp-authentication}?
|  |     |  +--rw key-chain?   kc:key-chain-ref
|  |     +--rw first-percentile?              percentile
|  |     +--rw second-percentile?             percentile
|  |     +--rw third-percentile?              percentile
|  +--rw stamp-session-reflector {session-reflector}?
|     +--rw reflector-enable?       enable
|     +--rw ref-wait?               uint32
|     +--rw reflector-mode-state?   session-reflector-mode
|     +--rw test-session* [session-id]
|        +--rw session-id               uint32
|        +--rw dscp-handling-mode?      session-dscp-mode
|        +--rw dscp-value?              inet:dscp
|        +--rw sender-ip                inet:ip-address
|        +--rw sender-udp-port          inet:port-number
|        +--rw reflector-ip             inet:ip-address
|        +--rw reflector-udp-port?      inet:port-number
|        +--rw authentication-params! {stamp-authentication}?
|           +--rw key-chain?   kc:key-chain-ref
+--ro stamp-state
   +--ro stamp-session-sender-state {session-sender}?
      |  +--ro test-session-state* [session-id]
      |     +--ro session-id              uint32
      |     +--ro sender-session-state?   enumeration
      |     +--ro current-stats
      |     |  +--ro start-time                yang:date-and-time
      |     |  +--ro packet-padding-size?      uint32
      |     |  +--ro interval?                 uint32
      |     |  +--ro duplicate-packets?        uint32
      |     |  +--ro reordered-packets?        uint32
      |     |  +--ro sender-ip                 inet:ip-address
      |     |  +--ro sender-udp-port           inet:port-number
      |     |  +--ro reflector-ip              inet:ip-address
      |     |  +--ro reflector-udp-port?       inet:port-number
      |     |  +--ro dscp?                     inet:dscp
      |     |  +--ro sent-packets?             uint32
      |     |  +--ro rcv-packets?              uint32
      |     |  +--ro sent-packets-error?       uint32
      |     |  +--ro rcv-packets-error?        uint32
```

```
        |       |  +--ro last-sent-seq?            uint32
        |       |  +--ro last-rcv-seq?             uint32
        |       |  +--ro two-way-delay
        |       |  |  +--ro delay
        |       |  |  |  +--ro min?   yang:gauge32
        |       |  |  |  +--ro max?   yang:gauge32
        |       |  |  |  +--ro avg?   yang:gauge32
        |       |  |  +--ro delay-variation
        |       |  |     +--ro min?   uint32
        |       |  |     +--ro max?   uint32
        |       |  |     +--ro avg?   uint32
        |       |  +--ro one-way-delay-far-end
        |       |  |  +--ro delay
        |       |  |  |  +--ro min?   yang:gauge32
        |       |  |  |  +--ro max?   yang:gauge32
        |       |  |  |  +--ro avg?   yang:gauge32
        |       |  |  +--ro delay-variation
        |       |  |     +--ro min?   uint32
        |       |  |     +--ro max?   uint32
        |       |  |     +--ro avg?   uint32
        |       |  +--ro one-way-delay-near-end
        |       |  |  +--ro delay
        |       |  |  |  +--ro min?   yang:gauge32
        |       |  |  |  +--ro max?   yang:gauge32
        |       |  |  |  +--ro avg?   yang:gauge32
        |       |  |  +--ro delay-variation
        |       |  |     +--ro min?   uint32
        |       |  |     +--ro max?   uint32
        |       |  |     +--ro avg?   uint32
        |       |  +--ro low-percentile
        |       |  |  +--ro delay-percentile
        |       |  |  |  +--ro rtt-delay?        percentile
        |       |  |  |  +--ro near-end-delay?   percentile
        |       |  |  |  +--ro far-end-delay?    percentile
        |       |  |  +--ro delay-variation-percentile
        |       |  |     +--ro rtt-delay-variation?        percentile
        |       |  |     +--ro near-end-delay-variation?   percentile
        |       |  |     +--ro far-end-delay-variation?    percentile
        |       |  +--ro mid-percentile
        |       |  |  +--ro delay-percentile
        |       |  |  |  +--ro rtt-delay?        percentile
        |       |  |  |  +--ro near-end-delay?   percentile
        |       |  |  |  +--ro far-end-delay?    percentile
        |       |  |  +--ro delay-variation-percentile
        |       |  |     +--ro rtt-delay-variation?        percentile
        |       |  |     +--ro near-end-delay-variation?   percentile
        |       |  |     +--ro far-end-delay-variation?    percentile
        |       |  +--ro high-percentile
```

```
|      |  |  +--ro delay-percentile
|      |  |  |  +--ro rtt-delay?         percentile
|      |  |  |  +--ro near-end-delay?    percentile
|      |  |  |  +--ro far-end-delay?     percentile
|      |  |  +--ro delay-variation-percentile
|      |  |     +--ro rtt-delay-variation?        percentile
|      |  |     +--ro near-end-delay-variation?   percentile
|      |  |     +--ro far-end-delay-variation?    percentile
|      |  +--ro two-way-loss
|      |  |  +--ro loss-count?          int32
|      |  |  +--ro loss-ratio?          percentage
|      |  |  +--ro loss-burst-max?      int32
|      |  |  +--ro loss-burst-min?      int32
|      |  |  +--ro loss-burst-count?    int32
|      |  +--ro one-way-loss-far-end
|      |  |  +--ro loss-count?          int32
|      |  |  +--ro loss-ratio?          percentage
|      |  |  +--ro loss-burst-max?      int32
|      |  |  +--ro loss-burst-min?      int32
|      |  |  +--ro loss-burst-count?    int32
|      |  +--ro one-way-loss-near-end
|      |     +--ro loss-count?          int32
|      |     +--ro loss-ratio?          percentage
|      |     +--ro loss-burst-max?      int32
|      |     +--ro loss-burst-min?      int32
|      |     +--ro loss-burst-count?    int32
|      +--ro history-stats* [id]
|         +--ro id                          uint32
|         +--ro end-time                    yang:date-and-time
|         +--ro number-of-packets?          uint32
|         +--ro packet-padding-size?        uint32
|         +--ro interval?                   uint32
|         +--ro duplicate-packets?          uint32
|         +--ro reordered-packets?          uint32
|         +--ro loss-packets?               uint32
|         +--ro sender-ip                   inet:ip-address
|         +--ro sender-udp-port             inet:port-number
|         +--ro reflector-ip                inet:ip-address
|         +--ro reflector-udp-port?         inet:port-number
|         +--ro dscp?                       inet:dscp
|         +--ro sent-packets?               uint32
|         +--ro rcv-packets?                uint32
|         +--ro sent-packets-error?         uint32
|         +--ro rcv-packets-error?          uint32
|         +--ro last-sent-seq?              uint32
|         +--ro last-rcv-seq?               uint32
|         +--ro two-way-delay
|         |  +--ro delay
```

```
        |          |  |  +--ro min?   yang:gauge32
        |          |  |  +--ro max?   yang:gauge32
        |          |  |  +--ro avg?   yang:gauge32
        |          |  +--ro delay-variation
        |          |     +--ro min?   uint32
        |          |     +--ro max?   uint32
        |          |     +--ro avg?   uint32
        |          +--ro one-way-delay-far-end
        |          |  +--ro delay
        |          |  |  +--ro min?   yang:gauge32
        |          |  |  +--ro max?   yang:gauge32
        |          |  |  +--ro avg?   yang:gauge32
        |          |  +--ro delay-variation
        |          |     +--ro min?   uint32
        |          |     +--ro max?   uint32
        |          |     +--ro avg?   uint32
        |          +--ro one-way-delay-near-end
        |             +--ro delay
        |             |  +--ro min?   yang:gauge32
        |             |  +--ro max?   yang:gauge32
        |             |  +--ro avg?   yang:gauge32
        |             +--ro delay-variation
        |                +--ro min?   uint32
        |                +--ro max?   uint32
        |                +--ro avg?   uint32
        +--ro stamp-session-refl-state {session-reflector}?
           +--ro reflector-light-admin-status    boolean
           +--ro test-session-state* [session-id]
              +--ro session-id            uint32
              +--ro sent-packets?         uint32
              +--ro rcv-packets?          uint32
              +--ro sent-packets-error?   uint32
              +--ro rcv-packets-error?    uint32
              +--ro last-sent-seq?        uint32
              +--ro last-rcv-seq?         uint32
              +--ro sender-ip             inet:ip-address
              +--ro sender-udp-port       inet:port-number
              +--ro reflector-ip          inet:ip-address
              +--ro reflector-udp-port?   inet:port-number

  rpcs:
    +---x stamp-sender-start
    |  +---w input
    |     +---w session-id    uint32
    +---x stamp-sender-stop
       +---w input
          +---w session-id    uint32
```

3.2. YANG Module

```
<CODE BEGINS> file "ietf-stamp@2017-10-20.yang"

module ietf-stamp {
   namespace "urn:ietf:params:xml:ns:yang:ietf-stamp";
    //namespace need to be assigned by IANA
   prefix "ietf-stamp";

   import ietf-inet-types {
     prefix inet;
   }
   import ietf-yang-types {
     prefix yang;
   }
   import ietf-key-chain {
     prefix kc;
   }

   organization
     "IETF IPPM (IP Performance Metrics) Working Group";

   contact
     "draft-mirsky-ippm-stamp-yang@tools.ietf.org";

   description "STAMP Data Model";

   revision "2017-10-20" {
     description
     "00 version. Base STAMP specification is covered";
     reference "";
   }

   feature session-sender {
     description
     "This feature relates to the device functions as the
     STAMP Session-Sender";
   }

   feature session-reflector {
     description
     "This feature relates to the device functions as the
     STAMP Session-Reflector";
   }

   feature stamp-authentication {
     description
```

```
     "STAMP authentication supported";
   }

   typedef enable {
      type boolean;
      description "enable";
   }

   typedef session-reflector-mode {
        type enumeration {
                enum stateful {
                        description
                        "When the Session-Reflector is stateful,
                        i.e. is aware of STAMP-Test session state.";
                }
                enum stateless {
                        description
                        "When the Session-Reflector is stateless,
                        i.e. is not aware of the state of
                        STAMP-Test session.";
                }
        }
        description "State of the Session-Reflector";
   }

   typedef session-dscp-mode {
        type enumeration {
                enum copy-received-value {
                        description
                        "Use DSCP value copied from received
                        STAMP test packet of the test session.";
                }
                enum use-configured-value {
                        description
                        "Use DSCP value configured for this
                        test session on the Session-Reflector.";
                }
        }
        description
        "DSCP handling mode by Session-Reflector.";
   }

   typedef percentage {
        type decimal64 {
                fraction-digits 5;
        }
        description "Percentage";
   }
```

```
   typedef percentile {
        type decimal64 {
                fraction-digits 2;
        }
        description
        "Percentile is a measure used in statistics
        indicating the value below which a given
        percentage of observations in a group of
        observations fall.";
   }

   grouping maintenance-statistics {
     description "Maintenance statistics grouping";
     leaf sent-packets {
       type uint32;
       description "Packets sent";
     }
     leaf rcv-packets {
       type uint32;
       description "Packets received";
     }
     leaf sent-packets-error {
       type uint32;
       description "Packets sent error";
     }
     leaf rcv-packets-error {
       type uint32;
       description "Packets received error";
     }
     leaf last-sent-seq {
       type uint32;
       description "Last sent sequence number";
     }
     leaf last-rcv-seq {
       type uint32;
       description "Last received sequence number";
     }
   }

grouping stamp-session-percentile {
   description "Percentile grouping";
   leaf first-percentile {
      type percentile;
      default 95.00;
      description
      "First percentile to report";
   }
   leaf second-percentile {
```

```
      type percentile;
      default 99.00;
      description
      "Second percentile to report";
   }
   leaf third-percentile {
      type percentile;
      default 99.90;
      description
      "Third percentile to report";
   }
}

   grouping delay-statistics {
     description "Delay statistics grouping";
     container delay {
       description "Packets transmitted delay";
                 leaf min {
                   type yang:gauge32;
                   units microseconds;
                   description
                   "Min of Packets transmitted delay";
                 }
                 leaf max {
                    type yang:gauge32;
                    units microseconds;
                    description
                    "Max of Packets transmitted delay";
                 }
                 leaf avg {
                    type yang:gauge32;
                    units microseconds;
                    description
                    "Avg of Packets transmitted delay";
                 }
     }

         container delay-variation {
                 description
                 "Packets transmitted delay variation";
                 leaf min {
                   type uint32;
                   units microseconds;
                   description
                   "Min of Packets transmitted
                   delay variation";
                 }
                 leaf max {
```

```
                  type uint32;
                  units microseconds;
                  description
                  "Max of Packets transmitted
                  delay variation";
                 }
                 leaf avg {
                  type uint32;
                  units microseconds;
                  description
                  "Avg of Packets transmitted
                  delay variation";
                 }
          }
   }
     grouping time-percentile-report {
        description "Delay percentile report grouping";
        container delay-percentile {
                description
                "Report round-trip, near- and far-end delay";
                leaf rtt-delay {
                        type percentile;
                        description
                        "Percentile of round-trip delay";
                }
                leaf near-end-delay {
                        type percentile;
                        description
                        "Percentile of near-end delay";
                }
                leaf far-end-delay {
                        type percentile;
                        description
                        "Percentile of far-end delay";
                }
        }
        container delay-variation-percentile {
                description
                "Report round-trip, near- and far-end delay variation";
                leaf rtt-delay-variation {
                        type percentile;
                        description
                        "Percentile of round-trip delay-variation";
                }
                leaf near-end-delay-variation {
                        type percentile;
                        description
                        "Percentile of near-end delay variation";
```

```
                }
                leaf far-end-delay-variation {
                        type percentile;
                        description
                        "Percentile of far-end delay-variation";
                }
        }
     }

     grouping packet-loss-statistics {
        description
        "Grouping for Packet Loss statistics";
        leaf loss-count {
                type int32;
                description
                "Number of lost packets
                during the test interval.";
        }
        leaf loss-ratio {
                type percentage;
                description
                "Ratio of packets lost to packets
                sent during the test interval.";
        }
        leaf loss-burst-max {
                type int32;
                description
                "Maximum number of consequtively
                lost packets during the test interval.";
        }
        leaf loss-burst-min {
                type int32;
                description
                "Minimum number of consequtively
                lost packets during the test interval.";
        }
                leaf loss-burst-count {
                type int32;
                description
                "Number of occasions with packet
                loss during the test interval.";
                }
     }

   grouping session-parameters {
     description
     "Parameters common among
     Session-Sender and Session-Reflector";
```

```
    leaf sender-ip {
      type inet:ip-address;
      mandatory true;
      description "Sender IP address";
    }
    leaf sender-udp-port {
      type inet:port-number {
        range "49152..65535";
      }
      mandatory true;
      description "Sender UDP port number";
    }
    leaf reflector-ip {
      type inet:ip-address;
      mandatory true;
      description "Reflector IP address";
    }
    leaf reflector-udp-port {
      type inet:port-number{
        range "862 | 49152..65535";
      }
      default 862;
      description "Reflector UDP port number";
    }
  }

  grouping session-auth-params {
     description
     "Grouping for STAMP authentication parameters";
     container authentication-params {
        if-feature stamp-authentication;
        presence "Enables STAMP authentication";
        description
        "Parameters for STAMP Light authentication";
        leaf key-chain {
           type kc:key-chain-ref;
           description "Name of key-chain";
        }
     }
  }

  /* Configuration Data */
  container stamp {
    description
    "Top level container for stamp configuration";

   container stamp-session-sender {
      if-feature session-sender;
```

```
      description "stamp Session-Sender container";

      leaf sender-enable {
        type enable;
        default "true";
        description
        "Whether this network element is enabled to
        act as STAMP Sender";
      }

      list test-session {
        key "session-id";
              unique "sender-ip sender-udp-port reflector-ip"
        +" reflector-udp-port dscp-value";
        description
        "This structure is a container of test session
        managed objects";

        leaf session-id {
          type uint32;
          description "Session ID";
        }

        leaf test-session-enable {
          type enable;
          default "true";
          description
          "Whether this STAMP Test session is enabled";
        }

        leaf number-of-packets {
          type union {
               type uint32 {
                       range 1..4294967294 {
                               description
                               "The overall number of UDP test packet
                               to be transmitted by the sender for this
                               test session";
                       }
               }
               type enumeration {
                       enum forever {
                               description
                               "Indicates that the test session SHALL
                               be run *forever*.";
                       }
               }
          }
```

```
          default 10;
          description
          "This value determines if the STAMP-Test session is
             bound by number of test packets or not.";
        }

        leaf packet-padding-size {
          type uint32;
          default 27;
          description
          "Size of the Packet Padding. Suggested to run
          Path MTU Discovery to avoid packet fragmentation in
          IPv4 and packet blackholing in IPv6";
        }

        leaf interval  {
          type uint32;
          units microseconds;
          description
          "Time interval between transmission of two
          consecutive packets in the test session in
          microseconds";
        }

           leaf session-timeout {
             when "../number-of-packets != 'forever'" {
               description
               "Test session timeout only valid if the
               test mode is periodic.";
          }
          type uint32;
          units "seconds";
          default 900;
          description
          "The timeout value for the Session-Sender to
          collect outstanding reflected packets.";
        }

        leaf measurement-interval {
          when "../number-of-packets = 'forever'" {
               description
               "Valid only when the test to run forever,
               i.e. continuously.";
          }
          type uint32;
          units "seconds";
          default 60;
          description
```

```
           "Interval to calculate performance metric when
                the test mode is 'continuous'.";
            }

            leaf repeat {
              type union {
                type uint32 {
                        range 0..4294967294;
                }
                type enumeration {
                        enum forever {
                                description
                                "Indicates that the test session SHALL
                                be repeated *forever* using the
                                information in repeat-interval
                                parameter, and SHALL NOT decrement
                                the value.";
                        }
                }
              }
              default 0;
              description
              "This value determines if the STAMP-Test session must
              be repeated. When a test session has completed, the
              repeat parameter is checked. The default value
              of 0 indicates that the session MUST NOT be repeated.
              If the repeat value is 1 through 4,294,967,294
              then the test session SHALL be repeated using the
              information in repeat-interval parameter.
              The implementation MUST decrement the value of repeat
              after determining a repeated session is expected.";
            }

            leaf repeat-interval {
                when "../repeat != '0'";
                type uint32;
                units seconds;
                default 0;
                description
                "This parameter determines the timing of repeated
                STAMP-Test sessions when repeat is more than 0.";
            }

            leaf dscp-value {
              type inet:dscp;
              default 0;
              description
              "DSCP value to be set in the test packet.";
```

```
            }

            leaf test-session-reflector-mode {
                 type session-reflector-mode;
                 default "stateless";
                 description
                 "The mode of STAMP-Reflector for the test session.";
            }

          uses session-parameters;
          uses session-auth-params;
          uses stamp-session-percentile;
        }
      }

      container stamp-session-reflector {
        if-feature session-reflector;
        description
        "stamp Session-Reflector container";
          leaf reflector-enable {
            type enable;
            default "true";
            description
            "Whether this network element is enabled to
            act as stamp Reflector";
          }

          leaf ref-wait {
          type uint32 {
            range 1..604800;
          }
          units seconds;
          default 900;
          description
          "REFWAIT(STAMP test session timeout in seconds),
          the default value is 900";
        }

        leaf reflector-mode-state {
          type session-reflector-mode;
          default stateless;
          description
          "The state of the mode of the stamp
          Session-Reflector";
         }

         list test-session {
           key "session-id";
```

```
              unique "sender-ip sender-udp-port reflector-ip"
          +" reflector-udp-port";
          description
          "This structure is a container of test session
          managed objects";

        leaf session-id {
          type uint32;
          description "Session ID";
        }

        leaf dscp-handling-mode {
          type session-dscp-mode;
          default copy-received-value;
          description
          "Session-Reflector handling of DSCP:
                - use value copied from received STAMP-Test packet;
                - use value explicitly configured";
           }

           leaf dscp-value {
             when "../dscp-handling-mode = 'use-configured-value'";
             type inet:dscp;
             default 0;
             description
             "DSCP value to be set in the reflected packet
                if dscp-handling-mode is set to use-configured-value.";
           }

        uses session-parameters;
        uses session-auth-params;
       }
     }
   }

 /* Operational state data nodes */
 container stamp-state{
   config "false";
   description
   "Top level container for stamp state data";

   container stamp-session-sender-state {
     if-feature session-sender;
     description
     "Session-Sender container for state data";
     list test-session-state{
       key "session-id";
       description
```

```
       "This structure is a container of test session
       managed objects";

       leaf session-id {
         type uint32;
         description "Session ID";
       }

       leaf sender-session-state {
         type enumeration {
           enum active {
             description "Test session is active";
           }
           enum ready {
             description "Test session is idle";
           }
         }
         description
         "State of the particular stamp test
         session at the sender";
       }

       container current-stats {
         description
         "This container contains the results for the current
          Measurement Interval in a Measurement session ";
          leaf start-time {
            type yang:date-and-time;
            mandatory true;
            description
            "The time that the current Measurement Interval started";
          }

          leaf packet-padding-size {
            type uint32;
            default 27;
            description
            "Size of the Packet Padding. Suggested to run
            Path MTU Discovery to avoid packet fragmentation
            in IPv4 and packet backholing in IPv6";
          }

          leaf interval  {
            type uint32;
            units microseconds;
           description
           "Time interval between transmission of two
           consecutive packets in the test session";
```

```
          }

          leaf duplicate-packets  {
            type uint32;
           description "Duplicate packets";
          }
          leaf reordered-packets  {
            type uint32;
           description "Reordered packets";
          }

          uses session-parameters;
          leaf dscp {
            type inet:dscp;
            description
            "The DSCP value that was placed in the header of
            STAMP UDP test packets by the Session-Sender.";
          }
          uses maintenance-statistics;

          container two-way-delay {
            description
            "two way delay result of the test session";
            uses delay-statistics;
          }

          container one-way-delay-far-end {
            description
            "one way delay far-end of the test session";
            uses delay-statistics;
          }

          container one-way-delay-near-end {
            description
            "one way delay near-end of the test session";
            uses delay-statistics;
          }

          container low-percentile {
                when "/stamp/stamp-session-sender/"
                +"test-session[session-id]/"
                +"first-percentile != '0.00'" {
                        description
                        "Only valid if the
                        the first-percentile is not NULL";
                }
                description
                "Low percentile report";
```

```
            uses time-percentile-report;
          }

          container mid-percentile {
                when "/stamp/stamp-session-sender/"
                +"test-session[session-id]/"
                +"second-percentile != '0.00'" {
                        description
                        "Only valid if the
                        the first-percentile is not NULL";
                }
                description
                "Mid percentile report";
                uses time-percentile-report;
          }

          container high-percentile {
                when "/stamp/stamp-session-sender/"
                +"test-session[session-id]/"
                +"third-percentile != '0.00'" {
                        description
                        "Only valid if the
                        the first-percentile is not NULL";
                }
                description
                "High percentile report";
                uses time-percentile-report;
          }

          container two-way-loss {
            description
            "two way loss count and ratio result of
            the test session";
            uses packet-loss-statistics;
          }
          container one-way-loss-far-end {
            when "/stamp/stamp-session-sender/"
            +"test-session[session-id]/"
            +"test-session-reflector-mode = 'stateful'" {
                description
                "One-way statistic is only valid if the
                session-reflector is in stateful mode.";
            }
            description
            "one way loss count and ratio far-end of
            the test session";
            uses packet-loss-statistics;
          }
```

```
          container one-way-loss-near-end {
            when "/stamp/stamp-session-sender/"
            +"test-session[session-id]/"
            +"test-session-reflector-mode = 'stateful'" {
                description
                "One-way statistic is only valid if the
                session-reflector is in stateful mode.";
            }
            description
            "one way loss count and ratio near-end of
            the test session";
            uses packet-loss-statistics;
          }
        }

        list history-stats {
          key id;
          description
          "This container contains the results for the history
           Measurement Interval in a Measurement session ";
           leaf id {
             type uint32;
             description
             "The identifier for the Measurement Interval
             within this session";
           }
           leaf end-time {
             type yang:date-and-time;
             mandatory true;
             description
             "The time that the Measurement Interval ended";
           }
           leaf number-of-packets {
             type uint32;
             description
             "The overall number of UDP test packets to be
             transmitted by the sender for this test session";
           }

           leaf packet-padding-size {
             type uint32;
             default 27;
             description
             "Size of the Packet Padding. Suggested to run
             Path MTU Discovery to avoid packet fragmentation
             in IPv4 and packet blackholing in IPv6";
           }
```

```
          leaf interval  {
            type uint32;
            units microseconds;
           description
           "Time interval between transmission of two
           consecutive packets in the test session";
          }
          leaf duplicate-packets  {
            type uint32;
           description "Duplicate packets";
          }
          leaf reordered-packets  {
            type uint32;
           description "Reordered packets";
          }
          leaf loss-packets  {
            type uint32;
           description "Loss packets";
          }

          uses session-parameters;
          leaf dscp {
            type inet:dscp;
            description
            "The DSCP value that was placed in the header of
            STAMP UDP test packets by the Session-Sender.";
          }
          uses maintenance-statistics;

          container two-way-delay{
            description
            "two way delay result of the test session";
            uses delay-statistics;
          }
          container one-way-delay-far-end{
            description
            "one way delay far end of the test session";
            uses delay-statistics;
          }
          container one-way-delay-near-end{
            description
            "one way delay near end of the test session";
            uses delay-statistics;
          }
        }
      }
    }
```

```
    container stamp-session-refl-state {
      if-feature session-reflector;
      description
      "stamp Session-Reflector container for
      state data";
      leaf reflector-light-admin-status {
        type boolean;
        mandatory "true";
        description
        "Whether this network element is enabled to
        act as stamp Reflector";
      }

     list test-session-state {
       key "session-id";
       description
       "This structure is a container of test session
       managed objects";

       leaf session-id {
         type uint32;
         description "Session ID";
       }

       uses maintenance-statistics;
       uses session-parameters;
     }
   }
 }

 rpc stamp-sender-start {
   description
         "start the configured sender session";
   input {
         leaf session-id {
           type uint32;
           mandatory true;
           description
                 "The session to be started";
         }
   }
 }

 rpc stamp-sender-stop {
   description
         "stop the configured sender session";
   input {
         leaf session-id {
```

```
            type uint32;
            mandatory true;
            description
                  "The session to be stopped";
          }
    }
  }
}
```

<CODE ENDS>

4. IANA Considerations

This document registers a URI in the IETF XML registry [RFC3688]. Following the format in [RFC3688], the following registration is requested to be made.

URI: urn:ietf:params:xml:ns:yang:ietf-stamp

Registrant Contact: The IPPM WG of the IETF.

XML: N/A, the requested URI is an XML namespace.

This document registers a YANG module in the YANG Module Names registry [RFC6020].

name: ietf-stamp

namespace: urn:ietf:params:xml:ns:yang:ietf-stamp

prefix: stamp

reference: RFC XXXX

5. Security Considerations

The configuration, state, action data defined in this document may be accessed via the NETCONF protocol [RFC6241]. SSH [RFC6242] is mandatory secure transport that is the lowest NETCONF layer. The NETCONF access control model [RFC6536] provides means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

6. Normative References

[I-D.mirsky-ippm-stamp]
Mirsky, G. and G. Jun, "Simple Two-way Active Measurement Protocol", draft-mirsky-ippm-stamp-00 (work in progress), October 2017.

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

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <https://www.rfc-editor.org/info/rfc6536>.

[RFC7750] Hedin, J., Mirsky, G., and S. Baillargeon, "Differentiated Service Code Point and Explicit Congestion Notification Monitoring in the Two-Way Active Measurement Protocol (TWAMP)", RFC 7750, DOI 10.17487/RFC7750, February 2016, <https://www.rfc-editor.org/info/rfc7750>.

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

Appendix A. Acknowledgements

Authors recognize and appreciate valuable comments providen by Adrian Pan.

Authors' Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Xiao Min
ZTE Corp.

Email: xiao.min2@zte.com.cn

Wei S Luo
Ericsson

Email: wei.s.luo@ericsson.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 13, 2019

T. Mizrahi
Huawei Network.IO Innovation Lab
C. Arad

G. Fioccola
Huawei Technologies
M. Cociglio
Telecom Italia
M. Chen
L. Zheng
Huawei Technologies
G. Mirsky
ZTE Corp.

October 10, 2018

Compact Alternate Marking Methods for Passive and Hybrid Performance Monitoring
draft-mizrahi-ippm-compact-alternate-marking-03

Abstract

This memo introduces new alternate marking methods that require a compact overhead of either a single bit per packet, or zero bits per packet. This memo also presents a summary of alternate marking methods, and discusses the tradeoffs among them. The target audience of this document is network protocol designers; this document is intended to help protocol designers choose the best alternate marking method(s) based on the protocol's constraints and requirements.

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 https://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 13, 2019.

Table of Contents

1. Introduction

1.1. Background

Alternate marking, defined in [RFC8321], is a method for measuring packet loss, packet delay, and packet delay variation. Typical delay measurement protocols require the two measurement points (MPs) to exchange timestamped test packets. In contrast, the alternate marking method does not require control packets to be exchanged. Instead, every data packet carries a color indicator, which divides the traffic into consecutive blocks of packets.

The color value is toggled periodically, as illustrated in Figure 1.

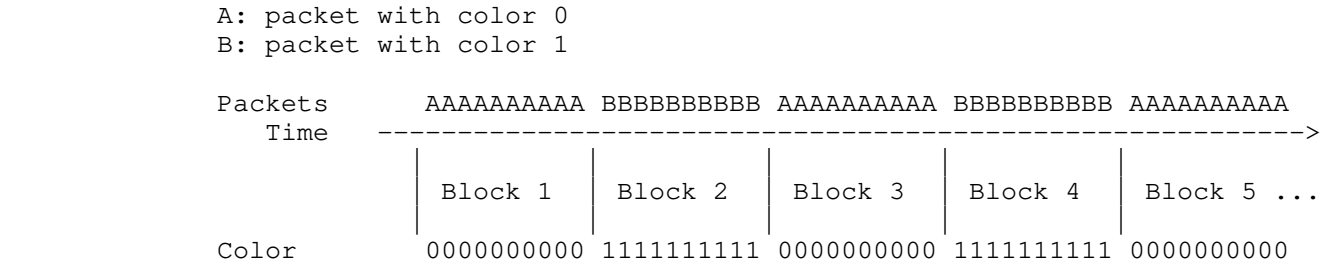


Figure 1: Alternate marking: packets are monitored on a per-color basis.

Alternate marking is used between two MPs, the initiating MP, and the monitoring MP. The initiating MP incorporates the marking field into en-route packets, allowing the monitoring MP to use the marking field in order to bind each packet to the corresponding block.

Each of the MPs maintains two counters, one per color. At the end of each block the counter values can be collected by a central management system, and analyzed; the packet loss can be computed by comparing the counter values of the two MPs.

When using alternate marking delay measurement can be performed in one of three ways (as per [RFC8321]):

- Single marking using the first packet: in this method each packet uses a single marking bit, used as a color indicator. The first packet of each block is used by both MPs as a reference for delay measurement. The timestamp of this packet is measured by the two measurement points, and can be collected by the mangement system from each of the measurement points, which can compute the path delay by comparing the two timestamps. The drawback of this

approach is that it is not accurate when packets arrive out-of-order, as the two MPs may have a different view of which packet was the first in the block.

- Single marking using the mean delay: as in the previous method, each packet uses a single marking method, indicating the color. Each of the MPs computes the average packet timestamp of each block. The management system can then compute the delay by comparing the average times of the two MPs. The drawback of this approach is that it may be computationally heavy, or difficult to implement at the data plane.

- Double marking: each packet uses two marking bits. One bit is used as a color indicator, and one is used as a timestamping indicator. This method resolves the drawbacks raised for the two previous methods, at the expense of an extra bit in the packet header.

The double marking method is the most straightforward approach. It allows for accurate measurement without incurring expensive computational load. However, in some cases allocating two bits for passive measurement is not possible. For example, if alternate marking is implemented over IPv4, allocating 2 marking bits in the IPv4 header is challenging, as every bit in the 20-octet header is costly; one of the possible approaches discussed in [RFC8321] is to reserve one or two bits from the DSCP field for remarking. In this case every marking bit comes at the expense of reducing the DSCP range by a factor of two.

1.2. The Scope of This Document

This memo extends the marking methods of [RFC8321], and introduces methods that require a single marking bit, or zero marking bits.

Two single-bit marking methods are proposed, multiplexed marking and pulse marking. In multiplexed marking the color indicator and the timestamp indicator are multiplexed into a single bit, providing the advantages of the double marking method while using a single bit in the packet header. In pulse marking both delay and loss measurement are triggered by a 'pulse' value in a single marking field.

This document also discusses zero-bit marking methods that leverage well-known hash-based selection approaches ([RFC5474], [RFC5475]).

Alternate marking is discussed in this memo as a single-bit or a two-bit marking method. However, these methods can similarly be applied to larger fields, such as an IPv6 Flow Label or an MPLS Label; single-bit marking can be applied using two reserved values, and two-

bit marking can be applied using four reserved values. Marking based on reserved values is further discussed in this document, including its application to MPLS and IPv6.

Finally, this memo summarizes the alternate marking methods, and discusses the tradeoffs among them. It is expected that different network protocols will have different constraints, and therefore may choose to use different alternate marking methods. In some cases it may be preferable to support more than one marking method; in this case the particular marking method may be signaled through the control plane.

2. Terminology

2.1. Requirements Language

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

2.2. Abbreviations

The following abbreviations are used in this document:

DSCP	Differentiated Services Code Point
DM	Delay Measurement
LM	Loss Measurement
LSP	Label Switched Path
MP	Measurement Point
MPLS	Multiprotocol Label Switching
SFL	Synonymous Flow Label [I-D.ietf-mpls-sfl-framework]

3. Marking Abstractions

The marking methods that were discussed in Section 1, as well as the methods introduced in this document, use two basic abstractions, pulse detection, and step detection.

The common thread along the various marking methods is that one or two marking bits are used by the MPs to signal a measurement event. The value of the marking bit indicates when the event takes place, in one of two ways:

Pulse An event is detected when the value of the marking bit is toggled in a single packet.

Step An event is detected when the value of the marking bit is toggled, and remains at the new value.

The double marking method (Section 1) uses pulse-based detection for DM, and step-based detection for LM.

Pulse-based detection affects the processing of a single packet; the packet that indicates the pulse is processed differently than the packets around it. For example, in the double marking method, the marked packet is timestamped for DM, without affecting the packets before or after it. Note that if the marked packet is lost, no pulse is detected, yielding a missing measurement (see Figure 2).

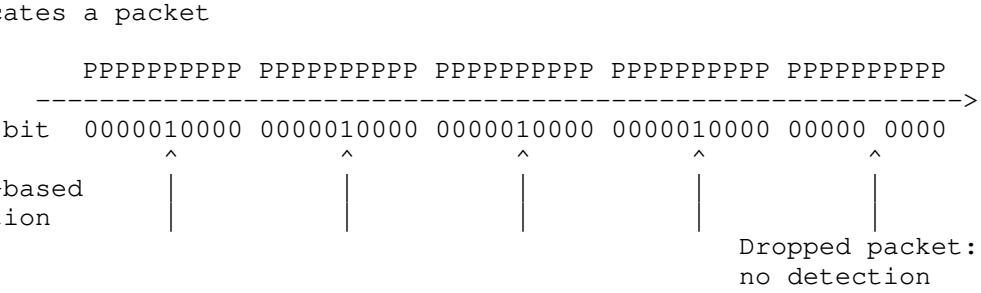


Figure 2: Pulse-based Detection.

In step-based detection the event is detected by observing a value change in stream of packets. Specifically, when the step approach is used for LM (as in the double marking method), two counters are used per flow; each MP decides which counter to use based on the value of the marking bit. Thus, the step-based approach allows accurate counting even when packets arrive out-of-order (see Figure 3). When the step approach is used for DM (e.g., single marking using the first packet), out-of-order causes the delay measurement to be false, without any indication to the management system.

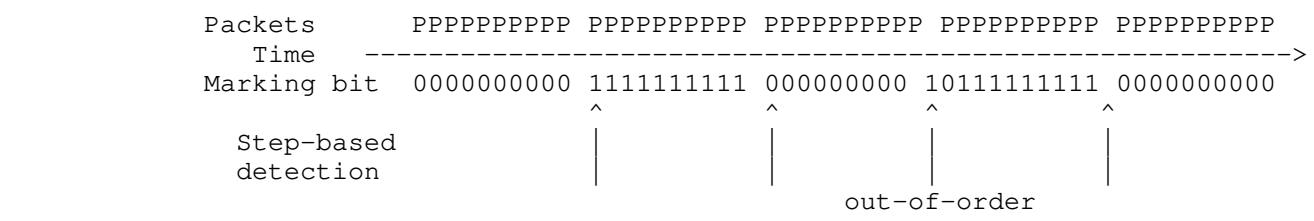


Figure 3: Step-based Detection.

4. Double Marking

The two-bit marking method of [RFC8321] uses two marking bits: a color indicator, and a delay measurement indicator. The color bit is used for step-based LM, while the delay bit is used as a pulse-based DM trigger. This double marking approach is the most straightforward of the approaches discussed in this memo, as it allows accurate measurement, it is resilient to out-of-order delivery, and is relatively simple to implement. The main drawback is that it requires two bits, which are not always available.

Figure 4 illustrates the double marking method: each block of packets includes a packet that is marked for timestamping, and therefore has its delay bit set.

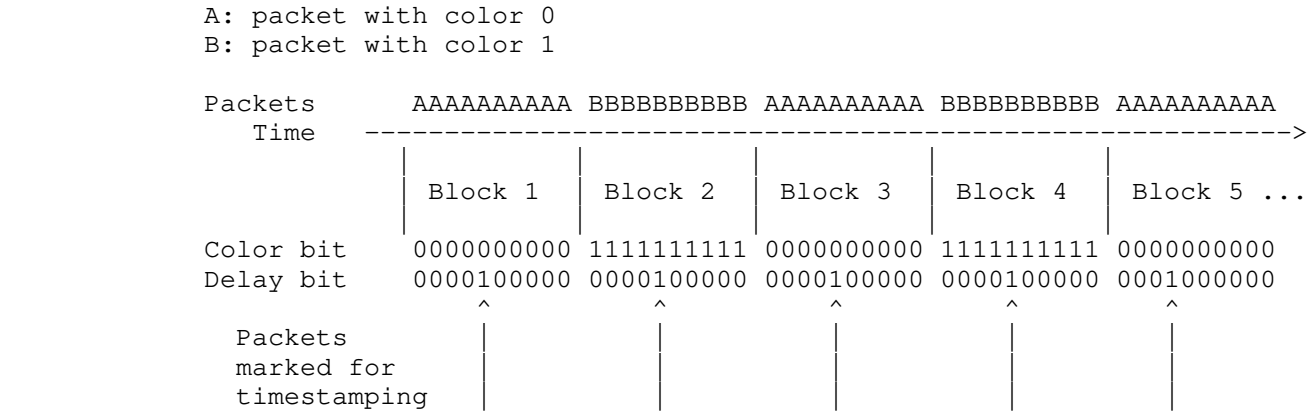


Figure 4: The double marking method.

5. Single-bit Marking

5.1. Single Marking Using the First Packet

This method uses a single marking bit that indicates the color, as described in [RFC8321]. Both LM and DM are implemented using a step-based approach; LM is implemented using two color-based counters per flow. The first packet of every period is used by the two MPs as the reference for measuring the delay. As denoted above, the delay computed in this method may be erroneous when packets are delivered out-of-order.

A: packet with color 0
B: packet with color 1

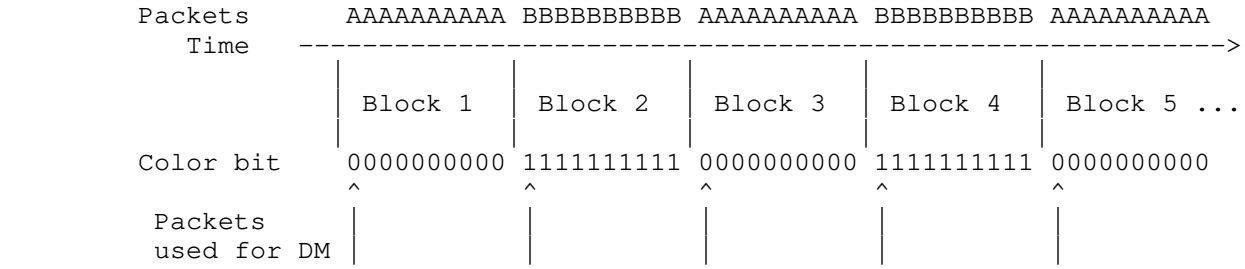


Figure 5: Single marking using the first packet of the block.

5.2. Single Marking using the Mean Delay

As in the first-packet approach, in the mean delay approach ([RFC8321]) a single marking bit is used to indicate the color, enabling step-based loss measurement. Delay is measured in each period by averaging the measured delay over all the packets in the period. As discussed above, this approach is not sensitive to out-of-order delivery, but may be heavy from a computational perspective.

5.3. Single Marking using a Multiplexed Marking Bit

5.3.1. Overview

This section introduces a method that uses a single marking bit that serves two purposes: a color indicator, and a timestamp indicator. The double marking method that was discussed in the previous section uses two 1-bit values: a color indicator C, and a timestamp indicator T. The multiplexed marking bit, denoted by M, is an exclusive or between these two values: M = C XOR T.

An example of the use of the multiplexed marking bit is depicted in Figure 6. The example considers two routers, R1 and R2, that use the multiplexed bit method to measure traffic from R1 to R2. In each block R1 designates one of the packets for delay measurement. In each of these designated packets the value of the multiplexed bit is reversed compared to the other packets in the same block, allowing R2 to distinguish the designated packets from the other packets.

A: packet with color 0
B: packet with color 1

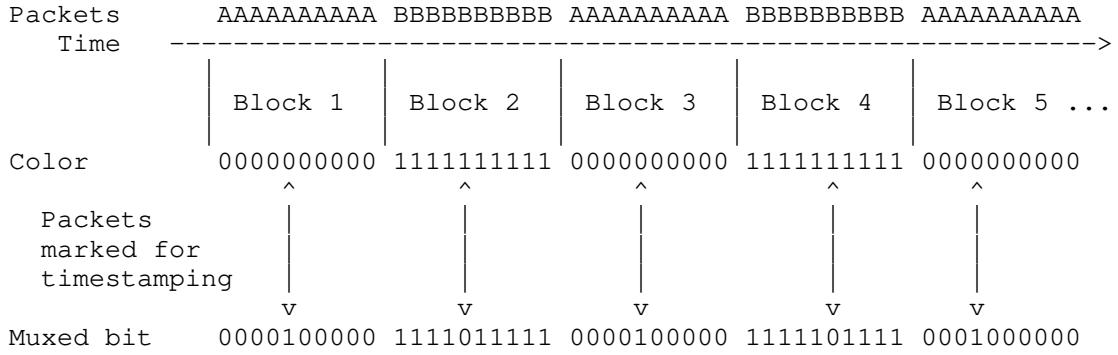


Figure 6: Alternate marking with multiplexed bit.

5.3.2. Timing and Synchronization Aspects

It is assumed that all MPs are synchronized to a common reference time with an accuracy of +/- A/2. Thus, the difference between the clock values of any two MPs is bounded by A. Clocks can be synchronized for example using NTP [RFC5905], PTP [IEEE1588], or by other means. The common reference time is used for dividing the time domain into equal-sized measurement periods, such that all packets forwarded during a measurement period have the same color, and consecutive periods have alternating colors.

The single marking bit incorporates two multiplexed values. From the monitoring MP's perspective, the two values are Time-Division Multiplexed (TDM), as depicted in Figure 7. It is assumed that the start time of every measurement period is known to both the initiating MP and the monitoring MP. If the measurement period is L, then during the first and the last L/4 time units of each block the marking bit is interpreted by the monitoring MP as a color indicator. During the middle part of the block, the marking bit is interpreted as a timestamp indicator; if the value of this bit is different than

the color value, the corresponding packet is used as a reference for delay measurement.

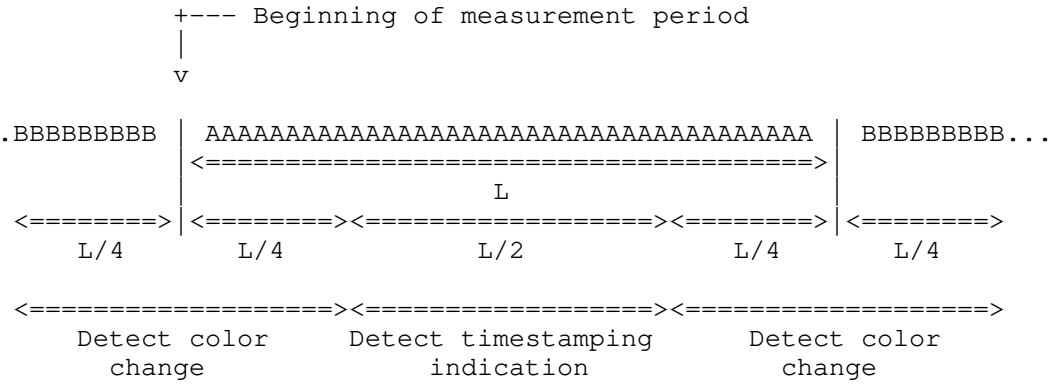


Figure 7: Multiplexed marking field interpretation at the receiving measurement point.

In order to prevent ambiguity in the receiver's interpretation of the marking field, the initiating MP is permitted to set the timestamp indication only during a specific interval, as depicted in Figure 8. Since the receiver is willing to receive the timestamp indication during the middle L/2 time units of the block, the sender refrains from sending the timestamp indication during a guardband interval of d time units at the beginning and end of the L/2-period.

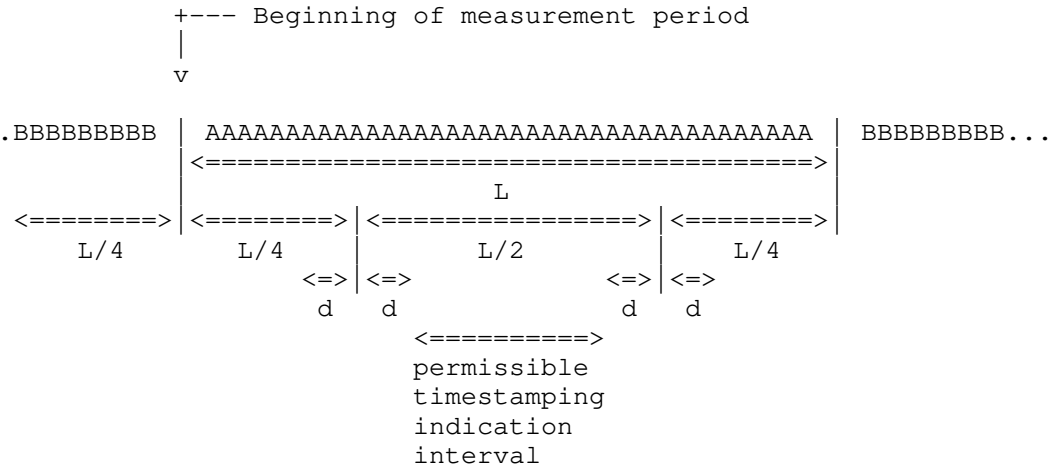


Figure 8: A time domain view.

The guardband d is given by d = A + D_max - D_min, where A is the clock accuracy, D_max is an upper bound on the network delay between the MPs, and D_min is a lower bound on the delay. It is straightforward from Figure 8 that d < L/4 must be satisfied. The latter implies a minimal requirement on the synchronization accuracy.

All MPs must be synchronized to the same reference time with an accuracy of +/- L/8. Depending on the system topology, in some systems the accuracy requirement will be even more stringent, subject to d < L/4. Note that the accuracy requirement of the conventional alternate marking method [RFC8321] is +/- L/2, while the multiplexed marking method requires an accuracy of +/- L/8.

Note that we assume that the middle L/2-period is designated as the timestamp indication period, allowing a sufficiently long guardband between the transitions. However, a system may be configured to use a longer timestamp indication period or a shorter one, if it is guaranteed that the synchronization accuracy meets the guardband requirements (i.e., the constraints on d).

5.4. Pulse Marking

Pulse marking uses a single marking bit that is used as a trigger for both LM and DM. In this method the two MPs maintain a single per-flow counter for LM, in contrast to the color-based methods which require two counters per flow. In each block one of the packets is marked. The marked packet triggers two actions in each of MPs:

- The timestamp is captured for DM.
- The value of the counter is captured for LM.

In each period, each of the MPs exports the timestamp and counter-stamp to the management system, which can then compute the loss and delay in that period. It should be noted that as in [RFC8321], if the length of the measurement period is L time units, then all network devices must be synchronized to the same clock reference with an accuracy of +/- L/2 time units.

The pulse marking approach is illustrated in Figure 9. Since both LM and DM use a pulse-based trigger, if the marked packet is lost then no measurement is available in this period. Moreover, the LM accuracy may be affected by out-of-order delivery.

P: packet - all packets have the same color

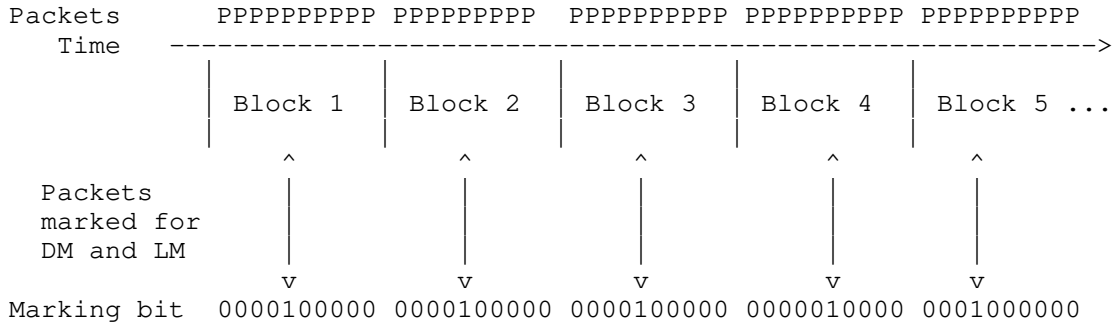


Figure 9: Pulse marking method.

6. Zero Marking Hashed

6.1. Hash-based Sampling

Hash based selection [RFC5475] is a well-known method for sampling a subset of packets. As defined in [RFC5475]:

> A Hash Function h maps the Packet Content c, or some portion of it, onto a Hash Range R. The packet is selected if h(c) is an element of S, which is a subset of R called the Hash Selection Range.

Hash-based selection can be leveraged as a marking method, allowing a zero-bit marking approach. Specifically, the pulse and step abstractions can be implemented using hashed selection:

- Hashed pulse-based trigger: in this approach, a packet is selected if h(c) is an element of S, which is a strict subset of the hash range R. When $|S| << |R|$, the average sampling period is long, reducing the probability of ambiguity between consecutive packets. $|S|$ and $|R|$ denote the number of elements in S and R, respectively.

- Hashed step-based trigger: the hash values of a given traffic flow are said to be monotonically increasing if for two packets p1 and p2, if p1 is sent before p2 then $h(p1) <= h(p2)$. If it is guaranteed that the hash values of a flow are monotonically increasing, then a step-based approach can be used on the range R. For example, in an IPv4 flow the Identification field can be used as the hash value of each packet. Since the Identification field is monotonically increasing, the step-based trigger can be

implemented using consecutive ranges of the Identification value. For example, the fourth bit of the Identification field is toggled every 8 packets. Thus, a possible hash function simply takes the fourth bit of the Identification field as the hash value. This hash value is toggled every 8 packets, simulating the alternate marking behavior of Section 4.

Note that as opposed to the double marking and single marking methods, hashed sampling is not based on fixed time intervals, as the duration between sampled packets depends only on the hash value.

It is also important to note that all methods that use hash-based marking require the hash function and the set S to be configured consistently across the MPs.

6.1.1. Hashed Pulse Marking

In this approach a hash is computed over the packet content, and both LM and DM are triggered based on the pulse-based trigger (Section 6.1). A pulse is detected when the hash value h(c) is equal to one of the values in S. The hash function h and the set S determine the probability (or frequency) of the pulse event.

6.1.2. Hashed Step Marking

As in the previous approach, hashed step marking also uses a hash that is computed over the packet content. In this approach DM is performed using a pulse-based trigger, whereas the LM trigger is step-based (Section 6.1). The main drawback of this method is that the step-based trigger is possible only under the assumption that the hash function is monotonically increasing, which is not necessarily possible in all cases. Specifically, a measured flow is not necessarily an IPv4 5-tuple. For example, a measured flow may include multiple IPv4 5-tuple flows, and in this case the Identification field is not monotonically increasing.

7. Single Marking Hashed

Mixed hashed marking combines the single marking approach with hash-based sampling. A single marking bit is used in the packet header as a color indicator, while a hash-based pulse is used to trigger DM. Although this method requires a single bit, it is described in this section as it is closely related to the other hash-based methods that require zero marking bits.

The hash-based selection for DM can be applied in one of two possible approaches: the basic approach, and the dynamic approach. In the basic approach, packets forwarded between two MPs, MP1 and MP2, are

selected using a hash function, as described above. One of the challenges is that the frequency of the sampled packets may vary considerably, making it difficult for the management system to correlate samples from the two MPs. Thus, the dynamic approach can be used.

In the dynamic hash-based sampling, alternate marking is used to create divide time into periods, so that hash-based samples are divided into batches, allowing to anchor the selected samples to their period. Moreover, by dynamically adapting the length of the hash value, the number of samples is bounded in each marking period. This can be realized by choosing first the maximum number of samples (NMAX) to be used with the initial hash length. The algorithm starts with only few hash bits, that permit to select a greater percentage of packets (e.g. with 1 bit of hash half of the packets are sampled). When the number of selected packets reaches NMAX, a hashing bit is added. As a consequence, the sampling proceeds at half of the original rate and the packets already selected that do not match the new hash are discarded. This step can be repeated iteratively. It is assumed that each sample includes the timestamp (used for DM) and the hash value, allowing the management system to match the samples received from the two MPs.

The dynamic process statistically converges at the end of a marking period and the number of selected samples beyond the initial NMAX samples mentioned above is between NMAX/2 and NMAX. Therefore, the dynamic approach paces the sampling rate, allowing to bound the number of sampled packets per sampling period.

8. Summary of Marking Methods

This section summarizes the marking methods described in this memo. Each row in the table of Figure 10 represents a marking method. For each method the table specifies the number of bits required in the header, the number of counters per flow for LM, the methods used for LM and DM (pulse or step), and also the resilience to disturbances.

Method	# of bits	# of counters	LM Method	DM Method	Resilience to Reordering		Resilience to Packet drops	
					LM	DM	LM	DM
Single marking - 1st packet	1	2	Step	Step	+	--	+	--
Single marking - mean delay	1	2	Step	Mean	+	+	+	-
Double marking	2	2	Step	Pulse	+	+	+	=
Single marking multiplexed	1	2	Step	Pulse	+	+	+	=
Pulse marking	1	1	Pulse	Pulse	--	+	-	=
Zero marking hashed	0	1 (2)	Hashed pulse (step)	Hashed pulse	-- (-)	+	-	+
Single marking hashed	1	2	Step	Hashed pulse	+	+	+	+

+ Accurate measurement.
= Invalidate only if a measured packet is lost (detectable)
- No measurement in case of disturbance (detectable).
-- False measurement in case of disturbance (not detectable).

Figure 10: Detailed Summary of Marking Methods

In the context of this comparison two possible disturbances are considered: out-of-order delivery, and packet drops. Generally speaking, pulse based methods are sensitive to packet drops, since if the marked packet is dropped no measurement is recorded in the current period. Notably, a missing measurement is detectable by the management system, and is not as severe as a false measurement. Step-based triggers are generally resilient to out-of-order delivery for LM, but are not resilient to out-of-order delivery for DM. Notably, a step-based trigger may yield a false delay measurement when packets are delivered out-of-order, and this inaccuracy is not detectable.

As mentioned above, the double marking method is the most straightforward approach, and is resilient to most of the

disturbances that were analyzed. Its obvious drawback is that it requires two marking bits.

Several single marking methods are discussed in this memo. In this case there is no clear verdict which method is the optimal one. The first packet method may be simple to implement, but may present erroneous delay measurements in case of dropped or reordered packets. Arguably, the mean delay approach and the multiplexed approach may be more difficult to implement (depending on the underlying platform), but are more resilient to the disturbances that were considered here. Note that the computational complexity of the mean delay approach can be reduced by combining it with a hashed approach, i.e., by computing the mean delay over a hash-based subset of the packets. The pulse marking method requires only a single counter per flow, while the other methods require two counters per flow.

The hash-based sampling approaches reduce the overhead to zero bits, which is a significant advantage. However, the sampling period in these approaches is not associated with a fixed time interval. Therefore, in some cases adjacent packets may be selected for the sampling, potentially causing measurement errors. Furthermore, when the traffic rate is low, measurements may become signifcantly infrequent.

It should be noted that most of the marking methods that were presented in this memo are intended for point-to-point measurements, e.g., from MP1 to MP2 in Figure 11. In point-to-multipoint measurements, the mean delay method can be used to measure the loss and delay of the entire point-to-multipoint flow (which includes all the traffic from MP3 to either MP4 or MP5), while other methods such as double marking can be used to measure the point-to-point performance, for example from MP3 to MP5. Alternate marking in multipoint scenarios is discussed in detail in [I-D.fioccola-ippm-multipoint-alt-mark].

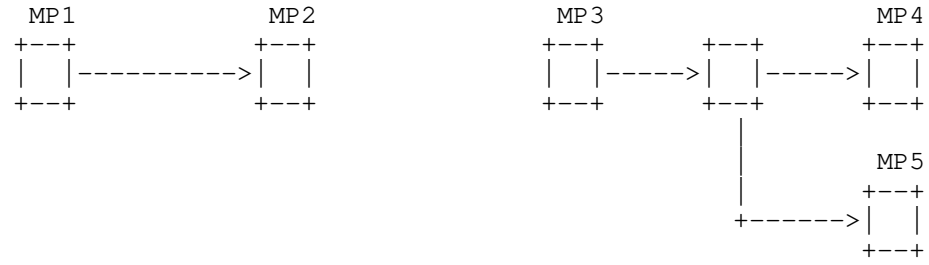


Figure 11: Point-to-point and point-to-multipoint measurements.

It is clear from the previous table that packet loss measurement can be considered resilient to both reordering and packet drops if at least one bit is used with a step-based approach. Thus, since the packet loss can be considered obvious, the previous table can be simplified into Figure 12, where only the characteristics of delay measurements are highlighted, along with multipoint-to-multipoint delay measurement compatibility (refer to [I-D.fioccola-ippm-multipoint-alt-mark] for more details).

Marking Method	# of bits	LM on All Packets	DM Resilience to Reordering	DM Resilience to Packet drops	DM Multipoint compatible
Single marking - 1st packet	1	Yes	--	-	No
Single marking - mean delay	1	Yes	+	-	Yes
Double marking	2	Yes	+	=	No
Single marking multiplexed	1	Yes	+	=	No
Pulse marking	1	No	+	=	No
Zero marking hashed	0	No	+	+	Yes
Single marking hashed	1	Yes	+	+	Yes

+ Accurate measurement.
= Invalidate only if a measured packet is lost (detectable)
- No measurement in case of disturbance (detectable).
-- False measurement in case of disturbance (not detectable).

Figure 12: Summary of Marking Methods: focus on Delay Measurement

In the context of delay measurement, both zero marking hashed and single marking hashed are resilient to packet drops. Using double marking it could also be possible to perform an accurate measurement in case of packet drops, as long as the packet that is marked for DM is not dropped.

The single marking hashed method seems the most complete approach, especially because it is also compatible with multipoint-to-multipoint measurements.

9. Alternate Marking using Reserved Values

As mentioned in Section 1, a marking bit is not necessarily a single bit, but may be implemented by using two well-known values in one of the header fields. Similarly, two-bit marking can be implemented using four reserved values.

A notable example is MPLS Synonymous Flow Labels (SFL), as defined in [I-D.ietf-mpls-rfc6374-sfl]. Two MPLS Label values can be used to indicate the two colors of a given LSP: the original Label value, and an SFL value. A similar approach can be applied to IPv6 using the Flow Label field.

The following example illustrates how alternate marking can be implemented using reserved values. The bit multiplexing approach of Section 5.3 is applicable not only to single-bit color indicators, but also to two-value indicators; instead of using a single bit that is toggled between '0' and '1', two values of the indicator field, U and W, can be used in the same manner, allowing both loss and delay measurement to be performed using only two reserved values. Thus, the multiplexing approach of Figure 6 can be illustrated more generally with two values, U and W, as depicted in Figure 13.

A: packet with color 0
B: packet with color 1

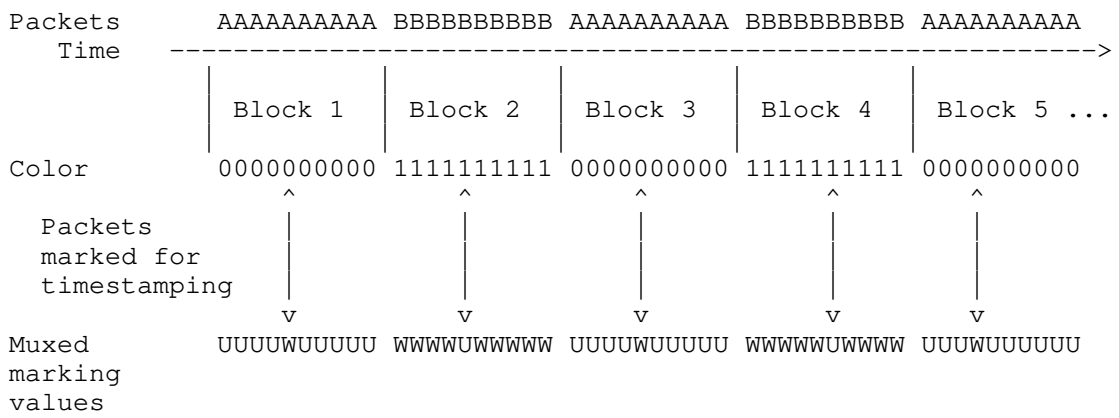


Figure 13: Alternate marking with two multiplexed marking values, U and W.

10. IANA Considerations

This memo includes no requests from IANA.

11. Security Considerations

The security considerations of the alternate marking method are discussed in [RFC8321]. The analysis of Section 8 emphasizes the sensitivity of some of the alternate marking methods to packet drops and to packet reordering. Thus, a malicious attacker may attempt to tamper with the measurements by either selectively dropping packets, or by selectively reordering specific packets. The multiplexed marking method Section 5.3 that is defined in this document requires slightly more stringent synchronization than the conventional marking method, potentially making the method more vulnerable to attacks on the time synchronization protocol. A detailed discussion about the threats against time protocols and how to mitigate them is presented in [RFC7384].

12. References

12.1. Normative References

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

[RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <https://www.rfc-editor.org/info/rfc8321>.

12.2. Informative References

[I-D.fioccola-ippm-multipoint-alt-mark] Fioccola, G., Cociglio, M., Sapio, A., and R. Sisto, "Multipoint Alternate Marking method for passive and hybrid performance monitoring", draft-fioccola-ippm-multipoint-alt-mark-04 (work in progress), June 2018.

[I-D.ietf-mpls-rfc6374-sfl] Bryant, S., Chen, M., Li, Z., Swallow, G., Sivabalan, S., Mirsky, G., and G. Fioccola, "RFC6374 Synonymous Flow Labels", draft-ietf-mpls-rfc6374-sfl-02 (work in progress), June 2018.

[I-D.ietf-mpls-sfl-framework]
Bryant, S., Chen, M., Li, Z., Swallow, G., Sivabalan, S., and G. Mirsky, "Synonymous Flow Label Framework", draft-ietf-mpls-sfl-framework-03 (work in progress), June 2018.

[IEEE1588]
IEEE, "IEEE 1588 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Version 2", 2008.

[RFC5474] Duffield, N., Ed., Chiou, D., Claise, B., Greenberg, A., Grossglauser, M., and J. Rexford, "A Framework for Packet Selection and Reporting", RFC 5474, DOI 10.17487/RFC5474, March 2009, <https://www.rfc-editor.org/info/rfc5474>.

[RFC5475] Zseby, T., Molina, M., Duffield, N., Niccolini, S., and F. Raspall, "Sampling and Filtering Techniques for IP Packet Selection", RFC 5475, DOI 10.17487/RFC5475, March 2009, <https://www.rfc-editor.org/info/rfc5475>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <https://www.rfc-editor.org/info/rfc7384>.

Authors' Addresses

Tal Mizrahi
Huawei Network.IO Innovation Lab
Israel

Email: tal.mizrahi.phd@gmail.com

Carmi Arad

Email: carmi.arad@gmail.com

Giuseppe Fioccola
Huawei Technologies

Email: giuseppe.fioccola@huawei.com

Mauro Cociglio
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: mauro.cociglio@telecomitalia.it

Mach(Guoyi) Chen
Huawei Technologies

Email: mach.chen@huawei.com

Lianshu Zheng
Huawei Technologies

Email: vero.zheng@huawei.com

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Network Working Group
Internet-Draft
Updates: 4656 and 5357 (if approved)
Intended status: Standards Track
Expires: May 17, 2018

A. Morton, Ed.
AT&T Labs
G. Mirsky, Ed.
ZTE Corp.

November 13, 2017

OWAMP and TWAMP Well-Known Port Assignments
draft-morton-ippm-port-twamp-test-02

Abstract

This memo explains the motivation and describes the re-assignment of well-known ports for the OWAMP and TWAMP protocols for control and measurement, and clarifies the meaning and composition of these standards track protocol names for the industry.

The memo updates RFC 4656 and RFC 5357, in terms of the UDP well-known port assignments.

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 https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on May 17, 2018.

Table of Contents

1. Introduction

The IETF IP Performance Metrics (IPPM) working group first developed the One-Way Active Measurement Protocol, OWAMP, specified in [RFC4656]. Further protocol development to support testing resulted in the Two-Way Active Measurement Protocol, TWAMP, specified in [RFC5357].

Both OWAMP and TWAMP require the implementation of a control and mode negotiation protocol (OWAMP-Control and TWAMP-Control) which employs the reliable transport services of TCP (including security configuration and key derivation). The control protocols arrange for the configuration and management of test sessions using the associated test protocol (OWAMP-Test or TWAMP-Test) on UDP transport.

This memo recognizes the value of assigning a well-known UDP port to the *-Test protocols, and that this goal can easily be arranged through port re-assignments.

2. Requirements Language

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

[RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Scope

The scope of this memo is to re-allocate well-known ports for the UDP Test protocols that compose necessary parts of their respective standards track protocols, OWAMP and TWAMP, along with clarifications of the complete protocol composition for the industry.

The memo updates [RFC4656] and [RFC5357], in terms of the UDP well-known port assignments.

4. Definitions

This section defines key terms and clarifies the required composition of the OWAMP and TWAMP standards-track protocols.

OWAMP-Control is the protocol defined in Section 3 of [RFC4656].

OWAMP-Test is the protocol defined in Section 4 of [RFC4656].

OWAMP is described in a direct quote from Section 1.1 of[RFC4656]: "OWAMP actually consists of two inter-related protocols: OWAMP-Control and OWAMP-Test." A similar sentence appears in Section 2 of [RFC4656]. Since the consensus of many dictionary definitions of "consist" is "composed or made up of", implementation of both OWAMP-Control and OWAMP-Test are REQUIRED for standards-track OWAMP specified in [RFC4656].

TWAMP-Control is the protocol defined in Section 3 of [RFC5357].

TWAMP-Test is the protocol defined in Section 4 of [RFC5357].

TWAMP is described in a direct quote from Section 1.1 of [RFC5357]: "Similar to OWAMP [RFC4656], TWAMP consists of two inter-related protocols: TWAMP-Control and TWAMP-Test." Since the consensus of many dictionary definitions of "consist" is "composed or made up of", implementation of both TWAMP-Control and TWAMP-Test are REQUIRED for standards-track TWAMP specified in [RFC5357].

TWAMP Light is an idea described in Informative Appendix I of [RFC5357], and includes an un-specified control protocol (possibly communicating through non-standard means) combined with the TWAMP-Test protocol. The TWAMP Light idea was relegated to the Appendix because it failed to meet the requirements for IETF protocols (there are no specifications for negotiating this form of

operation, and no specifications for mandatory-to-implement security features), as described in the references below:

- Lars Eggert's Area Director review [LarsAD], where he pointed out that having two variants of TWAMP, Light and Complete (called standards track TWAMP here), required a protocol mechanism to negotiate which variant will be used. See Lars' comment on Sec 5.2. The working group consensus was to place the TWAMP Light description in Appendix I, and to refer to the Appendix only as an "incremental path to adopting TWAMP, by implementing the TWAMP-Test protocol first".

- Tim Polk's DISCUSS Ballot, which points out that TWAMP Light was an incomplete specification because the key required for authenticated and encrypted modes depended on the TWAMP-Control Session key. See Tim's DISCUSS on 2008-07-16 [TimDISCUSS]. Additional requirement statements were added in the Appendix to address Tim's DISCUSS Ballot (see the last three paragraphs of Appendix I in [RFC5357]).

Since the idea of TWAMP Light clearly includes the TWAMP-Test component of TWAMP, it is considered reasonable for future systems to use the TWAMP-Test well-known UDP port (whose re-allocated assignment is requested here). Clearly, the TWAMP Light idea envisions many components and communication capabilities beyond TWAMP-Test (implementing the security requirements, for example), otherwise the Appendix would be one sentence long (equivocating TWAMP Light with TWAMP-Test only).

5. New Well-Known Ports

Originally, both TCP and UDP well-known ports were assigned to the control protocols that are essential components of standards track OWAMP and TWAMP.

Since OWAMP-Control and TWAMP-Control require TCP transport, they cannot make use of the UDP ports which were originally assigned. However, test sessions using OWAMP-Test or TWAMP-Test operate on UDP transport.

This memo requests re-assignment of the UDP well-known port from the Control protocol to the Test protocol (see the IANA Considerations Section 7). Use of this UDP port is OPTIONAL in standards-track OWAMP and TWAMP. It may simplify some operations to have a well-known port available for the Test protocols, or for future specifications involving TWAMP-Test to use this port as a default port.

5.1. Impact on TWAMP-Control Protocol

Section 3.5 [RFC5357] describes the detailed process of negotiating the Receiver Port number, on which the TWAMP Session-Reflector will send and receive TWAMP-Test packets. The Control-Client, acting on behalf of the Session-Sender, proposes the Receiver port number from the Dynamic Port range [RFC6335]:

> "The Receiver Port is the desired UDP port to which TWAMP-Test packets will be sent by the Session-Sender (the port where the Session-Reflector is asked to receive test packets). The Receiver Port is also the UDP port from which TWAMP-Test packets will be sent by the Session-Reflector (the Session-Reflector will use the same UDP port to send and receive packets)."

It is possible that the proposed Receiver Port may be not available, e.g., the port is in use by another test session or another application. In this case:

> "... the Server at the Session-Reflector MAY suggest an alternate and available port for this session in the Port field. The Control-Client either accepts the alternate port, or composes a new Session-Request message with suitable parameters. Otherwise, the Server uses the Accept field to convey other forms of session rejection or failure to the Control Client and MUST NOT suggest an alternate port; in this case, the Port field MUST be set to zero."

A Control Client that supports use of the allocated TWAMP-Test Receiver Port Section 7 MAY request to use that port number in the Request-TW-Session Command. If the Server does not support the allocated TWAMP-Test Receiver Port, then it sends an alternate port number in the Accept-Session message with Accept field = 0. Thus the deployment of the allocated TWAMP Receiver Port number is backward compatible with existing TWAMP-Control solutions that are based on [RFC5357]. Of course, use of a UDP port number chosen from the Dynamic Port range [RFC6335] will help to avoid the situation when the Control-Client or Server finds the proposed port being already in use.

5.2. Impact on OWAMP-Control Protocol

As described above, an OWAMP Control Client that supports use of the allocated OWAMP-Test Receiver Port Section 7 MAY request to use that port number in the Request-Session Command. If the Server does not support the allocated OWAMP-Test Receiver Port (or does not have the port available), then it sends an alternate port number in the Accept-Session message with Accept field = 0. Further exchanges proceed as already specified.

5.3. Impact on OWAMP/TWAMP-Test Protocols

OWAMP/TWAMP-Test may be used to measure IP performance metrics in an Equal Cost Multipath (ECMP) environment. Though algorithms to balance IP flows among available paths have not been standardized, the most common is the five-tuple that uses destination IP address, source IP address, protocol type, destination port number, and source port number. When attempting to monitor different paths in ECMP network, it is sufficient to vary only one of five parameters, e.g. the source port number. Thus, there will be no negative impact on ability to arrange concurrent OWAMP/TWAMP test sessions between the same test points to monitor different paths in the ECMP network when using the re-allocated UDP port number as the Receiver Port, as use of the port is optional.

6. Security Considerations

The security considerations that apply to any active measurement of live paths are relevant here as well (see [RFC4656] and [RFC5357]).

When considering privacy of those involved in measurement or those whose traffic is measured, the sensitive information available to potential observers is greatly reduced when using active techniques which are within this scope of work. Passive observations of user traffic for measurement purposes raise many privacy issues. We refer the reader to the security and privacy considerations described in the Large Scale Measurement of Broadband Performance (LMAP) Framework [RFC7594], which covers both active and passive techniques.

The registered UDP port as the Receiver Port for OWAMP/TWAMP-Test could become a target of denial-of-service (DoS) or used to aid man-in-the-middle (MITM) attacks. To improve protection from the DoS following methods are recommended:

- filtering access to the OWAMP/TWAMP Receiver Port by access list;
- using a non-globally routable IP address for the OWAMP/TWAMP Session-Reflector address.

A MITM attack may try to modify the content of the OWAMP/TWAMP-Test packets in order to alter the measurement results. However, an implementation can use authenticated mode to detect modification of data. In addition, use encrypted mode to prevent eavesdropping and un-detected modification of the OWAMP/TWAMP-Test packets.

7. IANA Considerations

This memo requests re-allocation of two UDP port numbers from the System Ports range [RFC6335]. Specifically, this memo requests that IANA re-allocate UDP ports 861 and 862 as shown below, leaving the TCP port assignments as-is:

Service Name	Port Number	Transport Protocol	Description	Reference
owamp-control	861	tcp	OWAMP-Control	[RFC4656]
owamp-test	861	udp	OWAMP-Test	[RFCXXXX]
twamp-control	861	tcp	TWAMP-Control	[RFC5357]
twamp-test	862	udp	TWAMP-Test Receiver Port	[RFCXXXX]

Table 1 Re-allocated OWAMP and TWAMP Ports

where RFCXXXX is this memo when published.

8. Contributors

Richard Foote and Luis M. Contreras made notable contributions on this topic.

9. Acknowledgements

The authors thank the IPPM working group for their rapid review; also Muthu Arul Mozhi Perumal and Luay Jalil for their participation and suggestions.

10. References

10.1. Normative References

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

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/info/rfc7594>.

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

10.2. Informative References

[LarsAD] "https://mailarchive.ietf.org/arch/msg/ippm/LzcTPYhPhWhbb5-ncR046XKpnzo", April 2008.

[TimDISCUSS] "https://datatracker.ietf.org/doc/rfc5357/history/", July 2008.

Authors' Addresses

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

Greg Mirsky (editor)
ZTE Corp.

Email: gregimirsky@gmail.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: October 18, 2018

H. Song, Ed.
T. Zhou
Huawei

April 16, 2018

In-situ OAM Data Type Extension
draft-song-ippm-ioam-data-extension-01

Abstract

This document describes a proposal which extends in-situ OAM to support potential future standard tracing data in addition to those currently defined. We provide use cases to motivate our proposal and base the modifications on the latest in-situ OAM header format specification.

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 https://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 October 18, 2018.

Table of Contents

1. Motivation for Data Type Extension

In-situ OAM (iOAM) [I-D.brockners-inband-oam-requirements] records OAM information within user packets while the packets traverse a network. The data types and data formats for in-situ OAM data records have been defined in [I-D.ietf-ippm-ioam-data].

Currently 12 data types and associated formats (including wide format and short format of the same data) are defined in [I-D.ietf-ippm-ioam-data] . The presence of data is indicated by a 16-bit bitmap in the "OAM-Trace-Type" field.

In the current specification only four bits are left to identify new standard data types. Moreover, some data is forced to be bundled together as a single unit to save bitmap space and pack data to the ideal size (e.g., the hop limit and the node id are bundled, and the ingress interface id and the egress interface id are bundled), regardless of the fact that an application may only ask for a part of the data. Last but not the least, each data is forced to be 4-byte aligned for easier access, resulting in waste of header space in many cases.

Since the data plane bandwidth, the data plane packet processing, and the management plane data handling are all precious yet scarce resource, the scheme should strive to be simple and precise. The application should be able to control the exact type and format of data it needs to collect and analyze. It is conceivable that more types of data may be introduced in the future. However, the current scheme cannot support it after all the bits in the bitmap are used up.

For example, when a flow traverses a series of middleboxes (e.g., Firewall, NAT, and load balancer), its identity (e.g., the 5-tuple)

is often altered, which makes the OAM system lose track of the flow trace. In this case, we may want to copy some of the original packet header fields into the iOAM header so the original flow can be identified at any point of the network.

For another example, in wireless, mobile, and optical network environments, some physical data associated with a flow (e.g., power, temperature, signal strength, GPS location) need to be collected to monitor the service performance.

For another example, some data may have different semantics and formats in different networks and application scenarios. An example is the timestamp data type in which NTP, PTP, or any other local defined approaches can be used.

All the above cases require new iOAM data types. More examples are listed in Section 2.2.

There are some other issues about the current specification. For example, bit 7 is used to indicate the presence of variable length opaque state snapshot data; Bit 5 and bit 10 are used to indicate the presence of the application specific data. While these data fields can be used to store arbitrary data, the data is difficult to be standardized and another schema is needed to decode the data, which may lead to low data plane performance as well as interoperability issues. More important, the existence of the variable length data complicates the data processing such as data packing and encapsulation. It is preferred to know the data type and size in advance for efficient hardware implementation.

2. Scalable Data Type Extension

Based on the observation in Section 1, we propose a method for data type encoding which can solve the current limitation and address future data requirements.

2.1. Data Type Bitmap

Bitmap is simple and efficient data structure for high performance data plane implementation. The base bitmap size is kept to be 16 bits. We use one bit to indicate a single type of data in a single format. The last bit in the bitmap (i.e., bit 15), if set, is used to indicate the presence of the next data type bitmap, which is 32 bits long. In the second bitmap, bit 31 is again reserved to indicate a third bitmap, and so on. With each extra bitmap, 31 more data types can be defined.

Figure 1 shows an example of the in-situ OAM header format with two extended OAM trace type fields. Except the OAM Trace Type fields, all other fields remain the same as defined in [I-D.ietf-ippm-ioam-data].

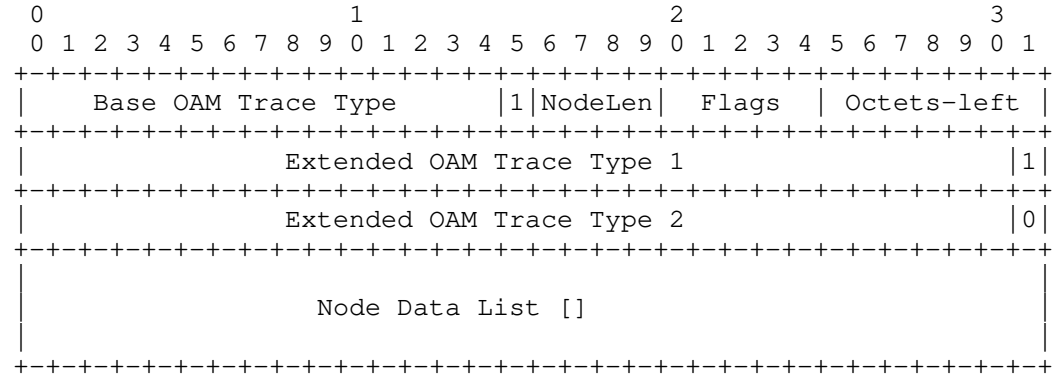


Figure 1: Extended OAM Trace Type Header Format

The specification of the Base OAM Trace Type is the same as the OAM Trace Type in [I-D.ietf-ippm-ioam-data] except the last bit, which is defined as follows:

o Bit 15: When set indicates presence of next bit map.

The OAM trace type fields are labeled as Base OAM Trace Type, Extended OAM Trace Type 1, Extended OAM Trace Type 2, and so on. The Base OAM Trace Type is always present. If no data type is asked by the application in Extended OAM Trace Type n and beyond, then the last bit in the previous bitmap is set to 1 and these extended fields are not included in the header. On the other hand, to eliminate ambiguity, if any data is asked for by the application in Extended OAM Trace Type n, then Extended OAM Trace Type 1 to (n-1) must be included in the header, even though no data type in these bitmaps are needed (i.e., all zero bitmap except the last bit).

The actual data in a node is packed together in the same order as listed in the OAM Trace Type bitmap. Each node is padded to be the multiple of 4 bytes.

2.2. Use Cases

New types of data can be potentially added and standardized, which demand new bits allocated in the OAM Trace Type bitmaps. Some examples are listed here.

- Metered flow bandwidth.
- Time gap between two consecutive flow packets.
- Remaining time budget to the packet delivery deadline.
- Buffer occupancy on the Node.
- Queue depth on each level of hierarchical QoS queues.
- Packet jitter at the Node.
- Current packet IP addresses.
- Current packet port numbers.
- Time using different network timing protocol.
- Other node statistics.

2.3. Consideration for Efficient Data Packing

The length of each data must be the multiple of 2 bytes. However, allowing different data type to have different length, while efficient in storage, makes data alignment and packing difficult.

If we can define the maximum number of data types that can be carried per packet, the offset of each data in the node can be pre-calculated and carried in the iOAM header. The overhead can be justified by the overall space saving of the node data list. Otherwise, each data's offset in the node must be calculated in each device, with the help of a table which stores the size of each data type. We can also arrange the bitmap to reflect the data availability order in the system (e.g., the bit for egress_if_id must be after the bit for ingress_if_id), so in a pipeline-based system, the required data can be packed one after one.

2.4. Alternative Data Extension Possibilities

Bitmap is simple and support parallel processing in hardware. However, it is not the only option to support data type extension. For example, cascaded TLV can be used to support arbitrary number of

new data types. This can be implemented by using a flag bit to indicate the presence of extra data types and packing the number of types and the list of the type IDs after the trace option header. The corresponding data is therefore added in each node data list in the order as its type ID is listed in the extended trace option header.

3. Security Considerations

There is no extra security considerations beyond those have been identified by the original in-situ OAM proposals.

4. IANA Considerations

This memo includes no request to IANA.

5. Acknowledgments

We would like to thank Frank Brockners, Carlos Pignataro, and Shwetha Bhandari for helpful comments and suggestions.

6. Contributors

The document is inspired by numerous discussions with James N. Guichard. He also provided significant comments and suggestions to help improve this document.

7. Informative References

[I-D.brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., <>, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-03 (work in progress), March 2017.

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-00 (work in progress), September 2017.

Authors' Addresses

Haoyu Song (editor)
Huawei
2330 Central Expressway
Santa Clara, 95050
USA

Email: haoyu.song@huawei.com

Tianran Zhou
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China

Email: zhoutianran@huawei.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: October 18, 2018

H. Song, Ed.
T. Zhou
Huawei

April 16, 2018

In-situ OAM Data Validation Option
draft-song-ippm-ioam-data-validation-option-02

Abstract

This document describes several potential performance scalability and capability issues when implementing in-situ OAM on heterogenous target network elements. The document proposes the corresponding solutions and modifications to the current in-situ OAM specification to mitigate the issues. Specifically, in-situ OAM is extended with data validation fields to cope with the node processing capability. We provide use cases to motivate our proposal and base the modifications on the current in-situ OAM header format specification.

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 https://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 October 18, 2018.

Copyright Notice

Table of Contents

1. Introduction

In-situ OAM (iOAM) [I-D.brockners-inband-oam-requirements] records OAM information within user packets while the packets traverse a network. The data types and data formats for in-situ OAM data records have been defined in [I-D.ietf-ippm-ioam-data]. We identify several scalability issues for implementing the current iOAM specification and propose solutions in this draft.

iOAM can designate the flow to add the iOAM header and collect data on the flow forwarding path. The flow can have arbitrary granularity. However, processing the data can be a heavy burden for the network nodes, especially when some data needs to be calculated by the node (e.g., the transit delay). If the flow traffic is heavy, the node may not be able to handle the iOAM processing so many performance issues may occur, such as long latency and packet drop.

Although it is good for the OAM applications to gain the detailed information on every packet at every node, in many cases, such information is often repetitive and redundant. The large quantity of data would also burden the management plane which needs to collect and stream the data for analytics. It is also possible that some nodes cannot provide the requested data at all or are unwilling to provide some data for security or privacy concerns. So a trade-off is needed to balance the performance impact and the data availability and completeness.

We provide several motivating examples. To minimize the network impact, a network operator decides to collect the iOAM data only for initial and last flow packets (e.g., TCP packets with SYN, FIN, and RST flags).

In another example, a head node alternates two iOAM headers with each requesting a subset of iOAM data. Hence, each node on the flow path only needs to handle partial data. The requests can be balanced without exhausting the network nodes.

The above two cases can be realized by manipulating the iOAM header at the domain edge. It is also possible that a node is temporarily under heavy traffic load. It is in danger of dropping packets if it tries to satisfy all the iOAM data requests. It is also possible that, due to the privacy concern or capability issues, a node cannot satisfy the data request indicated in the iOAM header. In these cases, it would rather deny some requests than drop user traffic. This case can be realized by adding some auxiliary fields in the iOAM header.

More examples are listed in Section 2.2.

2. In-situ OAM Sampling and Data Validation

Based on the observation in Section 1, the source edge node should be able to define either the period or the probability to add the iOAM header to the selected flow packet. In this way, only a subset of the flow/sec packets would carry the OAM data, which not only reduces the overall iOAM data quantity but also reduces the processing work load of the network nodes.

Different data type bitmap templates can also be defined and used selectively. For example, template A includes a subset of data and template B includes another subset of data. The two templates can be used in the iOAM header for a flow alternately or in any predefined pattern. This is also an effective way to reduce the node processing load.

2.1. Valid Node Bitmap and Valid Data Bitmap

It is possible that even an iOAM capable node will not add data to the node data list as requested. In some cases, a node can be too busy to handle the data request or some types of the requested data is not available due to privacy and capability reasons. Therefore, we propose to add two bitmaps, a valid node bitmap and a valid data bitmap, to the iOAM specification.

The Node Valid Bitmap (NVB) is inserted before the Node Data List as shown in Figure 1. Each bit in the NVB corresponds to a hop on the packet's forwarding path. The bits are listed in the same order as the hop on the packet's forwarding path. The bitmap is set to all one at first. If a hop cannot add data to the Node Data List, the corresponding bit in the NVB is cleared to 0. The bit location for a

hop can be calculated from the length field (e.g, the bit index is equal to SSize-RHop).The valid node data items in the node data list is equal to the number of 1's in the NVB.

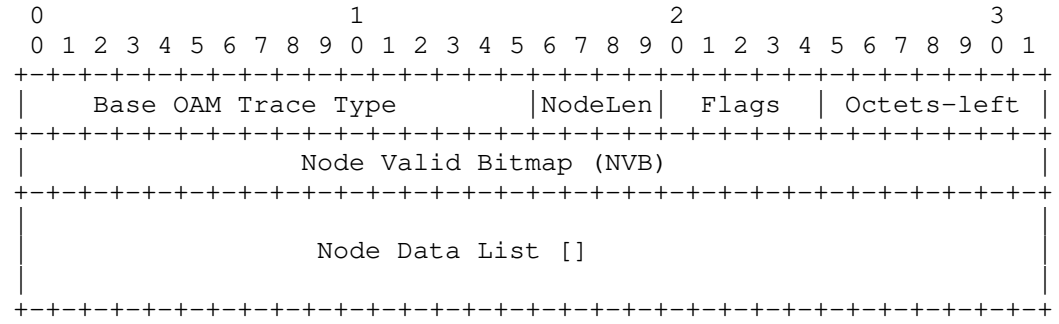


Figure 1: iOAM Header Format with Node Valid Bitmap (NVB)

NVB allows the head node to invalidate some nodes in advance. For example, if the head node wants to exclude the odd-numbered nodes from adding iOAM data, it can set all the corresponding bits to 0. Then at each node, if it finds its corresponding bit in the NVB is 0, it will simply skip the iOAM processing.

In addition to NVB, for each node data in the node data list, a Data Valid Bitmap (DVB) is added before the node data. The number of bits in the DVB is equal to the number of 1's in the OAM Trace Type bitmaps (excluding the next trace type bitmap indicator bits). When the bit is set, the corresponding data is valid in the node; otherwise, the corresponding data is invalid so the management plane should ignore it after the data is collected.

The size of the DVB can be padded to two or four bytes, which allow up to 16 or 32 types of data to be included in a node. The node data list format with the enhanced DVB is shown in Figure 2.

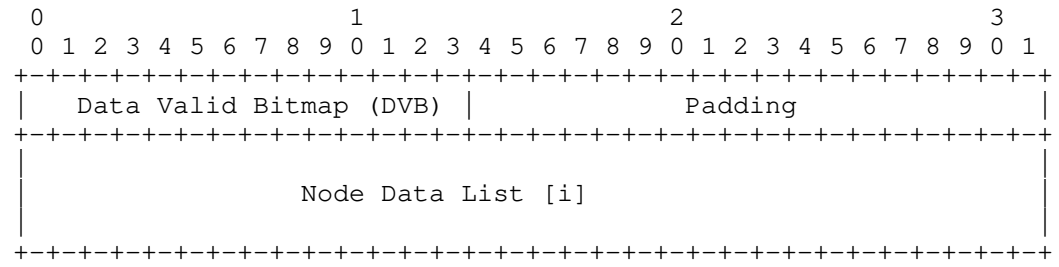


Figure 2: iOAM Node Data with Data Valid Bitmap (DVB)

2.2. Use Cases

We give some examples to show the usefulness of in-situ OAM sampling and data validation features.

- An application needs to track a flow's forwarding path and knows the path will not change frequently, so it sets a low sampling rate to periodically insert the iOAM header to request the node ID.

- In a heterogeneous data plane, some nodes support to provide data x but the other nodes do not support it. However, an application is still interested in collecting data x if available. In this case, iOAM header can still be configured to ask for data x but the nodes that cannot provide the data simply invalidates it by resetting the corresponding bit in the valid data bitmap.

- Multiple sampling rate and multiple data request schema can be defined for a flow based on applications requirements and the data property, so for a flow packet, there can be no iOAM header or different iOAM headers. The node does not need to process all data all the time.

- For security reason, a node decides to not participate in the iOAM data collection. While it processes the other iOAM header fields as usual, it does not set the node valid bit in the Node Valid Bitmap and add node data to the Node Data List.

- To reduce the node processing load, the head node alternately uses two NVBs with one of them invalidating all the even-numbered nodes and the other invalidating all the odd-numbered nodes. Therefore, a node only needs to process the iOAM for every two packets of the flow.

3. Security Considerations

There is no extra security considerations beyond those have been identified by in-situ OAM protocol.

4. IANA Considerations

This memo includes no request to IANA.

5. Acknowledgments

We would like to thank Frank Brockners, Carlos Pignataro, Shwetha Bhandari, and Tal Mizrahi for helpful comments and suggestions.

6. Contributors

The document is inspired by numerous discussions with James N. Guichard. He also provided significant comments and suggestions to help improve this document.

7. Informative References

[I-D.brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., <>, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-02 (work in progress), October 2016.

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-00 (work in progress), September 2017.

Authors' Addresses

Haoyu Song (editor)
Huawei
2330 Central Expressway
Santa Clara, 95050
USA

Email: haoyu.song@huawei.com

Tianran Zhou
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China

Email: zhoutianran@huawei.com

ippm
Internet-Draft
Intended status: Standards Track
Expires: October 19, 2018

H. Song, Ed.
T. Zhou
Huawei

April 17, 2018

Control In-situ OAM Overhead with Segment IOAM
draft-song-ippm-segment-ioam-01

Abstract

This document describes a proposal which partitions an in-situ OAM (iOAM) domain into multiple segments in order to control the iOAM data overhead, adapt to the path MTU limitations, and enable new applications. We discuss several use cases to motivate our proposal and base the necessary modifications on the current in-situ OAM header format specification.

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 https://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 October 19, 2018.

Table of Contents

1. Introduction

In-situ OAM (iOAM) [I-D.brockners-inband-oam-requirements] records OAM information within user packets while the packets traverse a network. The data types and data formats for in-situ OAM data records have been defined in [I-D.ietf-ippm-ioam-data].

iOAM may incur significant overhead on user packets. The overhead includes the iOAM header and the node data list for each network element.

The total size of data is limited by the MTU. When the number of required data types is large and the forwarding path length is long, it is possible that there is not enough space in the user packets to hold the iOAM header and data. The current proposal is to label the overflow status and stop adding new node data to the packet, leading to the loss of information.

Even if the header has enough space to hold the iOAM data, the overhead may be too large and consumes too much bandwidth. For example, if we assume moderate 20 bytes of data per node, a path with length of 10 will need 200 bytes to hold the data. This will inflate small 64-byte packets by more than four times. Even for the largest packet size (e.g., 1500 bytes), the overhead (>10%) is not negligible. Therefore, we need to limit the iOAM data overhead without sacrificing the data collection capability.

Here we have another interesting related issue. Packets can be dropped anywhere in a network for various reasons. If we can only collect iOAM data at the path end, we lose all data from the dropped

packets and have no idea where the packets are dropped. This defies the purpose of iOAM and makes those iOAM-enabled nodes work in vain.

2. Segment In-situ OAM

Based on the observation in Section 1, we propose a method to limit the size of the node data list.

2.1. Segment and Hops

A hop is a node on a flow's forwarding path which is capable of processing iOAM data. A segment is a fixed number hops on a flow's forwarding path. While working in the "per hop" trace mode, the segment size (SSize) and the remaining hops (RHop), is added to the iOAM header at the edge. Initially, RHop is equal to SSize. At each hop, if RH is not zero, the node data is added to the node data list at the corresponding location and then RH is decremented by 1. If RH is equal to 0 when receiving the packet, the node needs to remove (in incremental trace option) or clear (in pre-allocated trace option) the iOAM node data list and reset RHop to SSize. Then the node will add its data to the node data list as if it is the edge node.

The stripped iOAM data at the segement edge can be immediately exported to a collector.

Figure 1 shows the proposed in-situ OAM header format. The bit 23 in the Flags field is used to indicate the current header is a segment iOAM header. In this context, the last octet in the iOAM header is partitioned into two 4-bit nibbles. The first nibble (SSize) is used to save the segment size and the second nibble (RHop) is used to save the remaining hops. This limits the maximum segment size to 15.

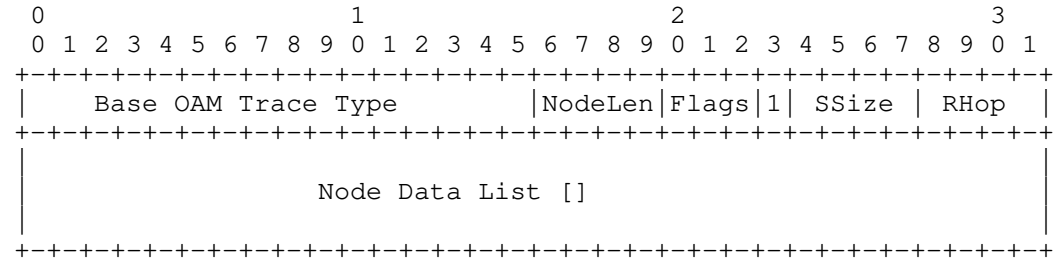


Figure 1: Segment iOAM Header Format

In the special case when SSize is set to 0, no data will be recorded in the node data list. The requested data listed in the OAM Trace

Type will be immediately exported to the collector. This way the iOAM overhead is minimized.

2.2. Considerations for Data Handling

At any hop when RHop is equal to 0, the node data list is copied from the iOAM header. The data can be encapsulated and reported to the controller or the edge node as configured. The encapsulation and report method is beyond the scope of this draft but should be comply with the method used by the iOAM edge node.

The actual size of the last segment may not be equal to SSize but this is not a problem.

2.3. Use Cases

Segment iOAM is necessary in the following example scenarios:

- Segment iOAM can be used to detect at which segment the flow packet is dropped. If the SSize is set to 1, then the exact drop node can be identified. The iOAM data before the dropping point is also retained.

- The path MTU allows to add at most k node data in the list to avoid fragmentation. Therefore SSize is set to k and at each hop where RHop is 0, the node data list is retrieved and sent in a standalone packet.

- A flow contains mainly short packets and travels a long path. It would be inefficient to keep a large node data list in the packet so the network bandwidth utilization rate is low. In this case, segment iOAM can be used to limit the ratio of the iOAM data to the flow packet payload.

- The network allows at most n bytes budget for the iOAM data. There is a tradeoff between the number of data types that can be collected and the number of hops for data collecting. The segment size is therefore necessary to meet the application's data requirement (i.e., SSize * Node Data Size < n).

3. Security Considerations

There is no extra security considerations beyond those have been identified by in-situ OAM protocol.

4. IANA Considerations

This memo includes no request to IANA.

5. Acknowledgments

We would like to thank Frank Brockners, Carlos Pignataro, and Shwetha Bhandari for helpful comments and suggestions.

6. Contributors

The document is inspired by numerous discussions with James N. Guichard. He also provided significant comments and suggestions to help improve this document.

7. Informative References

[I-D.brockners-inband-oam-requirements]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mozes, D., Mizrahi, T., <>, P., and r. remy@barefootnetworks.com, "Requirements for In-situ OAM", draft-brockners-inband-oam-requirements-02 (work in progress), October 2016.

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and d. daniel.bernier@bell.ca, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-00 (work in progress), September 2017.

Authors' Addresses

Haoyu Song (editor)
Huawei
2330 Central Expressway
Santa Clara, 95050
USA

Email: haoyu.song@huawei.com

Tianran Zhou
Huawei
156 Beiqing Road
Beijing, 100095
P.R. China

Email: zhoutianran@huawei.com

IPPM Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 21, 2019

X. Min
G. Mirsky
ZTE
L. Bo
China Telecom

December 18, 2018

Extended OAM to Carry In-situ OAM Capabilities
draft-xiao-ippm-ioam-conf-state-02

Abstract

This document describes an extension for OAM packets including MPLS LSP Ping/Traceroute [RFC8029], ICMP Ping/Traceroute for SRv6 [I-D.ali-spring-srv6-oam] and SFC Ping/Traceroute [I-D.ietf-sfc-multi-layer-oam], which can be used within an IOAM domain, allowing the IOAM encapsulating node to acquire IOAM capabilities of each IOAM transit node and/or IOAM decapsulating node easily and dynamically.

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 https://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 June 21, 2019.

Table of Contents

1. Introduction

The Data Fields for In-situ OAM (IOAM) [I-D.ietf-ippm-ioam-data] defines data fields for IOAM which records OAM information within the packet while the packet traverses a particular network domain, which is called an IOAM domain. IOAM can be used to complement OAM mechanisms based on, e.g., ICMP or other types of probe packets, and IOAM mechanisms can be leveraged where mechanisms using, e.g., ICMP do not apply or do not offer the desired results.

As specified in [I-D.ietf-ippm-ioam-data], within the IOAM-domain, the IOAM data may be updated by network nodes that the packet traverses. The device which adds an IOAM data container to the packet to capture IOAM data is called the "IOAM encapsulating node", whereas the device which removes the IOAM data container is referred to as the "IOAM decapsulating node". Nodes within the domain which are aware of IOAM data and read and/or write or process the IOAM data are called "IOAM transit nodes". Both the IOAM encapsulating node and the decapsulating node are referred to as domain edge devices, which can be hosts or network devices.

In order to add accurate IOAM data container to the packet, the IOAM encapsulating node needs to know IOAM capabilities at the IOAM transit nodes and/or the IOAM decapsulating node in a whole, e.g.,

how many IOAM transit nodes will add tracing data and what kinds of data fields will be added. This document describes an extension for OAM packets including MPLS LSP Ping/Traceroute [RFC8029], ICMP Ping/Traceroute for SRv6 [I-D.ali-spring-srv6-oam] and SFC Ping/Traceroute [I-D.ietf-sfc-multi-layer-oam], which can be used within an IOAM domain, allowing the IOAM encapsulating node to acquire IOAM capabilities of each IOAM transit node and/or IOAM decapsulating node easily and dynamically.

1.1. Conventions Used in This Document

1.1.1. Terminology

E2E: Edge to Edge

ICMP: Internet Control Message Protocol

IOAM: In-situ Operations, Administration, and Maintenance

LSP: Label Switched Path

MPLS: Multi-Protocol Label Switching

MTU: Maximum Transmission Unit

NTP: Network Time Protocol

OAM: Operations, Administration, and Maintenance

POSIX: Portable Operating System Interface

POT: Proof of Transit

PTP: Precision Time Protocol

SFC: Service Function Chain

SRv6: Segment Routing with IPv6 Data plane

TTL: Time to Live

1.1.2. Requirements Language

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

2. IOAM Capabilities Formats

2.1. IOAM Capabilities TLV

IOAM Capabilities uses TLV (Type-Length-Value tuple) which have the following format:

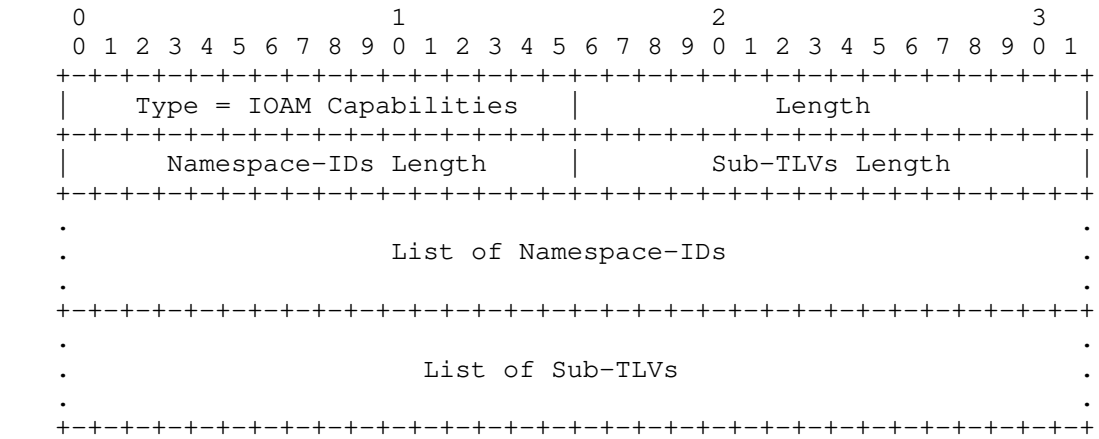


Figure 1: IOAM Capabilities TLV

When this TLV is present in the echo request sent by an IOAM encapsulating node, it means that the IOAM encapsulating node requests the receiving node to reply with its IOAM capabilities. If there is no IOAM capabilities to be reported by the receiving node, then this TLV SHOULD be ignored by the receiving node. List of Namespace-IDs MAY be included in this TLV of echo request, it means that the IOAM encapsulating node requests only the IOAM capabilities which matchs one of the Namespace-IDs. The Namespace-ID has the same definition as what's specified in [I-D.ietf-ippm-ioam-data].

When this TLV is present in the echo reply sent by an IOAM transit node and/or an IOAM decapsulating node, it means that IOAM function is enabled at this node and this TLV contains IOAM capabilities of the sender. List of Namespace-IDs MAY be included in this TLV of echo reply. It means that the IOAM capabilities included in this TLV match one of the Namespace-IDs. If a List of Namespace-IDs is present in the TLV of echo request, then the List of Namespace-IDs in the TLV of echo reply MUST be a subset of that one. List of Sub-TLVs which contain the IOAM capabilities SHOULD be included in this TLV of the echo reply. Note that the IOAM encapsulating node or the IOAM decapsulating node can also be an IOAM transit node.

Type is set to the value which indicates that it's an IOAM Capabilities TLV.

Length is the length of the TLV's Value field in octets, Namespace-IDs Length is the Length of the List of Namespace-IDs field in octets, Sub-TLVs Length is the length of the List of Sub-TLVs field in octets.

Value field of this TLV or any Sub-TLV is zero padded to align to a 4-octet boundary. Based on the data fields for IOAM specified in [I-D.ietf-ippm-ioam-data], four kinds of Sub-TLVs are defined in this document, and in an IOAM Capabilities TLV the same kind of Sub-TLV can appear more times than one with different Namespace-ID.

2.1.1. IOAM Tracing Capabilities sub-TLV

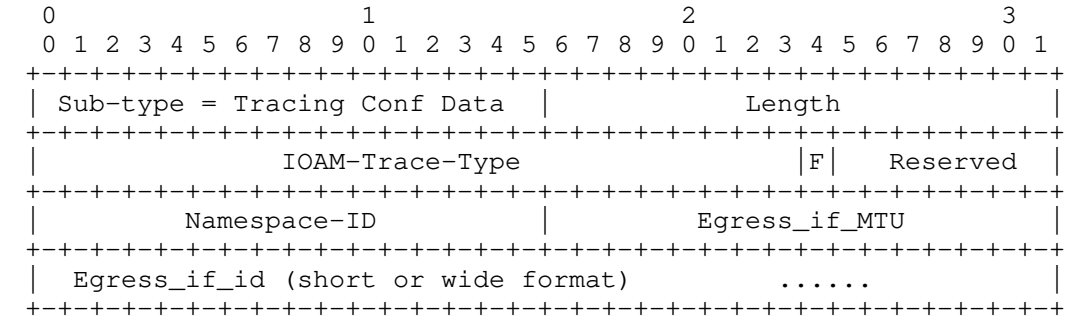


Figure 2: IOAM Tracing Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM transit node and IOAM tracing function is enabled at this IOAM transit node.

Sub-type is set to the value which indicates that it's an IOAM Tracing Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, if Egress_if_id is in the short format which is 16 bits long, it MUST be set to 10, and if Egress_if_id is in the wide format which is 32 bits long, it MUST be set to 12.

IOAM-Trace-Type field has the same definition as what's specified in section 4.2 of [I-D.ietf-ippm-ioam-data].

F bit is specified to indicate whether the pre-allocated trace or incremental trace is enabled. F bit is set to 1 when pre-allocated trace is enabled and set to 0 when the incremental trace is enabled. The meaning and difference of pre-allocated trace and incremental trace are described in section 4.1 of [I-D.ietf-ippm-ioam-data]. If the IOAM encapsulating node receives different F bit value from different IOAM transit node, then the IOAM encapsulating node will reserve data space in the IOAM header for the IOAM transit node that set F bit to 1, and the IOAM encapsulating node won't reserve data space in the IOAM header for the IOAM transit node that set F bit to 0.

Reserved field is reserved for future use and MUST be set to zero.

Namespace-ID field has the same definition as what's specified in section 4.2 of [I-D.ietf-ippm-ioam-data].

Egress_if_MTU field has 16 bits and specifies the MTU of the egress interface out of which the sending node would forward the received echo request.

Egress_if_id field has 16 bits (in short format) or 32 bits (in wide format) and specifies the identifier of the egress interface out of which the sending node would forward the received echo request.

2.1.2. IOAM Proof of Transit Capabilities sub-TLV

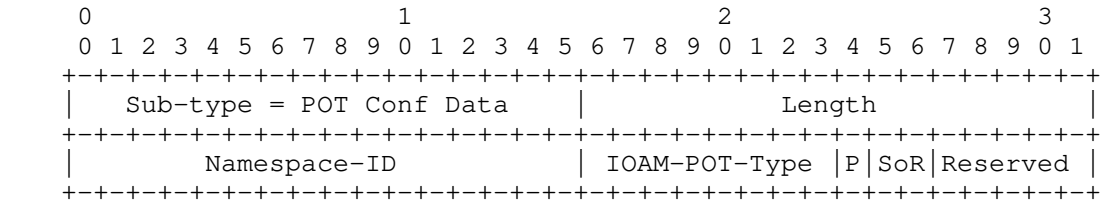


Figure 3: IOAM Proof of Transit Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM transit node and IOAM proof of transit function is enabled at this IOAM transit node.

Sub-type is set to the value which indicates that it's an IOAM Proof of Transit Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 4.

Namespace-ID field has the same definition as what's specified in section 4.3 of [I-D.ietf-ippm-ioam-data].

IOAM-POT-Type field and P bit have the same definition as what's specified in section 4.3 of [I-D.ietf-ippm-ioam-data]. If the IOAM encapsulating node receives IOAM-POT-Type and/or P bit values from an IOAM transit node that are different from its own, then the IOAM encapsulating node MAY choose to abandon the proof of transit function or to select one kind of IOAM-POT-Type and P bit, it's based on the policy applied to the IOAM encapsulating node.

SoR field has two bits which means the size of "Random" and "Cumulative" data, which are specified in section 4.3 of [I-D.ietf-ippm-ioam-data]. This document defines SoR as follow:

0b00 means 64-bit "Random" and 64-bit "Cumulative" data.

0b01~0b11: Reserved for future standardization

Reserved field is reserved for future use and MUST be set to zero.

2.1.3. IOAM Edge-to-Edge Capabilities sub-TLV

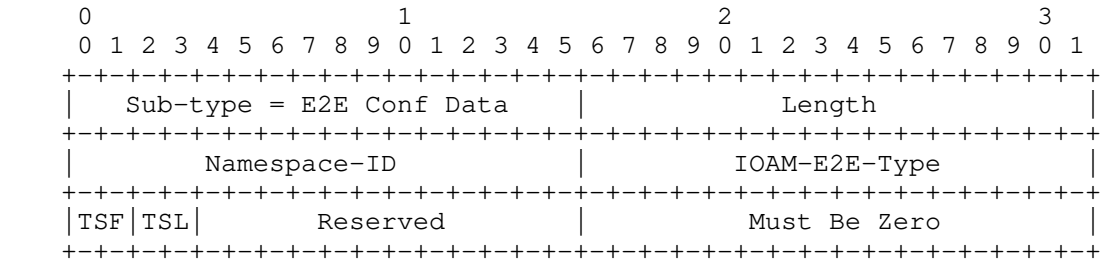


Figure 4: IOAM Edge-to-Edge Capabilities Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM decapsulating node and IOAM edge-to-edge function is enabled at this IOAM decapsulating node. That is to say, if the IOAM encapsulating node receives this sub-TLV, the IOAM encapsulating node can determine that the node which sends this sub-TLV is an IOAM decapsulating node.

Sub-type is set to the value which indicates that it's an IOAM Edge-to-Edge Capabilities sub-TLV.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 8.

Namespace-ID field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data].

IOAM-E2E-Type field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data].

TSF field specifies the timestamp format used by the sending node. This document defines TSF as follow:

0b00: PTP timestamp format

0b01: NTP timestamp format

0b10: POSIX timestamp format

0b11: Reserved for future standardization

TSL field specifies the timestamp length used by the sending node. This document defines TSL as follow:

When TSF field is set to 0b00 which indicates PTP timestamp format:

0b00: 64-bit PTPv1 timestamp as defined in IEEE1588-2008 [IEEE1588v2]

0b01: 80-bit PTPv2 timestamp as defined in IEEE1588-2008 [IEEE1588v2]

0b10~0b11: Reserved for future standardization

When TSF field is set to 0b01 which indicates NTP timestamp format:

0b00: 32-bit NTP timestamp as defined in NTPv4 [RFC5905]

0b01: 64-bit NTP timestamp as defined in NTPv4 [RFC5905]

0b10: 128-bit NTP timestamp as defined in NTPv4 [RFC5905]

0b11: Reserved for future standardization

When TSF field is set to 0b10 or 0b11, the TSL field would be ignored.

Reserved field is reserved for future use and MUST be set to zero.

2.1.4. IOAM End-of-Domain sub-TLV

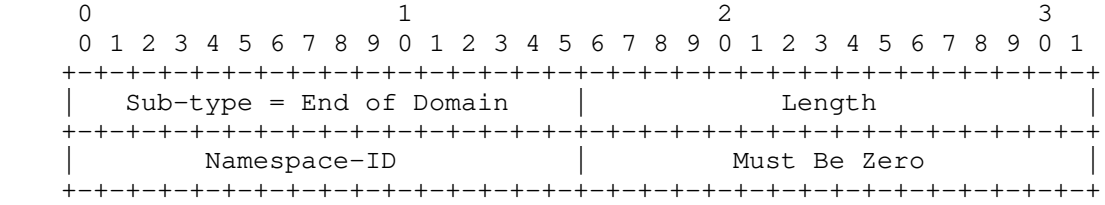


Figure 5: IOAM End of Domain Sub-TLV

When this sub-TLV is present in the IOAM Capabilities TLV, it means that the sending node is an IOAM decapsulating node. That is to say, if the IOAM encapsulating node receives this sub-TLV, the IOAM encapsulating node can determine that the node which sends this sub-TLV is an IOAM decapsulating node. When the IOAM Edge-to-Edge Capabilities sub-TLV is present in the IOAM Capabilities TLV sent by the IOAM decapsulating node, the IOAM End-of-Domain sub-TLV doesn't need to be present in the same IOAM Capabilities TLV, otherwise the End-of-Domain sub-TLV MUST be present in the IOAM Capabilities TLV sent by the IOAM decapsulating node. Since both the IOAM Edge-to-Edge Capabilities sub-TLV and the IOAM End-of-Domain sub-TLV can be used to indicate that the sending node is an IOAM decapsulating node, it's recommended to include only the IOAM Edge-to-Edge Capabilities sub-TLV if IOAM edge-to-edge function is enabled at this IOAM decapsulating node.

Length is the length of the sub-TLV's Value field in octets, and MUST be set to 4.

Namespace-ID field has the same definition as what's specified in section 4.4 of [I-D.ietf-ippm-ioam-data].

3. Operational Guide

Once the IOAM encapsulating node is triggered to acquire IOAM capabilities of each IOAM transit node and/or IOAM decapsulating node, the IOAM encapsulating node will send a batch of echo requests that include the IOAM Capabilities TLV, first with TTL equal to 1 to reach the nearest node which may be an IOAM transit node or not, then with TTL equal to 2 to reach the second nearest node which also may be an IOAM transit node or not, on the analogy of this to increase 1 to TTL every time the IOAM encapsulating node sends a new echo

request, until the IOAM encapsulating node receives echo reply sent by the IOAM decapsulating node, which must contain the IOAM Capabilities TLV including the IOAM Edge-to-Edge Capabilities sub-TLV or the IOAM End-of-Domain sub-TLV.

The IOAM encapsulating node may be triggered by the device administrator, the network management, the network controller, or even the live user traffic, and the specific triggering mechanisms are outside the scope of this document.

Each IOAM transit node and/or IOAM decapsulating node that receives an echo request containing the IOAM Capabilities TLV will send an echo reply to the IOAM encapsulating node, and within the echo reply, there must be an IOAM Capabilities TLV containing one or more sub-TLVs. The IOAM Capabilities TLV contained in the echo request would be ignored by the receiving node that is unaware of IOAM.

4. Security Considerations

Knowledge of the state of the IOAM domain may be considered confidential. Implementations SHOULD provide a means of filtering the addresses to which echo reply messages, MPLS LSP Ping/Traceroute, ICMP Ping/Traceroute for SRv6 or SFC Ping/Traceroute, may be sent.

5. IANA Considerations

This document has no IANA actions.

6. Acknowledgements

The authors appreciate the f2f discussion with Frank Brockners on this document.

7. Normative References

[I-D.ali-spring-srv6-oam]
Ali, Z., Filsfils, C., Kumar, N., Pignataro, C., faiqbal@cisco.com, f., Gandhi, R., Leddy, J., Matsushima, S., Raszuk, R., daniel.voyer@bell.ca, d., Dawra, G., Peirens, B., Chen, M., and G. Naik, "Operations, Administration, and Maintenance (OAM) in Segment Routing Networks with IPv6 Data plane (SRv6)", draft-ali-spring-srv6-oam-02 (work in progress), October 2018.

[I-D.ietf-ippm-ioam-data]
Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., daniel.bernier@bell.ca, d., and J. Lemon, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data-04 (work in progress), October 2018.

[I-D.ietf-sfc-multi-layer-oam]
Mirsky, G., Meng, W., Khasnabish, B., and C. Wang, "Active OAM for Service Function Chains in Networks", draft-ietf-sfc-multi-layer-oam-00 (work in progress), November 2018.

[IEEE1588v2]
Institute of Electrical and Electronics Engineers, "IEEE Std 1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", IEEE Std 1588-2008, 2008, <http://standards.ieee.org/findstds/standard/1588-2008.html>.

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

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC8029] Kompella, K., Swallow, G., Pignataro, C., Ed., Kumar, N., Aldrin, S., and M. Chen, "Detecting Multiprotocol Label Switched (MPLS) Data-Plane Failures", RFC 8029, DOI 10.17487/RFC8029, March 2017, <https://www.rfc-editor.org/info/rfc8029>.

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

Authors' Addresses

Xiao Min
ZTE
Nanjing
China

Phone: +86 25 88016574
Email: xiao.min2@zte.com.cn

Greg Mirsky
ZTE
USA

Email: gregimirsky@gmail.com

Lei Bo
China Telecom
Beijing
China

Phone: +86 10 50902903
Email: leibo.bri@chinatelecom.cn

IPPM Working Group X. Min
Internet-Draft D. Zhanwei
Intended status: Standards Track ZTE
Expires: April 15, 2018 October 12, 2017

TWAMP Extensions for Direct Loss Measurement
draft-xiao-ippm-twamp-ext-direct-loss-01

Abstract

This document describes an optional extension for Two-Way Active Measurement Protocol (TWAMP) allowing direct loss measurement of IP traffic with the TWAMP-Test protocol.

Table of Contents

1. Introduction

The Two-Way Active Measurement Protocol (TWAMP) [RFC5357] is an extension of the One-Way Active Measurement Protocol (OWAMP) [RFC4656]. The TWAMP is a well-defined protocol which is widely used for measurement of two-way or round-trip metrics, in addition to the one-way metrics of OWAMP.

When TWAMP or OWAMP is used for measurement of metric loss, it actually measures the loss of test packets, so it's a kind of "synthetic" loss measurement. In some cases, considering the IP traffic loss characteristics of short-time burst loss, it's expected to get more accurate loss measurement results when measuring the direct loss of IP traffic instead of test packets.

To address this, this document describes an optional and simple feature for TWAMP, which allows TWAMP-Test protocol to be used for direct loss measurement of IP traffic.

1.1. Conventions Used in This Document

1.1.1. Terminology

DSCP: Differentiated Services Code Point

IPPM: IP Performance Metrics

TWAMP: Two-Way Active Measurement Protocol

OWAMP: One-Way Active Measurement Protocol

UDP: User Datagram Protocol

1.1.2. Requirements Language

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

2. TWAMP-Control Extension

TWAMP connection establishment follows the procedure defined in Section 3.1 of [RFC4656] and Section 3.1 of [RFC5357] where the Modes field is used to identify and select specific communication capabilities. At the same time, the Modes field is recognized and used as an extension mechanism [RFC6038]. The new feature requires a new flag, Direct Loss Measurement flag, to identify the ability of both Session-Sender and Session-Reflector to perform direct loss measurement, and to support the new Session-Sender packet format and the new Session-Reflector packet format in the TWAMP-Test protocol. See Section 6 for details on the assigned bit position.

2.1. Connection Setup with Direct Loss Measurement Mode

The Server sets the Direct Loss Measurement flag in the Modes field of the Server Greeting message to indicate its capability and willingness to perform it. If the Control-Client agrees to perform direct loss measurement on some or all test sessions invoked with this control connection, it MUST set the Direct Loss Measurement flag in the Modes field in the Setup Response message.

3. TWAMP-Test Extensions

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. TWAMP, see Section 4 of [RFC5357], defines two additional test packet formats for packets transmitted by the Session-Reflector. The appropriate format depends on the security mode chosen. The new mode specified in this document adds counter(s) of IP traffic packets into each test packet format.

When the Server and Control-Client have agreed to use the direct loss measurement mode during control connection setup, then the Session-

Sender and the Session-Reflector SHOULD all conform to the requirements of that mode, as identified below.

3.1. Sender Test Packet Format and Content

Formats of the test packet transmitted by the Session-Sender in unauthenticated, authenticated, and encrypted modes have been defined in Section 4.1.2 of [RFC4656] (as indicated in Section 4.1.2 of [RFC5357]). For the Session-Sender that supports direct loss measurement, these formats are displayed in Figures 1 and 2.

For unauthenticated mode:

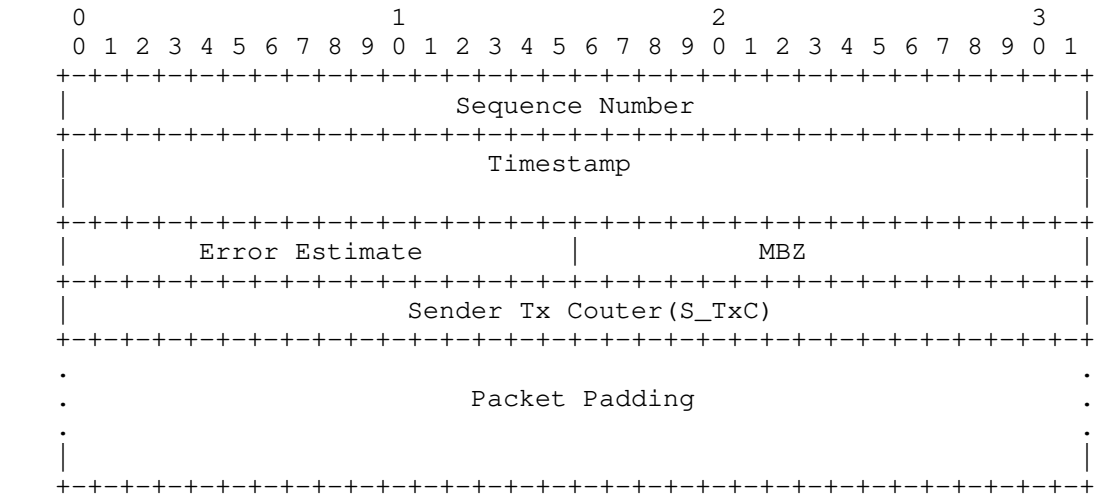


Figure 1: Session-Sender Test Packet Format with direct loss measurement in Unauthenticated Mode

For authenticated and encrypted modes:

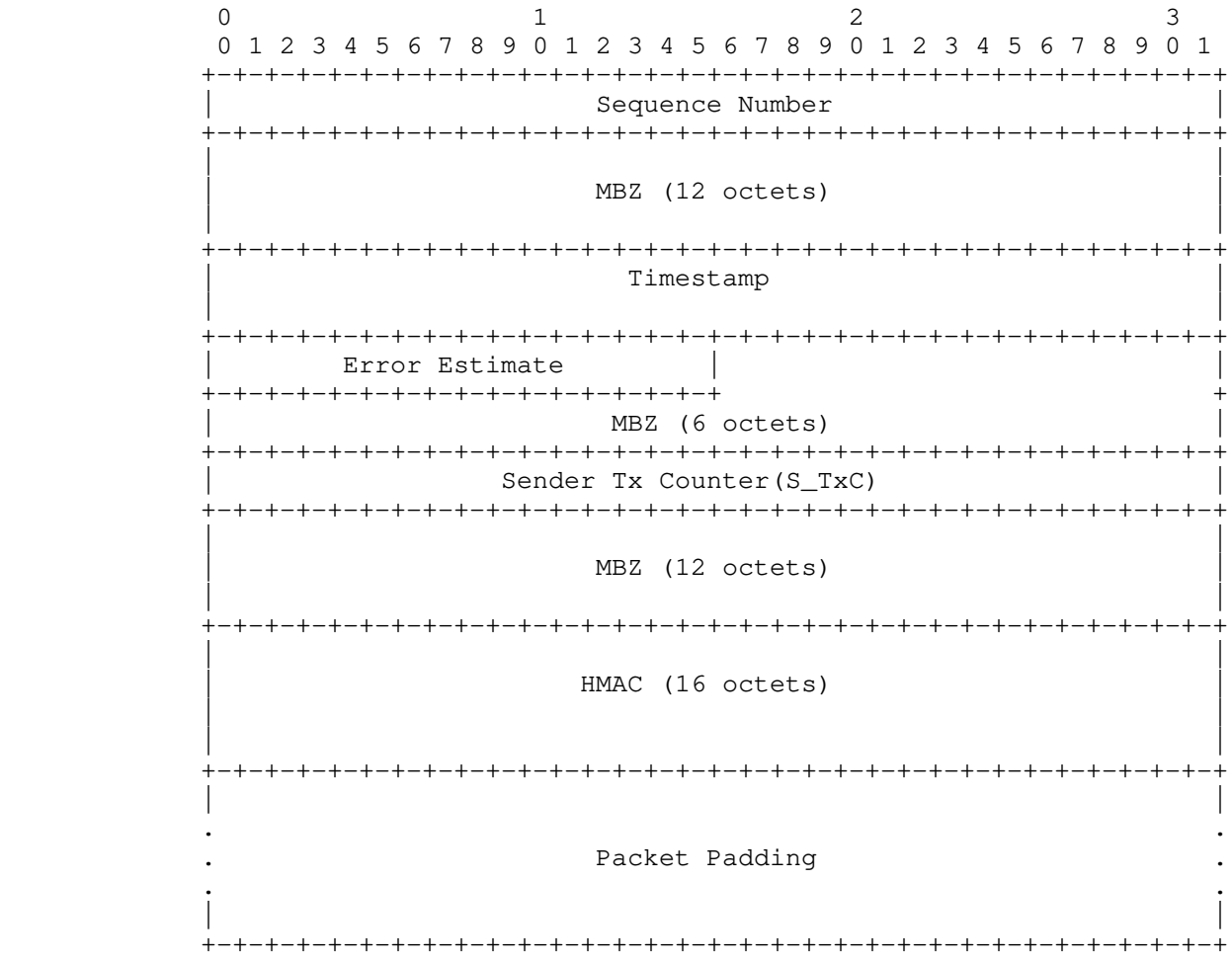


Figure 2: Session-Sender Test Packet Format with Direct Loss Measurement in Authenticated and Encrypted Modes

The Sender Tx Counter (S_TxC) is set to the number of IP packets of the particular monitored flow transmitted towards the Reflector. Section 4 provides operational guide on how to determine the scope of IP traffic packets that need to be counted. Note that the Sender test packets are not counted.

In authenticated and encrypted modes, the S_TxC is followed by a new 12 octets MBZ (MUST be zero) field to make it 16-octet aligned, which is required for authentication and encryption.

The intention of embedding S_TxC in the Session-Sender test packets is for the Session-Sender to calculate direct loss of IP traffic, and the loss calculation algorithm is described in Section 3.3.

The new direct loss measurement mode defined in this document and the two extended TWAMP modes defined in [RFC6038] can be selected simultaneously.

When the Symmetrical Size mode defined in [RFC6038] is also selected, S_TxC SHOULD be embedded in the Session-Sender Packet formatted in Section 5.1.4 of [RFC6038], with the same position as depicted in Figure 1.

When the Reflect Octets mode defined in [RFC6038] is also selected, S_TxC SHOULD be embedded in the Session-Sender Packet formatted in Section 5.1.2 of [RFC6038], with the same position as depicted in Figure 1.

When both the Symmetrical Size mode and the Reflect Octets mode are also selected, S_TxC SHOULD be embedded in the Session-Sender Packet formatted in Section 5.1.5 of [RFC6038], with the same position as depicted in Figure 1.

3.2. Reflector Test Packet Format and Content

Formats of the test packet transmitted by the Session-Reflector in unauthenticated, authenticated, and encrypted modes have been defined in Section 4.2.1 of [RFC5357]. For the Session-Reflector that supports direct loss measurement, these formats are displayed in Figures 3 and 4.

For unauthenticated mode:

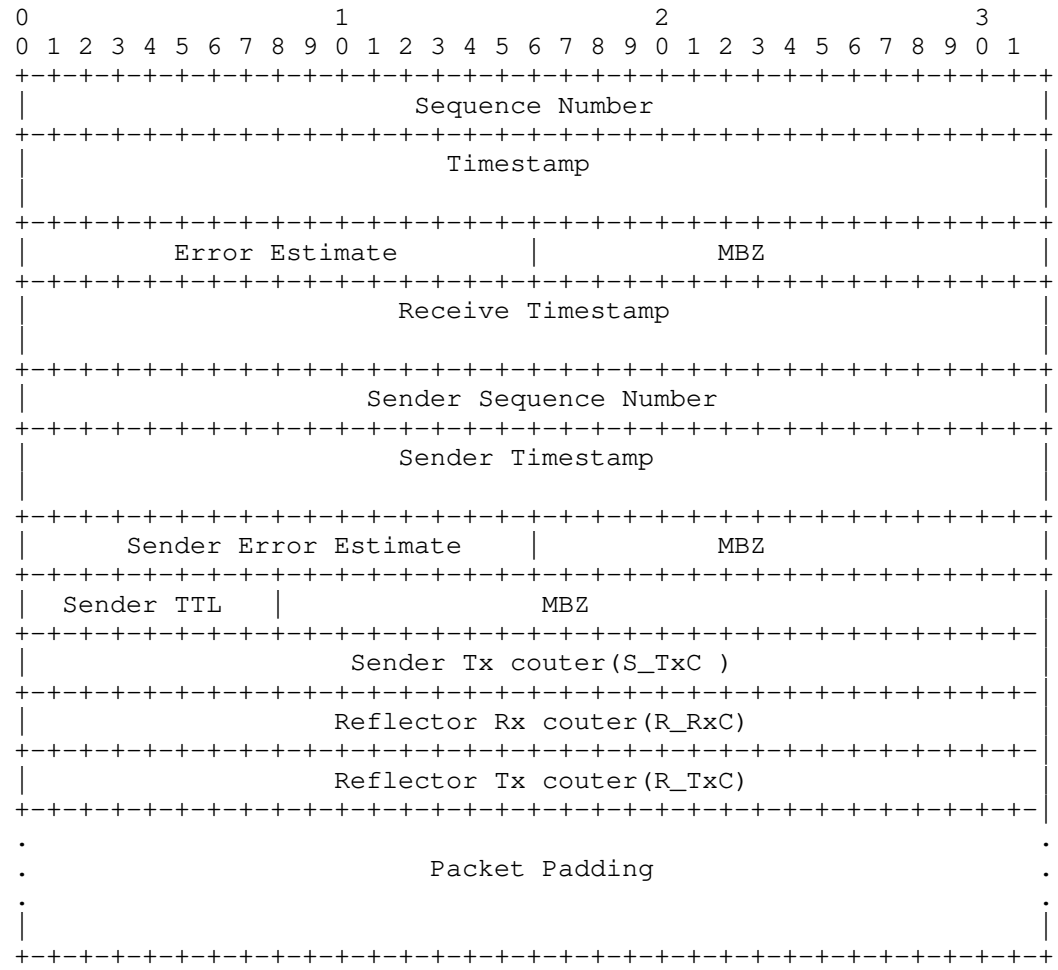
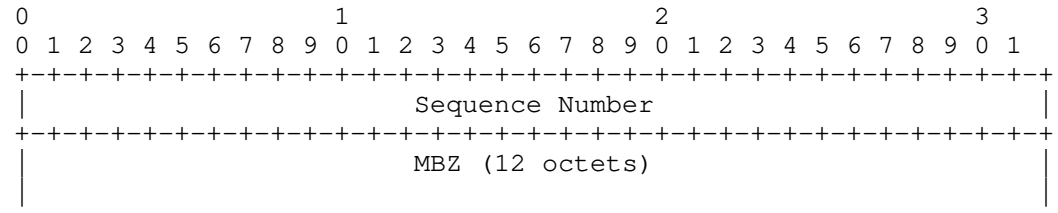
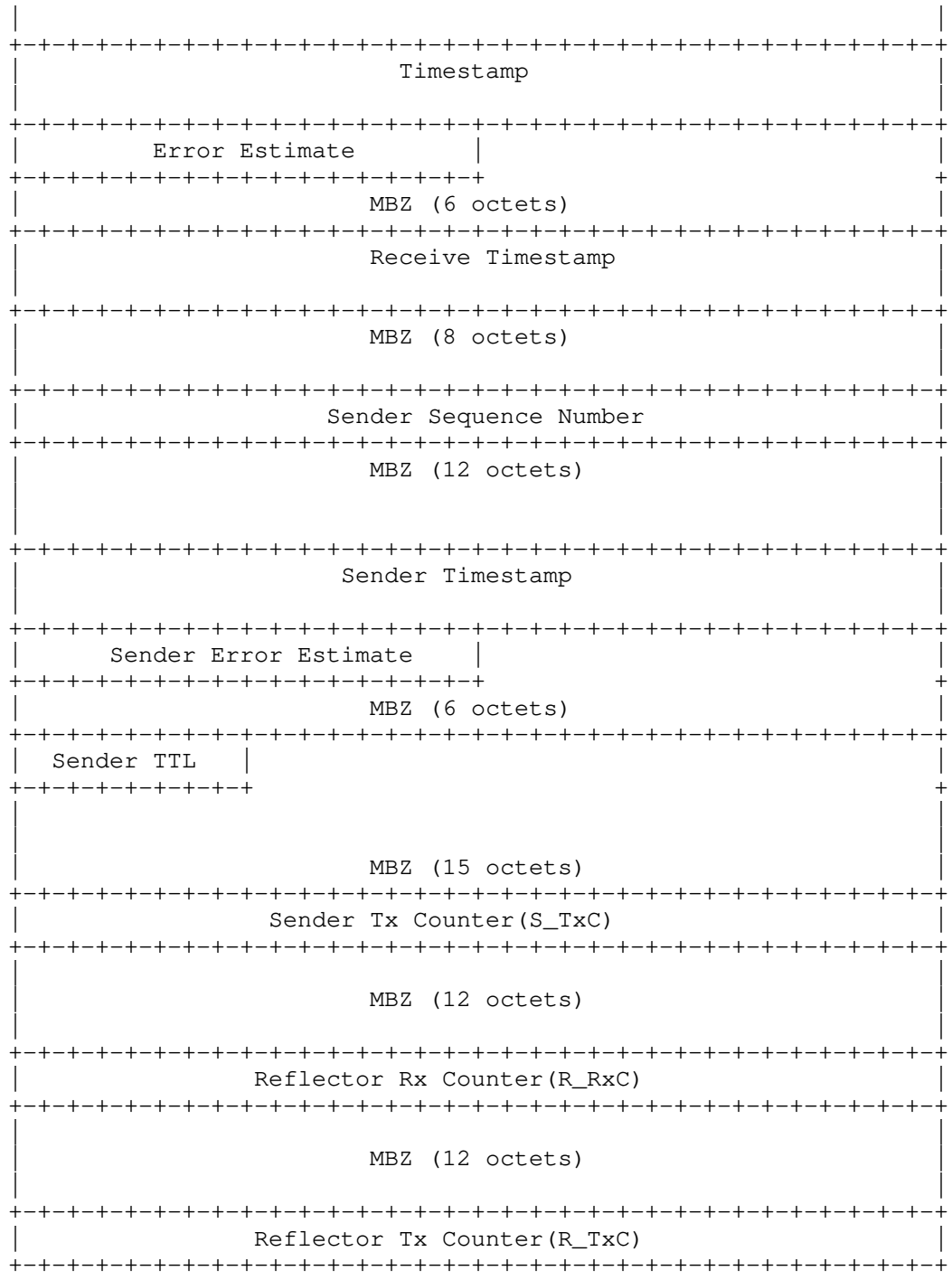


Figure 3: Session-Reflector Test Packet Format with direct loss measurement in Unauthenticated Mode

For authenticated and encrypted modes:




Timestamp
Error Estimate
MBZ (6 octets)
Receive Timestamp
MBZ (8 octets)
Sender Sequence Number
MBZ (12 octets)
Sender Timestamp
Sender Error Estimate
MBZ (6 octets)
Sender TTL
MBZ (15 octets)
Sender Tx Counter(S_TxC)
MBZ (12 octets)
Reflector Rx Counter(R_RxC)
MBZ (12 octets)
Reflector Tx Counter(R_TxC)

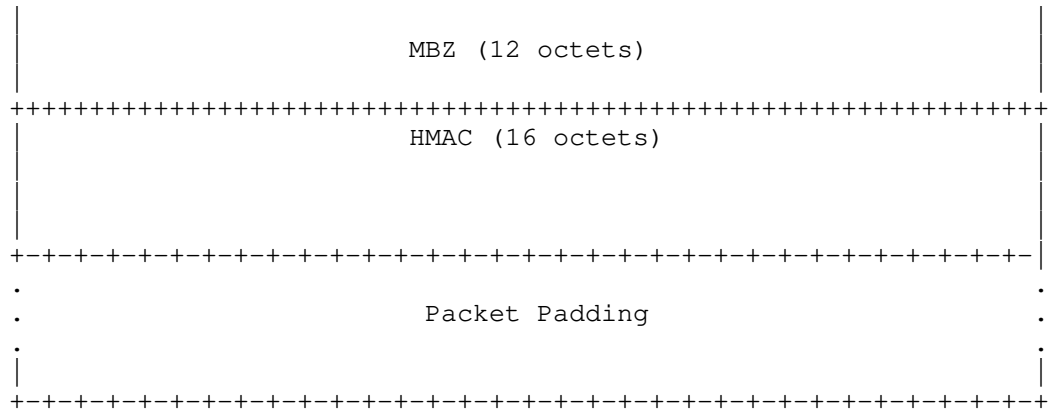


Figure 4: Session-Reflector Test Packet Format with Direct Loss Measurement in Authenticated and Encrypted Modes

The Sender Tx Counter (S_TxC) is copied from the received Sender Test Packet.

The Reflector Rx Counter (R_RxC) is set to the number of IP traffic packets received by the Reflector. Section 4 provides operational guide on how to determine the scope of IP traffic packets that need to be counted. Note that the Sender test packets are not counted.

The Reflector Tx Counter (R_TxC) is set to the number of IP traffic packets transmitted towards the Sender. Section 4 provides operational guide on how to determine the scope of IP traffic packets that need to be counted. Note that the Reflector test packets are not counted.

In authenticated and encrypted modes, the S_TxC, R_RxC and R_TxC are respectively followed by a new 12 octets MBZ (MUST be zero) field to make it 16-octet aligned, which is required for authentication and encryption.

The intention of embedding S_TxC, R_RxC and R_TxC in the Session-Reflector test packets is for the Session-Sender to calculate direct loss of IP traffic, and the loss calculation algorithm is described in Section 3.3.

When the Symmetrical Size mode defined in [RFC6038] is also selected, basing on what's specified in Section 5.2.2 of [RFC6038], the Session-Reflector packet format would follow Figure 3.

When the Reflect Octets mode defined in [RFC6038] is also selected, S_TxC, R_RxC and R_TxC SHOULD be embedded in the Session-Reflector Packet formatted in Section 5.2.1 of [RFC6038], with the same position as depicted in Figure 3.

When both the Symmetrical Size mode and the Reflect Octets mode are also selected, S_TxC, R_RxC and R_TxC SHOULD be embedded in the Session- Reflector Packet formatted in Section 5.2.1 of [RFC6038], with the same position as depicted in Figure 3.

3.3. Traffic Loss Calculation

Upon receiving a Reflector Test Packet, the Session-Sender uses the following values to make loss calculation:

o Received S_TxC, R_RxC and R_TxC values embedded in Reflector Test Packet and local counter S_RxC value at the time this Reflector Test Packet was received. These values are represented as S_TxC[n], R_RxC[n], R_TxC[n], and S_RxC[n], where n is the reception time of the current Reflector Test Packet.

o Previous Received S_TxC, R_RxC and R_TxC values embedded in Reflector Test Packet and local counter S_RxC value at the time the previous Reflector Test Packet was received. These values are represented as S_TxC[n-1], R_RxC[n-1], R_TxC[n-1], and S_RxC[n-1], where n-1 is the reception time of the previous Reflector Test Packet.

The formulas for calculating the far-end loss, near-end loss, far-end loss rate and near-end loss rate are as following:

o Far-end loss: F_Loss[n-1,n] = (S_TxC[n]-S_TxC[n-1])-(R_RxC[n]-R_RxC[n-1])

o Near-end loss: N_Loss[n-1,n] = (R_TxC[n]-R_TxC[n-1])-(S_RxC[n]-S_RxC[n-1])

o Far-end loss rate: F_LossRate[n-1,n] = F_Loss[n-1,n]/(S_TxC[n]-S_TxC[n-1])

o Near-end loss rate: N_LossRate[n-1,n] = N_Loss[n-1,n]/(R_TxC[n]-R_TxC[n-1])

Here far-end means the direction from the Session-Sender to the Session-Reflector and near-end means the direction from the Session-Reflector to the Session-Sender.

4. Operational Guide

In order to make meaningful loss measurement, in general, the scope of IP traffic packets that need to be counted, i.e. the IP traffic packets counting rules, should be provisioned before starting Test Sessions, and the provisioned arguments usually include ingress port, source IP address, destination IP address, IP DSCP and UDP port number. For the scenarios where the exact source/destination IP address and IP DSCP of IP traffic can be known, such as mobile backhaul, the Test Packets should use the same source/destination IP address and IP DSCP as IP traffic, and it shall result in more accurate measurements.

5. Security Considerations

Use of direct loss measurement in a test session does not appear to introduce any additional security threat to hosts that communicate with TWAMP as defined in [RFC5357]. The security considerations that apply to any active measurement of live networks are relevant here as well. See the Security Considerations sections in [RFC4656] and [RFC5357].

6. IANA Considerations

In the TWAMP-Modes registry defined in [RFC5618], a new Direct Loss Measurement Capability is requested from IANA as follows:

Bit Pos	Description	Semantics Definition	Reference
10	Direct Loss Measurement Capability	Section 2	This Document

Table 1: New Direct Loss Measurement Capability

7. Acknowledgements

The authors would like to thank Greg Mirsky and Guo Jun for their valuable comments.

8. Normative References

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

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/info/rfc4656>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC5618] Morton, A. and K. Hedayat, "Mixed Security Mode for the Two-Way Active Measurement Protocol (TWAMP)", RFC 5618, DOI 10.17487/RFC5618, August 2009, <https://www.rfc-editor.org/info/rfc5618>.

[RFC6038] Morton, A. and L. Ciavattone, "Two-Way Active Measurement Protocol (TWAMP) Reflect Octets and Symmetrical Size Features", RFC 6038, DOI 10.17487/RFC6038, October 2010, <https://www.rfc-editor.org/info/rfc6038>.

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

Authors' Addresses

Xiao Min
ZTE
Nanjing
CN

Phone: +86 25 88016576
Email: xiao.min2@zte.com.cn

Dou Zhanwei
ZTE
Nanjing
CN

Phone: +86 25 52874656
Email: dou.zhanwei@zte.com.cn