Geneve encapsulation for In-situ OAM Data
draft-brockners-ippm-ioam-geneve-01

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 outlines how IOAM data fields are encapsulated in Geneve.

Status of This Memo

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

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

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

In-situ OAM (IOAM) records OAM information within the packet while the packet traverses a particular network domain. The term "in-situ" refers to the fact that the IOAM data fields are added to the data packets rather than being sent within packets specifically dedicated to OAM. This document defines how IOAM data fields are transported as part of the Geneve [I-D.ietf-nvo3-geneve] encapsulation. The IOAM data fields are defined in [I-D.ietf-ippm-ioam-data].
2. Conventions

2.1. Requirement 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 [RFC2119].

2.2. Abbreviations

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

Geneve: Generic Network Virtualization Encapsulation

3. IOAM Data Field Encapsulation in Geneve

Geneve is defined in [I-D.ietf-nvo3-geneve]. IOAM data fields are carried in the Geneve header as a tunnel option, using a single Geneve Option Class TBD_IOAM. The different IOAM data fields defined in [I-D.ietf-ippm-ioam-data] are added as TLVs using that Geneve Option Class. In an administrative domain where IOAM is used, insertion of the IOAM header in Geneve is enabled at the Geneve tunnel endpoints, which also serve as IOAM encapsulating/decapsulating nodes by means of configuration.
The Geneve header and fields are defined in [I-D.ietf-nvo3-geneve].
The Geneve Option Class value for use with IOAM is TBD_IOAM.

The fields related to the encapsulation of IOAM data fields in Geneve are defined as follows:

Option Class: 16-bit unsigned integer that determines the IOAM option class. The value is from the IANA registry setup for Geneve option classes as defined in [I-D.ietf-nvo3-geneve].

Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

R (3 bits): Option control flags reserved for future use. MUST be zero on transmission and ignored on receipt.

Length: 5-bit unsigned integer. Length of the IOAM HDR in 4-octet units.

IOAM Option and Data Space: IOAM option header and data is present as defined by the Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].
Multiple IOAM options MAY be included within the Geneve encapsulation. For example, if a Geneve encapsulation contains two IOAM options before a data payload, there would be two fields with TBD_IOAM Option Class each, differentiated by the Type field which specifies the type of the IOAM data included.

4. Considerations

This section summarizes a set of considerations on the overall approach taken for IOAM data encapsulation in Geneve, as well as deployment considerations.

4.1. Discussion of the encapsulation approach

This section is to support the working group discussion in selecting the most appropriate approach for encapsulating IOAM data fields in Geneve.

An encapsulation of IOAM data fields in Geneve should be friendly to an implementation in both hardware as well as software forwarders and support a wide range of deployment cases, including large networks that desire to leverage multiple IOAM data fields at the same time.

Hardware and software friendly implementation: Hardware forwarders benefit from an encapsulation that minimizes iterative look-ups of fields within the packet: Any operation which looks up the value of a field within the packet, based on which another lookup is performed, consumes additional gates and time in an implementation - both of which are desired to be kept to a minimum. This means that flat TLV structures are to be preferred over nested TLV structures. IOAM data fields are grouped into three option categories: Trace, proof-of-transit, and edge-to-edge. Each of these three options defines a TLV structure. A hardware-friendly encapsulation approach avoids grouping these three option categories into yet another TLV structure, but would rather carry the options as a serial sequence.

Total length of the IOAM data fields: The total length of IOAM data can grow quite large in case multiple different IOAM data fields are used and large path-lengths need to be considered. If for example an operator would consider using the IOAM trace option and capture node-id, app_data, egress/ingress interface-id, timestamp seconds, timestamps nanoseconds at every hop, then a total of 20 octets would be added to the packet at every hop. In case this particular deployment would have a maximum path length of 15 hops in the IOAM domain, then a maximum of 300 octets of IOAM data were to be encapsulated in the packet.
Concerns with the current encapsulation approach:

Hardware support: Using Geneve tunnel options to encapsulate IOAM data fields leads to a nested TLV structure. Each IOAM data field option (trace, proof-of-transit, and edge-to-edge) represents a type, with the different IOAM data fields being TLVs within this particular option type. Nested TLVs require iterative look-ups, a fact that creates potential challenges for implementations in hardware. It would be desirable to offer a way to encapsulate IOAM in a way that keeps TLV nesting to a minimum.

Length: Geneve tunnel option length is a 5-bit field in the current specification [I-D.ietf-nvo3-geneve] resulting in a maximum option length of 128 (2^5 x 4) octets which constrains the use of IOAM to either small domains or a few IOAM data fields only. Support for large domains with a variety of IOAM data fields would be desirable.

4.2. IOAM and the use of the Geneve O-bit

[I-D.ietf-nvo3-geneve] defines an "O bit" for OAM packets. Per [I-D.ietf-nvo3-geneve] the O bit indicates that the packet contains a control message instead of data payload. Packets that carry IOAM data fields in addition to regular data payload / customer traffic must not set the O bit. Packets that carry only IOAM data fields without any payload must set the O bit.

4.3. Transit devices

If IOAM is deployed in domains where UDP port numbers are not controlled and do not have a domain-wide meaning, such as on the global Internet, transit devices MUST NOT attempt to modify the IOAM data contained in the IOAM option class. In case UDP port numbers are not controlled there might be UDP packets, which leverage the UDP port number that Geneve utilizes, i.e. 6081, but the payload of these packets isn’t Geneve. The scenario and associated reasoning is discussed in [RFC7605] which states that "it is important to recognize that any interpretation of port numbers -- except at the endpoints -- may be incorrect, because port numbers are meaningful only at the endpoints."

5. IANA Considerations

IANA is requested to allocate a Geneve "option class" numbers for IOAM:
6. Security Considerations

The security considerations of Geneve are discussed in [I-D.ietf-nvo3-geneve], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srihari Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Stefano Previdi, Hemant Singh, Erik Nordmark, LJ Wobker, and Andrew Yourtchenko for the comments and advice.

8. Normative References

[I-D.ietf-ippm-ioam-data]

[I-D.ietf-nvo3-geneve]


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

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
7200-11 Kit Creek Road
Research Triangle Park, NC 27709
United States

Email: cpignata@cisco.com
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 outlines how IOAM data fields are encapsulated in VXLAN-GPE.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."
In-situ OAM spreads 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 fields are added to the data packets rather than being sent within packets specifically dedicated to OAM. This document defines how OAM data fields are transported as part of the VXLAN-GPE [I-D.ietf-nvo3-vxlan-gpe] encapsulation. The OAM data fields are defined in [I-D.ietf-ippm-ioam-data]. An implementation of OAM which leverages VXLAN-GPE to carry the OAM data is available from the FD.io open source software project [FD.io].
2. Conventions

2.1. Requirement 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 [RFC2119].

2.2. Abbreviations

Abbreviations used in this document:

IOAM: In-situ Operations, Administration, and Maintenance

OAM: Operations, Administration, and Maintenance

VXLAN-GPE: Virtual eXtensible Local Area Network, Generic Protocol Extension

3. IOAM Data Field Encapsulation in VXLAN-GPE

VXLAN-GPE is defined in [I-D.ietf-nvo3-vxlan-gpe]. IOAM data fields are carried in VXLAN-GPE using a next protocol value of TBD_IOAM. An IOAM header is added containing the different IOAM data fields defined in [I-D.ietf-ippm-ioam-data]. In an administrative domain where IOAM is used, insertion of the IOAM header in VXLAN-GPE is enabled at the VXLAN-GPE tunnel endpoints, which also serve as IOAM encapsulating/decapsulating nodes by means of configuration.
The VXLAN-GPE header and fields are defined in [I-D.ietf-nvo3-vxlan-gpe]. The VXLAN Next Protocol value for IOAM is TBD_IOAM.

The IOAM related fields in VXLAN-GPE are defined as follows:

- **IOAM-Type**: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].
- **IOAM HDR len**: 8-bit unsigned integer. Length of the IOAM HDR in 4-octet units.
- **Reserved**: 8-bit reserved field MUST be set to zero upon transmission and ignored upon receipt.
- **Next Protocol**: 8-bit unsigned integer that determines the type of header following IOAM protocol. The value is from the IANA.
registry setup for VXLAN GPE Next Protocol defined in [I-D.ietf-nvo3-vxlan-gpe].

IOAM Option and Data Space: IOAM option header and data is present as specified by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

Multiple IOAM options MAY be included within the VXLAN-GPE encapsulation. For example, if a VXLAN-GPE encapsulation contains two IOAM options before a data payload, the Next Protocol field of the first IOAM option will contain the value of TBD_IOAM, while the Next Protocol field of the second IOAM option will contain the VXLAN "Next Protocol" number indicating the type of the data payload.

4. Considerations

This section summarizes a set of considerations on the overall approach taken for IOAM data encapsulation in VXLAN-GPE, as well as deployment considerations.

4.1. Discussion of the encapsulation approach

This section is to support the working group discussion in selecting the most appropriate approach for encapsulating IOAM data fields in VXLAN-GPE.

An encapsulation of IOAM data fields in VXLAN-GPE should be friendly to an implementation in both hardware as well as software forwarders. Hardware forwarders benefit from an encapsulation that minimizes iterative look-ups of fields within the packet: Any operation which looks up the value of a field within the packet, based on which another lookup is performed, consumes additional gates and time in an implementation - both of which are desired to be kept to a minimum. This means that flat TLV structures are to be preferred over nested TLV structures. IOAM data fields are grouped into three option categories: Trace, proof-of-transit, and edge-to-edge. Each of these three options defines a TLV structure. A hardware-friendly encapsulation approach avoids grouping these three option categories into yet another TLV structure, but would rather carry the options as a serial sequence.

Two approaches for encapsulating IOAM data fields in VXLAN-GPE could be considered:

1. Use a single GPE protocol type for all IOAM types: IOAM would receive a single GPE protocol type code point. A "sub-type" field would then specify what IOAM options type (trace, proof-of-transit, edge-to-edge) is carried.
2. Use one GPE protocol type per IOAM options type: Each IOAM data field option (trace, proof-of-transit, and edge-to-edge) would be specified by its own "next protocol", i.e. each IOAM options type becomes its own GPE protocol type with a dedicated code point. This implies that in case additional IOAM option types would be added in the future, additional GPE protocol type code points would need to be allocated.

The first option has been chosen here. Multiple back-to-back IOAM options can be encoded as a succession of IOAM headers, with the same single GPE protocol type appearing as the next protocol before each IOAM header, but different sub-types within each IOAM header.

4.2. IOAM and the use of the VXLAN O-bit

[I-D.ietf-nvo3-vxlan-gpe] defines an "O bit" for OAM packets. Per [I-D.ietf-nvo3-vxlan-gpe] the O bit indicates that the packet contains an OAM message instead of data payload. Packets that carry IOAM data fields in addition to regular data payload / customer traffic must not set the O bit. Packets that carry only IOAM data fields without any payload must set the O bit.

4.3. Transit devices

If IOAM is deployed in domains where UDP port numbers are not controlled and do not have a domain-wide meaning, such as on the global Internet, transit devices MUST NOT attempt to modify the IOAM data contained in the IOAM header following the VXLAN-GPE header. In case UDP port numbers are not controlled there might be UDP packets specifying the same UDP port number that VXLAN-GPE utilizes, i.e. 4790, but with a payload that is not VXLAN-GPE. The scenario and associated reasoning is discussed in [RFC7605] which states that "it is important to recognize that any interpretation of port numbers -- except at the endpoints -- may be incorrect, because port numbers are meaningful only at the endpoints."

5. IANA Considerations

IANA is requested to allocate a protocol number for the following VXLAN-GPE "Next Protocols" related to IOAM:

<table>
<thead>
<tr>
<th>Next Protocol</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>TBD_IOAM</td>
<td>This document</td>
</tr>
</tbody>
</table>

6. Security Considerations

The security considerations of VXLAN-GPE are discussed in [I-D.ietf-nvo3-vxlan-gpe], and the security considerations of IOAM in general are discussed in [I-D.ietf-ippm-ioam-data].

IOAM is considered a "per domain" feature, where one or several operators decide on leveraging and configuring IOAM according to their needs. Still, operators need to properly secure the IOAM domain to avoid malicious configuration and use, which could include injecting malicious IOAM packets into a domain.

7. Acknowledgements

The authors would like to thank Eric Vyncke, Nalini Elkins, Srinath Raghavan, Ranganathan T S, Karthik Babu Harichandra Babu, Akshaya Nadahalli, Stefano Previdi, Hemant Singh, Erik Nordmark, LJ Wobker, and Andrew Yourtchenko for the comments and advice.

8. References

8.1. Normative References


8.2. Informative References


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

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@cisco.com

Barak Gafni
Mellanox Technologies, Inc.
350 Oakmead Parkway, Suite 100
Sunnyvale, CA  94085
U.S.A.

Email: gbarak@mellanox.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA  94025
US

Email: petr@fb.com

Mickey Spiegel
Barefoot Networks
2185 Park Boulevard
Palo Alto, CA  94306
US

Email: mspiegel@barefootnetworks.com
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.
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 (Broadcast Unknown 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.
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-R4), (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:
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 measurements 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


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 caught 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 caught 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 measurements 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


14.2. Informative References

Internet-Draft                Multipoint AM                    June 2018

[I-D.mizrahi-ippm-compact-alternate-marking]


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
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/.
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:
o The version field is 4

o The Internet Header Length (IHL) value is >= 5; the checksum is correct.

o Its total length as given in the IPv4 header corresponds to the size of the IPv4 header plus the size of the payload.

o 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.

o 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:

o The version field is 6.

o 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.

o The payload length value for this packet (including Extension Headers) conforms to the IPv6 specifications.

o 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.

o 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).

o 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, [RFC4021]) 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.
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


9.2. Informative References

[I-D.ietf-ippm-ioam-data]

[IANA-6P]

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

* Revised implementation of Passive TCP RTT metrics in section 10 (from comments).

* remaining question on DNS measurement method(s)

Still need: Add MBM metric entry.

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 January 1, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the
document authors. All rights reserved.

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

Table of Contents

1.  Introduction ............................................. 7
2.  Scope .................................................. 8
3.  Registry Categories and Columns ......................... 8
4.  UDP Round-trip Latency and Loss Registry Entries .......... 9
   4.1.  Summary .............................................. 10
   4.1.1.  ID (Identifier) ............................... 10
   4.1.2.  Name .............................................. 10
   4.1.3.  URIs .............................................. 10
   4.1.4.  Description ....................................... 10
   4.1.5.  Change Controller .............................. 10
   4.1.6.  Version (of Registry Format) ................... 10
   4.2.  Metric Definition .................................. 11
   4.2.1.  Reference Definition ........................... 11
   4.2.2.  Fixed Parameters ............................... 12
   4.3.  Method of Measurement ............................. 12
   4.3.1.  Reference Method ............................... 13
   4.3.2.  Packet Stream Generation ...................... 14
   4.3.3.  Traffic Filtering (observation) Details ........ 14
   4.3.4.  Sampling Distribution ......................... 15
   4.3.5.  Run-time Parameters and Data Format ............ 15
   4.3.6.  Roles ............................................. 15
   4.4.  Output .............................................. 16
   4.4.1.  Type .............................................. 16
   4.4.2.  Reference Definition ........................... 16
   4.4.3.  Metric Units ..................................... 17
   4.4.4.  Calibration ...................................... 17
   4.5.  Administrative items ............................... 18
   4.5.1.  Status ............................................ 18
   4.5.2.  Requestor (keep?) .............................. 18

4.5.3. Revision ........................................... 18
4.5.4. Revision Date ..................................... 18
4.6. Comments and Remarks ................................ 18
5. Packet Delay Variation Registry Entry .................. 18
  5.1. Summary .............................................. 18
      5.1.1. ID (Identifier) ................................... 18
      5.1.2. Name .............................................. 18
      5.1.3. URIs .............................................. 19
      5.1.4. Description ....................................... 19
      5.1.5. Change Controller ................................ 19
      5.1.6. Version (of Registry Format) ..................... 19
  5.2. Metric Definition .................................... 19
      5.2.1. Reference Definition ............................. 19
      5.2.2. Fixed Parameters ................................. 20
  5.3. Method of Measurement ................................ 21
      5.3.1. Reference Method ................................ 21
      5.3.2. Packet Stream Generation ......................... 22
      5.3.3. Traffic Filtering (observation) Details ........ 22
      5.3.4. Sampling Distribution ............................ 23
      5.3.5. Run-time Parameters and Data Format ............. 23
      5.3.6. Roles ............................................ 23
  5.4. Output ................................................. 23
      5.4.1. Type .............................................. 24
      5.4.2. Reference Definition ............................. 24
      5.4.3. Metric Units ...................................... 24
      5.4.4. Calibration ....................................... 25
  5.5. Administrative items .................................. 25
      5.5.1. Status ........................................... 25
      5.5.2. Requestor (keep?) ............................... 26
      5.5.3. Revision ......................................... 26
      5.5.4. Revision Date .................................... 26
  5.6. Comments and Remarks ................................ 26
  6. DNS Response Latency and Loss Registry Entries ......... 26
  6.1. Summary .............................................. 26
      6.1.1. ID (Identifier) ................................... 27
      6.1.2. Name .............................................. 27
      6.1.3. URI .............................................. 27
      6.1.4. Description ....................................... 27
      6.1.5. Change Controller ................................ 27
      6.1.6. Version (of Registry Format) ..................... 27
  6.2. Metric Definition .................................... 27
      6.2.1. Reference Definition ............................. 28
      6.2.2. Fixed Parameters ................................. 28
  6.3. Method of Measurement ................................ 30
      6.3.1. Reference Method ................................ 30
      6.3.2. Packet Stream Generation ......................... 32
      6.3.3. Traffic Filtering (observation) Details ........ 32
      6.3.4. Sampling Distribution ............................ 32
8.2.2. Fixed Parameters .................................. 49
8.3. Method of Measurement ................................ 50
  8.3.1. Reference Method .................................. 51
  8.3.2. Packet Stream Generation ............................ 51
  8.3.3. Traffic Filtering (observation) Details .......... 52
  8.3.4. Sampling Distribution .............................. 52
  8.3.5. Run-time Parameters and Data Format ............. 52
  8.3.6. Roles ............................................ 53

8.4. Output ............................................... 53
  8.4.1. Type .............................................. 53
  8.4.2. Reference Definition ................................ 53
  8.4.3. Metric Units ...................................... 56
  8.4.4. Calibration ....................................... 56

8.5. Administrative items ................................ 57
  8.5.1. Status ............................................ 57
  8.5.2. Requestor (keep?) .................................. 57
  8.5.3. Revision ........................................... 57
  8.5.4. Revision Date ...................................... 58

8.6. Comments and Remarks ................................ 58

9. ICMP Round-trip Latency and Loss Registry Entries .......... 58
  9.1. Summary ............................................. 58
    9.1.1. ID (Identifier) .................................. 58
    9.1.2. Name ............................................. 58
    9.1.3. URIs ............................................. 59
    9.1.4. Description ...................................... 59
    9.1.5. Change Controller ................................ 59
    9.1.6. Version (of Registry Format) ..................... 59
  9.2. Metric Definition ................................... 59
    9.2.1. Reference Definition ................................ 59
    9.2.2. Fixed Parameters .................................. 60
  9.3. Method of Measurement ................................. 61
    9.3.1. Reference Method .................................. 61
    9.3.2. Packet Stream Generation ............................ 62
    9.3.3. Traffic Filtering (observation) Details .......... 63
    9.3.4. Sampling Distribution .............................. 63
    9.3.5. Run-time Parameters and Data Format ............. 63
    9.3.6. Roles ............................................ 64

9.4. Output ............................................... 64
  9.4.1. Type .............................................. 64
  9.4.2. Reference Definition ................................ 64
  9.4.3. Metric Units ...................................... 66
  9.4.4. Calibration ....................................... 67

9.5. Administrative items ................................ 67
  9.5.1. Status ............................................ 67
  9.5.2. Requestor (keep?) .................................. 67
  9.5.3. Revision ........................................... 67
  9.5.4. Revision Date ...................................... 67

9.6. Comments and Remarks ................................ 67
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.</td>
<td>TCP Round-Trip Delay and Loss Registry Entries</td>
</tr>
<tr>
<td>10.1.</td>
<td>Summary</td>
</tr>
<tr>
<td>10.1.1.</td>
<td>ID (Identifier)</td>
</tr>
<tr>
<td>10.1.2.</td>
<td>Name</td>
</tr>
<tr>
<td>10.1.3.</td>
<td>URIs</td>
</tr>
<tr>
<td>10.1.4.</td>
<td>Description</td>
</tr>
<tr>
<td>10.1.5.</td>
<td>Change Controller</td>
</tr>
<tr>
<td>10.1.6.</td>
<td>Version (of Registry Format)</td>
</tr>
<tr>
<td>10.2.</td>
<td>Metric Definition</td>
</tr>
<tr>
<td>10.2.1.</td>
<td>Reference Definitions</td>
</tr>
<tr>
<td>10.2.2.</td>
<td>Fixed Parameters</td>
</tr>
<tr>
<td>10.3.</td>
<td>Method of Measurement</td>
</tr>
<tr>
<td>10.3.1.</td>
<td>Reference Methods</td>
</tr>
<tr>
<td>10.3.2.</td>
<td>Packet Stream Generation</td>
</tr>
<tr>
<td>10.3.3.</td>
<td>Traffic Filtering (observation) Details</td>
</tr>
<tr>
<td>10.3.4.</td>
<td>Sampling Distribution</td>
</tr>
<tr>
<td>10.3.5.</td>
<td>Run-time Parameters and Data Format</td>
</tr>
<tr>
<td>10.3.6.</td>
<td>Roles</td>
</tr>
<tr>
<td>10.4.</td>
<td>Output</td>
</tr>
<tr>
<td>10.4.1.</td>
<td>Type</td>
</tr>
<tr>
<td>10.4.2.</td>
<td>Reference Definition</td>
</tr>
<tr>
<td>10.4.3.</td>
<td>Metric Units</td>
</tr>
<tr>
<td>10.4.4.</td>
<td>Calibration</td>
</tr>
<tr>
<td>10.5.</td>
<td>Administrative items</td>
</tr>
<tr>
<td>10.5.1.</td>
<td>Status</td>
</tr>
<tr>
<td>10.5.2.</td>
<td>Requestor (keep?)</td>
</tr>
<tr>
<td>10.5.3.</td>
<td>Revision</td>
</tr>
<tr>
<td>10.5.4.</td>
<td>Revision Date</td>
</tr>
<tr>
<td>10.6.</td>
<td>Comments and Remarks</td>
</tr>
<tr>
<td>11.</td>
<td>ver08 BLANK Registry Entry</td>
</tr>
<tr>
<td>11.1.</td>
<td>Summary</td>
</tr>
<tr>
<td>11.1.1.</td>
<td>ID (Identifier)</td>
</tr>
<tr>
<td>11.1.2.</td>
<td>Name</td>
</tr>
<tr>
<td>11.1.3.</td>
<td>URIs</td>
</tr>
<tr>
<td>11.1.4.</td>
<td>Description</td>
</tr>
<tr>
<td>11.1.5.</td>
<td>Reference</td>
</tr>
<tr>
<td>11.1.6.</td>
<td>Change Controller</td>
</tr>
<tr>
<td>11.1.7.</td>
<td>Version (of Registry Format)</td>
</tr>
<tr>
<td>11.2.</td>
<td>Metric Definition</td>
</tr>
<tr>
<td>11.2.1.</td>
<td>Reference Definition</td>
</tr>
<tr>
<td>11.2.2.</td>
<td>Fixed Parameters</td>
</tr>
<tr>
<td>11.3.</td>
<td>Method of Measurement</td>
</tr>
<tr>
<td>11.3.1.</td>
<td>Reference Method</td>
</tr>
<tr>
<td>11.3.2.</td>
<td>Packet Stream Generation</td>
</tr>
<tr>
<td>11.3.3.</td>
<td>Traffic Filtering (observation) Details</td>
</tr>
<tr>
<td>11.3.4.</td>
<td>Sampling Distribution</td>
</tr>
<tr>
<td>11.3.5.</td>
<td>Run-time Parameters and Data Format</td>
</tr>
<tr>
<td>11.3.6.</td>
<td>Roles</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>Metric Definition</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>Method of Measurement</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>Administrative Information</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
</tr>
<tr>
<td><strong>Comments and Remarks</strong></td>
</tr>
</tbody>
</table>

#### 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)

<insert a numeric identifier, an integer, TBD>

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

4.1.2. Name

<insert name according to metric naming convention>

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

<Full bibliographic reference to an immutable doc.>


[RFC2681]

<specific section reference and additional clarifications, if needed>

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.


[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

<list and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed>

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)

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

- UDP Payload
  * total of 100 bytes

Other measurement parameters:

- 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).

<section reference>.

NA
4.3.4. Sampling Distribution

<insert time distribution details, or how this is diff from the filter>

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.

<list of run-time parameters, and their data formats>

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

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

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

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

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

<describe the reference data format for each type of result>

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_RFCXXXsecY_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_RFCXXXsecY_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

<insert units for the measured results, and the reference specification>.

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 or deprecated>

4.5.2. Requestor (keep?)

name or RFC, etc.

4.5.3. Revision

1.0

4.5.4. Revision Date

YYYY-MM-DD

4.6. Comments and Remarks

Additional (Informational) details for this entry

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.

<skipping some Summary columns for now>

5.1.1. ID (Identifier)

<insert numeric identifier, an integer>

5.1.2. Name

<insert name according to metric naming convention>

OWPDV_Active_IP-UDP-Periodic_RFCXXXsecY_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

<org or person >

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

<Full bibliographic reference to an immutable doc.>


 spécifique section reference and additional clarifications, if needed>
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

<list and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed>

- 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

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

NA
5.3.4. Sampling Distribution

<insert time distribution details, or how this is diff from the filter>

NA

5.3.5. Run-time Parameters and Data Format

<list of run-time parameters, and their data formats>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src</td>
<td>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])</td>
</tr>
<tr>
<td>Dst</td>
<td>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])</td>
</tr>
<tr>
<td>T0</td>
<td>a time, the start of a measurement interval, (format &quot;date-and-time&quot; 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 &quot;all-zeros&quot;, 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.</td>
</tr>
<tr>
<td>Tf</td>
<td>a time, the end of a measurement interval, (format &quot;date-and-time&quot; 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 &quot;all-zeros&quot;, a end time date is ignored and Tf is interpreted as the Duration of the measurement interval.</td>
</tr>
</tbody>
</table>

5.3.6. Roles

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

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src</td>
<td>launches each packet to Dst.</td>
</tr>
<tr>
<td>Dst</td>
<td>waits for each packet from Src.</td>
</tr>
</tbody>
</table>

5.4. Output

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

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

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

<insert units for the measured results, and the reference specification>.

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).

\textbf{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 or deprecated>
5.5.2. Requestor (keep?)

    <name of individual or RFC, etc.>

5.5.3. Revision

    1.0

5.5.4. Revision Date

    YYYY-MM-DD

5.6. Comments and Remarks

    <Additional (Informational) details for this entry>

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

@@@ comment from Brian: there is an interesting method for DNS measurement by encoding information in the query itself. It is a question of what exactly we are trying to measure: specific RR, or the infrastructure itself. (at this time we measure a specific RR).

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

<insert name according to metric naming convention>

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

<Full bibliographic reference to an immutable doc.>

[RFC1035]

[RFC2681]

<specific section reference and additional clarifications, if needed>

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).

[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

<list and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed>
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)

- UDP header values:
  - Source port: 53
  - Destination port: 53
  - Checksum: the checksum must be calculated and included in the header

- 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:

o 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

<for metric, insert relevant section references and supplemental info>

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 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. Therefore, 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. Sequence number is part of the payload described under Fixed Parameters.

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.

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

This would require support of ID generation and population in the Message. An alternative would be to use a random Source port on the Query Message, but we would choose ONE before proceeding.

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 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.

<list of generation parameters and section/spec references if needed>

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).

<section reference>.

NA

6.3.4. Sampling Distribution

<insert time distribution details, or how this is diff from the filter>

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.

<list of run-time parameters, and their data formats>
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

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

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

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

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

<describe the data format for each type of result>

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

<insert units for the measured results, and the reference specification>.

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 or deprecated>

6.5.2. Requestor

name or RFC, etc.

6.5.3. Revision

1.0
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)

<insert numeric identifier, an integer, one corresponding to each name below>

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

7.1.2. Name

<insert name according to metric naming convention>

OWDelay_Active_IP-UDP-Poisson-Payload250B_RFCXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- 95Percentile
- Mean
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 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:


[RFC7679]


[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:


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

List and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed.
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.

ünde reference method and section/spec references if needed>
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.

<list of run-time parameters, and their data formats>
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

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

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

See subsection titles below for Types.

7.4.2. Reference Definition

<describe the data format for each type of result>

For all output types ---
To 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

<insert units for the measured results, and the reference
specification>.

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 or deprecated>

#### 7.5.2. Requestor (keep?)

name or RFC, etc.

#### 7.5.3. Revision

1.0
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)

<insert numeric identifier, an integer, one corresponding to each name below>

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

8.1.2. Name

<insert name according to metric naming convention>

OWDelay_Active_IP-UDP-Periodic-
 Payload142B_RFCXXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- 95Percentile
- Mean
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:


[RFC7679]


[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:


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

<List and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed>
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.
  - 142 octets total, including the TWAMP format

Other measurement parameters:

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

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
dT the duration of the interval for allowed sample start times
T0 the actual start time of the periodic stream

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.

<list of run-time parameters, and their data formats>

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.

should Periodic run-time params be fixed instead? Probably yes if modeling a specific version of tests. Note in the NAME, i.e. Poisson3.3

8.3.6. 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.

8.4. Output

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

8.4.1. Type

See subsection titles in Reference Definition for Latency Types.

8.4.2. Reference Definition

<describe the data format for each type of result>

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)
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:

\[
\text{Max} = (\text{FiniteDelay}[j])
\]

such that for some index, j, where 1 <= j <= N  
\[
\text{FiniteDelay}[j] >= \text{FiniteDelay}[n] \text{ for all n}
\]

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

<insert units for the measured results, and the reference specification>.

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 or deprecated>

8.5.2. Requestor (keep?)

  name or RFC, etc.

8.5.3. Revision

  1.0
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. 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)

<insert a numeric identifier, an integer, TBD>

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

9.1.2. Name

<insert name according to metric naming convention>

RTDelay_Active_IP-ICMP-SendOnRcv_RFCXXXXsecY_Seconds_<statistic>

where <statistic> is one of:

- Mean
- Min
- Max
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

<Full bibliographic reference to an immutable doc.>


Morton, et al. Expires January 1, 2019
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.


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)

- 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

<for metric, insert relevant section references and supplemental info>

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:

\[ incT \text{ the nominal duration of inter-packet interval, first bit to first bit} \]
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).

<section reference>.

NA

9.3.4. Sampling Distribution

<insert time distribution details, or how this is diff from the filter>

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.

<list of run-time parameters, and their data formats>

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

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

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

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

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

<describe the data format for each type of result>

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
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:

\[ \text{Max} = (\text{FiniteDelay} [j]) \]

such that for some index, \(j\), where \(1 \leq j \leq N\)

\(\text{FiniteDelay}[j] \geq \text{FiniteDelay}[n]\) 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

<insert units for the measured results, and the reference specification>.

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 or deprecated>

9.5.2. Requestor (keep?)

name or RFC, etc.

9.5.3. Revision

1.0

9.5.4. Revision Date

YYYY-MM-DD

9.6. Comments and Remarks

Additional (Informational) details for this entry
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)

<insert a numeric identifier, an integer, TBD>

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

10.1.2. Name

<insert name according to metric naming convention>

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 opportunity to compose a single RTDelay on the TCP Hand Shake.

RTRLoss_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

<Full bibliographic reference to an immutable doc.>

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

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 the two single-direction counts of inferred loss is:

\[ \text{RTLoss} = \text{RTL}_\text{fwd} + \text{RTL}_\text{rev} \]

10.2.2. Fixed Parameters

- Traffic Filters:
  - IPv4 header values:
    - DSCP: set to 0
    - Protocol: Set to 06 (TCP)
  - IPv6 header values:
    - DSCP: set to 0
    - Protocol: Set to 06 (TCP)
  - TCP header values:
    - Flags: ACK, SYN, FIN, @@@@ others??
    - Timestamp Option (TSopt): Set
      - Kind: 8
      - Length: 10 bytes

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.
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

<insert time distribution details, or how this is diff from the filter>

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.

<list of run-time parameters, and their data formats>

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

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

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

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

See subsection titles in Reference Definition for RTDelay Types.

For RTLoss -- the count of lost packets.

10.4.2. Reference Definition

<describe the data format for each type of result>

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:

\[
\text{Max} = (\text{FiniteDelay}[j])
\]

such that for some index, \( j \), where \( 1 \leq j \leq N \)
\[
\text{FiniteDelay}[j] \geq \text{FiniteDelay}[n] \text{ for all } n
\]

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

<insert units for the measured results, and the reference specification>.

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 or deprecated>

10.5.2. Requestor (keep?)
   name or RFC, etc.

10.5.3. Revision
   1.0

10.5.4. Revision Date
   YYYY-MM-DD

10.6. Comments and Remarks
   Additional (Informational) details for this entry

11. ver08 BLANK Registry Entry
   This section gives an initial registry entry for ....

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

11.1.1. ID (Identifier)
   <insert numeric identifier, an integer>

11.1.2. Name
   <insert name according to metric naming convention>

11.1.3. URIs
   URI: Prefix urn:ietf:metrics:perf:<name>
   URL:

11.1.4. Description
   TBD.
11.1.5. Reference

<reference to the RFC of spec where the registry entry is defined>

11.1.6. Change Controller

<org or person>

11.1.7. Version (of Registry Format)

<currently 1.0>

11.2. 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.

11.2.1. Reference Definition

<Full bibliographic reference to an immutable doc.>

<specific section reference and additional clarifications, if needed>

11.2.2. Fixed Parameters

<list and specify Fixed Parameters, input factors that must be determined and embedded in the measurement system for use when needed>

11.3. 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.

11.3.1. Reference Method

<for metric, insert relevant section references and supplemental info>

11.3.2. Packet Stream Generation

<list of generation parameters and section/spec references if needed>
11.3.3. Traffic Filtering (observation) Details

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

11.3.4. Sampling Distribution

<insert time distribution details, or how this is diff from the filter>

11.3.5. Run-time Parameters and Data Format

<list of run-time parameters, and any reference(s)>

11.3.6. Roles

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

11.4. Output

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

11.4.1. Type

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

11.4.2. Reference Definition

<pointer to section/spec where output type/format is defined>

11.4.3. Metric Units

<insert units for the measured results, and the reference specification>

11.4.4. Calibration

<describe the error calibration, a way to indicate that the results were collected in a calibration mode of operation, and a way to report internal status metrics related to calibration, such as time offset>
11.5. Administrative items

11.5.1. Status

$current or deprecated$

11.5.2. Requestor

$name of individual or Internet Draft, etc.$

11.5.3. Revision

1.0

11.5.4. Revision Date

$YYYY-MM-DD$

11.6. Comments and Remarks

Additional (Informational) details for this entry

12. Example RTCP-XR Registry Entry

This section is MAY BE DELETED or adapted before submission.

This section gives an example registry entry for the end-point metric described in RFC 7003 [RFC7003], for RTCP-XR Burst/Gap Discard Metric reporting.

12.1. Registry Indexes

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

12.1.1. Identifier

An integer having enough digits to uniquely identify each entry in the Registry.

12.1.2. Name

A metric naming convention is TBD.
12.1.3. URI

Prefix urn:ietf:metrics:param:<name>

12.1.4. Status

current

12.1.5. Requestor

Alcelip Mornuley

12.1.6. Revision

1.0

12.1.7. Revision Date

2014-07-04

12.1.8. Description

TBD.

12.1.9. Reference Specification(s)

[RFC3611][RFC4566][RFC6776][RFC6792][RFC7003]

12.2. 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. Section 3.2 of
[RFC7003] provides the reference information for this category.

12.2.1. Reference Definition

Packets Discarded in Bursts:

The total number of packets discarded during discard bursts. The
measured value is unsigned value. If the measured value exceeds
0xFFFFFD, the value 0xFFFFFE MUST be reported to indicate an over-
range measurement. If the measurement is unavailable, the value
0xFFFFFFFF MUST be reported.
12.2.2. Fixed Parameters

Fixed Parameters are input factors that must be determined and embedded in the measurement system for use when needed. The values of these parameters is specified in the Registry.

Threshold: 8 bits, set to value = 3 packets.

The Threshold is equivalent to Gmin in [RFC3611], i.e., the number of successive packets that must not be discarded prior to and following a discard packet in order for this discarded packet to be regarded as part of a gap. Note that the Threshold is set in accordance with the Gmin calculation defined in Section 4.7.2 of [RFC3611].

Interval Metric flag: 2 bits, set to value 11=Cumulative Duration

This field is used to indicate whether the burst/gap discard metrics are Sampled, Interval, or Cumulative metrics [RFC6792]:

I=10: Interval Duration - the reported value applies to the most recent measurement interval duration between successive metrics reports.

I=11: Cumulative Duration - the reported value applies to the accumulation period characteristic of cumulative measurements.

Senders MUST NOT use the values I=00 or I=01.

12.3. 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. For the Burst/Gap Discard Metric, it appears that the only guidance on methods of measurement is in Section 3.0 of [RFC7003] and its supporting references. Relevant information is repeated below, although there appears to be no section titled "Method of Measurement" in [RFC7003].

12.3.1. Reference Method

Metrics in this block report on burst/gap discard in the stream arriving at the RTP system. Measurements of these metrics are made at the receiving end of the RTP stream. Instances of this metrics block use the synchronization source (SSRC) to refer to the separate auxiliary Measurement Information Block [RFC6776], which describes measurement periods in use (see [RFC6776], Section 4.2).
This metrics block relies on the measurement period in the Measurement Information Block indicating the span of the report. Senders MUST send this block in the same compound RTCP packet as the Measurement Information Block. Receivers MUST verify that the measurement period is received in the same compound RTCP packet as this metrics block. If not, this metrics block MUST be discarded.

12.3.2. Stream Type and Stream Parameters

Since RTCP-XR Measurements are conducted on live RTP traffic, the complete description of the stream is contained in SDP messages that proceed the establishment of a compatible stream between two or more communicating hosts. See Run-time Parameters, below.

12.3.3. Output Type and Data Format

The output type defines the type of result that the metric produces.

- Value: Packets Discarded in Bursts
- Data Format: 24 bits
- Reference: Section 3.2 of [RFC7003]

12.3.4. Metric Units

The measured results are apparently expressed in packets, although there is no section of [RFC7003] titled "Metric Units".

12.3.5. 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. However, the values of these parameters is not specified in the Registry, rather these parameters are listed as an aid to the measurement system implementor or user (they must be left as variables, and supplied on execution).

The Data Format of each Run-time Parameter SHALL be specified in this column, to simplify the control and implementation of measurement devices.

- SSRC of Source: 32 bits As defined in Section 4.1 of [RFC3611].
- SDP Parameters: As defined in [RFC4566]
- Session description v= (protocol version number, currently only 0)
o= (originator and session identifier: username, id, version number, network address)

s= (session name: mandatory with at least one UTF-8-encoded character)

i=* (session title or short information) u=* (URI of description)

e=* (zero or more email address with optional name of contacts)

p=* (zero or more phone number with optional name of contacts)

c=* (connection information—not required if included in all media)

b=* (zero or more bandwidth information lines) One or more Time descriptions ("t=" and "r=" lines; see below)

z=* (time zone adjustments)

k=* (encryption key)

a=* (zero or more session attribute lines)

Zero or more Media descriptions (each one starting by an "m=" line; see below)

m= (media name and transport address)

i=* (media title or information field)

c=* (connection information—optional if included at session level)

b=* (zero or more bandwidth information lines)

k=* (encryption key)

a=* (zero or more media attribute lines—overriding the Session attribute lines)

An example Run-time SDP description follows:

v=0

o=jdoe 2890844526 2890842807 IN IP4 192.0.2.5

s=SDP Seminar i=A Seminar on the session description protocol
12.4. Comments and Remarks

TBD.

13. Revision History

This section may be removed for publication. It contains overview information on updates.

This draft replaced draft-mornuley-ippm-initial-registry.

In version 02, Section 4 has been edited to reflect recent discussion on the ippm-list: * Removed the combination or "Raw" and left 95th percentile. * Hanging Indent on Run-time parameters (Fixed parameters use bullet lists and other indenting formats. * Payload format for measurement has been removed. * Explanation of Conditional delay distribution.

Version 03 addressed Phil Eardley’s comments and suggestions in sections 1-4. and resolved the definition of Percentiles.

Version 04 * All section 4 parameters reference YANG types for alternate data formats. * Discussion has concluded that usecase(s) for machine parse-able registry columns are not needed.

Version 05 * Revised several Poisson streams to Periodic, sections 4 & 5. * Addition of ICMP (ping) metrics in section 9. * First implementation of Passive TCP RTT metrics in section 10.
14. Security Considerations

These registry entries represent no known security implications for Internet Security. Each referenced Metric contains a Security Considerations section.

15. 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.

16. 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, 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.

17. References

17.1. Normative References

[I-D.ietf-ippm-metric-registry]


17.2. Informative References


Authors' Addresses

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

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

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.
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 29, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ........................................... 3
2. Conventions ........................................... 3
3. Scope, Applicability, and Assumptions .................. 4
4. IOAM Data Types and Formats ............................ 5
   4.1. IOAM Tracing Options ............................... 6
      4.1.1. Pre-allocated and Incremental Trace Options .... 8
      4.1.2. IOAM node data fields and associated formats.... 12
      4.1.3. Examples of IOAM node data .................... 17
      4.2. IOAM Proof of Transit Option ..................... 19
      4.2.1. IOAM Proof of Transit Type 0 .................. 20
      4.3. IOAM Edge-to-Edge Option ........................ 21
6. IOAM Data Export ........................................ 27
7. IANA Considerations .................................... 28
   7.1. Creation of a new In-Situ OAM Protocol Parameters
       Registry (IOAM) Protocol Parameters IANA registry .... 28
   7.2. IOAM Type Registry .................................. 28
   7.3. IOAM Trace Type Registry ............................ 29
   7.4. IOAM Trace Flags Registry ........................... 29
   7.5. IOAM POT Type Registry .............................. 29
   7.6. IOAM POT Flags Registry ............................. 29
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
3. Scope, Applicability, and Assumptions

IOAM deployment assumes a set of constraints, requirements, and guiding principles which are described in this section.

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, i.e. 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 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:
o 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.

o Identification of the interface that a packet was received on, i.e. ingress interface.

o Identification of the interface that a packet was sent out on, i.e. egress interface.

o 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.

o 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.

o 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.

4.1.1. 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:

```
0                   1                   2                   3
+---+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        IOAM-Trace-Type        | NodeLen | Flags | RemainingLen |
+---+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The trace option data MUST be 4-octet aligned:

```
+----------------------------------------------------------<
| node data list [0] |
+----------------------------------------------------------
| node data list [1] |
| ...               |
| node data list [n-1] |
+----------------------------------------------------------
| node data list [n] |
```

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 4.1.2. 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 app_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 app_data wide in the node data.

Bit 11 When set indicates presence of the Checksum Complement node data.

Bit 12-15 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.

Section 4.1.2 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.
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.

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.

4.1.2.  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:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Hop_Lim | node_id |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```
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  | egress_if_id     |
+----------------+------------------+
```

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.

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|O|                     transit delay                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

app_data: 4-octet placeholder which can be used by the node to add application specific data. App_data represents a "free-format" 4-octet bit field with its semantics defined by a specific deployment.

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       app_data                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

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).

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       queue depth                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

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 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. The 24-bit "Schema Id" field in the field 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:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                       ingress_if_id                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                       egress_if_id                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

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.

app_data wide: 8-octet placeholder which can be used by the node to add application specific data. App data represents a "free-format" 8-octet bit field with its semantics defined by a specific deployment.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                       app data                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                       app data (contd)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
```

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].

4.1.3. 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:

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
| Hop_Lim     |              node_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                  ingress_if_id             |         egress_if_id          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                      timestamp subseconds                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                            app_data                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-

0xC000: IOAM-Trace-Type is 0xC000 then the format is:
0x9000: IOAM-Trace-Type is 0x9000 then the format is:

```
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
```

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
| node_id |
+----------------------------------------------------------+
```

0x8400: IOAM-Trace-Type is 0x8400 then the format is:

```
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
```

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
| node_id |
+----------------------------------------------------------+
```

0x9400: IOAM-Trace-Type is 0x9400 then the format is:

```
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
```

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
| node_id |
+----------------------------------------------------------+
```

0x3180: IOAM-Trace-Type is 0x3180 then the format is:

```
```

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+----------------------------------------------------------+
| Hop_Lim | node_id |
+----------------------------------------------------------+
| node_id |
+----------------------------------------------------------+
```

```
0x3180: IOAM-Trace-Type is 0x3180 then the format is:

```
```
4.2. 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 POT Type | IOAM POT flags | Reserved |
+---------------------------------------------+
```

IOAM proof of transit option data MUST be 4-octet aligned:

```
+---------------------------------------------+
| POT Option data field determined by IOAM-POT-Type |
+---------------------------------------------+
```

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.

- Reserved: 16-bit Reserved bits are present for future use. The reserved bits 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.

4.2.1. IOAM Proof of Transit Type 0
### IOAM proof of transit option of IOAM POT Type 0:

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IOAM POT Type=0</td>
<td>P</td>
<td>R R R R R R R</td>
</tr>
<tr>
<td>+---------------------------------------------</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random(contd)</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cumulative</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cumulative (contd)</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------</td>
<td>---</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**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.

**Reserved:** 16-bit Reserved bits are present for future use. The reserved bits 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".

4.3. 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:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         IOAM-E2E-Type         |             Reserved          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IOAM edge-to-edge option data MUST be 4-octet aligned:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       E2E Option data field determined by IOAM-E2E-Type       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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 \texttt{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 \texttt{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.

Reserved: 16-bits Reserved bits are present for future use. The reserved bits Must be set to zero upon transmission and ignored 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 [IEEE1588] 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.

```
+----------------+----------------+----------------+----------------+
| Seconds        | Fraction        |
+----------------+----------------+----------------+----------------+
| 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 |
+----------------+----------------+----------------+----------------+
```

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.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Seconds                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Microseconds                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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

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 4.1. The meaning of Bit 0 - 11 for trace type are defined in this document in Paragraph 1 of (Section 4.1.1). The meaning for Bit 12 - 15 are available for assignment via RFC Required process as per [RFC8126].

7.4. IOAM Trace Flags Registry

This registry defines code point for each bit in the 4 bit flags for Pre-allocated trace option and Incremental trace option defined in Section 4.1. The meaning of Bit 0 - 1 for trace flags are defined in this document in Paragraph 5 of Section 4.1.1. The meaning for Bit 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 4.2. The code point value 0 is defined in this document, 1 - 255 are available for assignment via RFC Required process as per [RFC8126].

7.6. IOAM POT Flags Registry

This registry defines code point for each bit in the 8 bit flags for IOAM POT option defined in Section 4.2. The meaning of Bit 0 for IOAM POT flags is defined in this document in Section 4.2. The meaning for Bit 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 4.3. The meaning of Bit 0 - 3 are defined in this document. The meaning of Bit 4 - 15 are available for assignments via RFC Required process as per [RFC8126].

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 4.2) 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, 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


10.2. Informative References

[I-D.ietf-ntp-packet-timestamps]  

[I-D.ietf-nvo3-geneve]  

[I-D.ietf-nvo3-vxlan-gpe]  

[I-D.ietf-sfc-nsh]  

[I-D.kitamura-ipv6-record-route]  

[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]  

[RFC7384]  
Internet-Draft               In-situ OAM Data Fields               June 2018

            Chaining (SFC) Architecture", RFC 7665,
            DOI 10.17487/RFC7665, October 2015, <https://www.rfc-
            editor.org/info/rfc7665>.

            Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799,

[ RFC 7820 ] Mizrahi, T., "UDP Checksum Complement in the One-Way
            Active Measurement Protocol (OWAMP) and Two-Way Active
            Measurement Protocol (TWAMP)", RFC 7820,
            editor.org/info/rfc7820>.

[ RFC 7821 ] Mizrahi, T., "UDP Checksum Complement in the Network Time
            Protocol (NTP)", RFC 7821, DOI 10.17487/RFC7821, March

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
Marvell
6 Hamada St.
Yokneam 2066721
Israel

Email: talmi@marvell.com

David Mozes

Email: mosesster@gmail.com

Petr Lapukhov
Facebook
1 Hacker Way
Menlo Park, CA 94025
US

Email: petr@fb.com
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 January 1, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document.
Table of Contents

1. Introduction ........................................... 3
2. Terminology ............................................. 4
3. Scope .................................................... 6
4. Motivation for a Performance Metrics Registry ............ 7
   4.1. Interoperability .................................... 7
   4.2. Single point of reference for Performance Metrics .... 8
   4.3. Side benefits ....................................... 8
5. Criteria for Performance Metrics Registration ........... 9
6. Performance Metric Registry: Prior attempt ............... 9
   6.1. Why this Attempt Will Succeed ....................... 10
7. Definition of the Performance Metric Registry ........... 11
   7.1. Summary Category ................................... 12
       7.1.1. Identifier ...................................... 12
       7.1.2. Name ........................................... 13
       7.1.3. URIs ........................................... 16
       7.1.4. Description ..................................... 17
       7.1.5. Reference ....................................... 17
       7.1.6. Change Controller ............................... 17
       7.1.7. Version (of Registry Format) .................... 17
   7.2. Metric Definition Category .......................... 17
       7.2.1. Reference Definition ............................ 17
       7.2.2. Fixed Parameters ............................... 18
   7.3. Method of Measurement Category ....................... 18
       7.3.1. Reference Method ................................ 18
       7.3.2. Packet Stream Generation ....................... 19
       7.3.3. Traffic Filter .................................. 19
       7.3.4. Sampling Distribution ......................... 20
       7.3.5. Run-time Parameters ............................ 20
       7.3.6. Role ........................................... 21
   7.4. Output Category ..................................... 21
       7.4.1. Type ............................................ 22
       7.4.2. Reference Definition ............................ 22
       7.4.3. Metric Units ................................... 22
       7.4.4. Calibration ..................................... 22
   7.5. Administrative information .......................... 23
       7.5.1. Status .......................................... 23
       7.5.2. Requester ....................................... 23
       7.5.3. Revision ........................................ 23
       7.5.4. Revision Date ................................... 23
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.

Bagnulo, et al. Expires January 1, 2019
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 [RFC5226] 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 [RFC5226] 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 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

<table>
<thead>
<tr>
<th>Category</th>
<th>Column</th>
<th>Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Identifier</td>
<td>Name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metric Definition</th>
<th>Reference Definition</th>
<th>Fixed Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method of Measurement</td>
<td>Reference</td>
<td>Method</td>
</tr>
<tr>
<td>Output</td>
<td>Type</td>
<td>Reference</td>
</tr>
<tr>
<td>Administrative Information</td>
<td>Status</td>
<td>Request</td>
</tr>
</tbody>
</table>

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 (observations on one stream that combine both active and passive methods)

HybridType2 (observations 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-formatted 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 revisions to the registry 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 unambiguous 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 whether 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 whether 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 accommodated, 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 result of a calibration 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 in accordance with Section 7 are 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 WG mailing list.

8.1. Adding new Performance Metrics to the Performance Metrics Registry

Requests to change 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 RFC5226 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 RFC5226.

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 Registry 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 is held 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 is 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 Revising Registered Performance Metrics; or"

2. "the deprecation harmonizes with an external reference that was itself deprecated through that reference’s accepted deprecation method; or"

A request for deprecation is sent to IANA, which passes it to the Performance Metric Expert 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 ID 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 doesn’t introduce any new security considerations for the Internet. However, the definition of Performance Metrics may introduce some security concerns, and should be reviewed with security in mind.

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 [RFC5226], 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:
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 [RFC5226], 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


12.2. Informative References


[I-D.ietf-ippm-initial-registry]


Internet-Draft Registry for Performance Metrics June 2018


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
Astral 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
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 September 21, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Port Num</th>
<th>Transp. Protocol</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>owamp-control</td>
<td>861</td>
<td>tcp</td>
<td>OWAMP-Control</td>
<td>[RFC4656]</td>
</tr>
<tr>
<td>owamp-test</td>
<td>861</td>
<td>udp</td>
<td>OWAMP-Test</td>
<td>[RFCXXXX]</td>
</tr>
<tr>
<td>twamp-control</td>
<td>862</td>
<td>tcp</td>
<td>TWAMP-Control</td>
<td>[RFC5357]</td>
</tr>
<tr>
<td>twamp-test</td>
<td>862</td>
<td>udp</td>
<td>TWAMP-Test Receiver</td>
<td>[RFCXXXX]</td>
</tr>
</tbody>
</table>

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


10.2. Informative References


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

Morton & Mirsky Expires September 21, 2018 [Page 8]
Greg Mirsky (editor)
ZTE Corp.

Email: gregimirsky@gmail.com
Advanced Unidirectional Route Assessment (AURA)

draft-ietf-ippm-route-02

Abstract

This memo introduces an advanced unidirectional route assessment (AURA) 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", "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 January 3, 2019.
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. All route measurements are limited by the willingness of hosts along the path to be discovered, to cooperate with the methods used, or to recognize that the measurement operation is taking place (such as when tunnels are present).

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 two hosts. Although primarily intended for hosts communicating on the Internet with IP, the definitions and metrics are constructed to be applicable to other network domains, if desired. The methods of measurement to assess the path may not be able to discover all hosts comprising the path, but such omissions are often deterministic and explainable sources of error.

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 delays experienced by 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:

Host Identity  The unique address for hosts communicating within the network domain. For hosts communicating on the Internet with IP, it is 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 identify unique hosts.

Discoverable Host  Hosts that convey their Host Identity according to the requirements of their network domain, such as when error conditions are detected by that host. For hosts communicating with IP packets, 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 address of a host (such as the globally routable IP address).
- **Dst**, the address of a host (such as the globally routable IP address).
- **i**, the limit on the number of Hops a specific packet may visit as it traverses from the host at Src to the host at Dst (such as the TTL or Hop Limit).
- **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), such that the distribution of round-trip 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) result in identical
treatment in a multi-path decision (such as the decision 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 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 count of hops needed by the packet on Member Route j to reach Dst. Member Routes must be unique. The uniqueness property 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 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 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 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 network address, regardless of the interface a packet arrives on, then multiple parallel links cannot be detected in that network domain.
- 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. This applies to other network domains with layers below them, as well.

This paragraph on Temporal Composition moved to support a more complete section on Methodology (section 4).

When a route assessment employs IP packets (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).

The Temporal Measurement and Route Class C (unrelated to address classes of the past) is now partly addressed in Section 4.

3.6. Reporting the Metric

now partly addressed, based on feedback at IETF-101:

An Information Model and an XML Data Model for Storing Traceroute Measurements is available in [RFC5388]. The measured information at each hop includes four pieces of information: a one-dimensional hop
The description of Hop information that may be collected according to this memo covers more dimensions, as defined in Section 3.3 above. For example, the Hop index is two-dimensional to capture the complexity of a Route Ensemble, and it contains corresponding host identities at a minimum. The models need to be expanded to include these features, as well as Arrival Interface ID, Departure Interface ID, and Arrival Timestamp, when available.

@@@ can we leave updates to RFC 5388 for further work? Or, do we need to take-on this topic in an Appendix here?

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 (done for Active).

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 in the original traceroute), UDP or TCP. The later are used when a particular characteristic needs to be to verified, such as filtering or traffic shaping on specific ports (i.e., services). [SCAMPER] supports IPv6 traceroute measurements, keeping the FlowLable constant in all packets.
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 keeping the type, code, and ICMP checksum constant; which occupy the corresponding positions 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 and 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.

4.1.1. Temporal Composition for Route Metrics

The Active Route Assessment Methods described above have the ability to discover portions of a path where ECMP load balancing is present, observed as two or more unique Member Routes having one or more distinct Hops which are part of the Route Ensemble. Likewise, attempts to deliberately vary the flow characteristics to discover all Member Routes will reveal portions of the path which are flow-invariant.

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). There is 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.

For Route Metric measurements, the value of Temporal Composition is to reduce the measurement iterations required with repeated measurements. Reduced iterations are possible by inferring that current measurements using fixed and previously measured flow characteristics:

will have many common hops with previous measurements.
will have relatively time-stable results at the ingress and egress portions of the path when measured from user locations, as opposed to measurements of backbone networks and across inter-domain gateways.

may have greater potential for time-variation in path portions where ECMP load balancing is observed (because increasing or decreasing the pool of links changes the hash calculations).

Optionally, measurement systems may take advantage of the inferences above when seeking to reduce measurement iterations, after exhaustive measurements indicate that the time-stable properties are present. Repetitive Active Route measurement systems:

1. SHOULD occasionally check path portions which have exhibited stable results over time, particularly ingress and egress portions of the path.
2. SHOULD continue testing portions of the path that have previously exhibited ECMP load balancing.
3. SHALL trigger re-assessment of the complete path and Route Ensemble, if any change in hops is observed for a specific (and previously tested) flow.

Comments on this new material are very welcome!

4.1.2. Routing Class C Identification

There is an opportunity to apply the [RFC2330] 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 (unrelated to address classes of the past) on a path potentially reduces the number of flows required for a given method to assess a Route Ensemble over time.

First, recognize that each Member Route of a Route Ensemble will have a corresponding Routing Class C. Class C can be discovered by testing with multiple flows, all of which traverse the unique set of hops that comprise a specific Member Route.

Second, recognize that the different Routing Classes depend primarily on the hash functions used at each instance of ECMP load balancing on the path.

Third, recognize the synergy with Temporal Composition methods (described above) where evaluation intends to discover time-stable
portions of each Member Route so that more emphasis can be placed on ECMP portions that also determine Class C.

The methods to assess the various Routing Class C characteristics benefit from the following measurement capabilities:

- flows designed to determine which n-tuple header fields are considered by a given hash function and ECMP hop on the path, and which are not. This operation immediately narrows the search space, where possible, and partially defines a Routing Class C.

- a priori knowledge of the possible types of hash functions in use also helps to design the flows for testing (major router vendors publish information about these hash functions, examples are here https://www.researchgate.net/publication/281571413_COMPARISON_OF_HASH_STRATEGIES_FOR_FLOW-BASED_LOAD_BALANCING).

- ability to direct the emphasis of current measurements on ECMP portions of the path, based on recent past measurement results (the Class C of some portions of the path is essentially "all packets").

@@@ Comments on this new material are very welcome! Especially suggestions for tools that might lend themselves to support these measurements.

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 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:
input: W (window time of the measurement)
  i_t (time between two measurements)
  E (True: exhaustive, False: a single path)
  Dst (destination IP address)
output: Qs (quartiles for every hop and alt in the path(s) to Dst)

T <$> start_timer(W)
while T is not finished do:
  start_timer(i_t)
  RTD(hop,alt) = advanced-traceroute(Dst,E)
  for each hop and alt in RTD do:
    Qs[Dst,hop,alt] <$> ComputeQs(RTD(hop,alt))
  done
wait until i_t timer is expired
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. We thank the good folks at IANA for having checked this section anyway.

11. Acknowledgements

The original 3 authors acknowledge Ruediger Geib, for his penetrating comments on the initial draft, and his initial text for the Appendix on MPLS. 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. Appendix I MPLS Methods for Route Assessment

A host assessing an MPLS path must be part of the MPLS domain where the path is implemented. When this condition is met, RFC 8029 provides a powerful set of mechanisms to detect "correct operation of the data plane, as well as a mechanism to verify the data plane against the control plane" [RFC8029].

MPLS routing is based on the presence of a Forwarding Equivalence Class (FEC) Stack in all visited hosts. Selecting one of several Equal Cost Multi Path (ECMP) is however based on information hidden deeper in the stack. Early deployments may support a so called "Entropy label" for this purpose. State of the art deployments base their choice of an ECMP member based on the IP addresses. Both methods allow load sharing information to be decoupled from routing information. Thus, an MPLS traceroute is able to check how packets with a contiguous number of ECMP relevant addresses (and the same destination) are routed by a particular router. The minimum number of MPLS paths traceable at a router should be 32. Implementations supporting more paths are available.

The MPLS echo request and reply messages offering this feature must support the Downstream Detailed Mapping TLV (was Downstream Mapping initially, but the latter has been deprecated). The MPLS echo
response includes the incoming interface where a router received the MPLS Echo request. The MPLS Echo reply further informs which of the addresses relevant for the load sharing decision results in a particular next hop interface and contains the next hop’s interface address (if available). This ensures that the next hop will receive a properly coded MPLS Echo request in the next step route of assessment.

RFC to be 8403 (draft-ietf-spring-oam-usecase-10) explains how a central Path Monitoring System could be used to detect arbitrary MPLS paths between any routers within a single MPLS domain. The combination of MPLS forwarding, Segment Routing and MPLS traceroute offers a simple architecture and a powerful mechanism to detect and validate (segment routed) MPLS paths.

13. References

13.1. Normative References

[I-D.ietf-ippm-ioam-data]


13.2. Informative References


Internet-Draft           AURA Metrics & Methods                July 2018


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: acm@research.att.com

Joachim Fabini
TU Wien
Gussmannstrasse 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/
Abstract

This document describes a Simple 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 September 2, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1.  Introduction ...................................................... 2
2.  Conventions used in this document .............................. 3
   2.1.  Terminology .................................................. 3
   2.2.  Requirements Language .............................. 3
3.  Softwarization of Performance Measurement .................. 3
4.  Theory of Operation .............................................. 4
   4.1.  Session-Sender Behavior and Packet Format .......... 4
       4.1.1.  Session-Sender Packet Format in Unauthenticated Mode 4
       4.1.2.  Session-Sender Packet Format in Authenticated and Encrypted Modes ......................... 7
   4.2.  Session-Reflector Behavior and Packet Format ........ 8
       4.2.1.  Session-Reflector Packet Format in Unauthenticated Mode ......................... 9
       4.2.2.  Session-Reflector Packet Format in Authenticated and Encrypted Modes ......................... 10
   4.3.  Interoperability with TWAMP Light ....................... 12
5.  IANA Considerations .............................................. 12
6.  Security Considerations ......................................... 12
7.  Acknowledgments ................................................. 12
8.  References ....................................................... 13
   8.1.  Normative References ...................................... 13
   8.2.  Informative References ..................................... 14
Authors’ Addresses .................................................. 14

1.  Introduction

Development and deployment of Two-Way Active Measurement Protocol (TWAMP) [RFC5357] and its extensions, e.g. [RFC6038] that defined features such as Reflect Octets and Symmetrical Size for TWAMP, provided invaluable experience. Several independent implementations exist, have been deployed and provide important operational performance measurements. At the same time there has been noticeable interest in using a simpler mechanism for active performance monitoring that can provide deterministic behaviour and inherit separation of control (vendor-specific configuration or orchestration) and test functions. One of such is Performance Measurement from IP Edge to Customer Equipment using TWAMP Light from Broadband Forum ([BBF.TR-390]). This document defines active performance measurement test protocol, Simple Two-way Active Measurement Protocol (STAMP), that enables measurement of both one-way and round-trip performance metrics like delay, delay variation and packet loss.
2. Conventions used in this document

2.1. Terminology

STAMP - Simple Two-way Active Measurement Protocol
NTP - Network Time Protocol
PTP - Precision Time Protocol
HMAC - Hashed Message Authentication Code
OWAMP - One-Way Active Measurement Protocol
TWAMP - Two-Way Active Measurement 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

Figure 1 presents Simple Two-way Active Measurement Protocol (STAMP) Session-Sender and Session-Reflector with a measurement session. The configuration and management of the STAMP Session-Sender, Session-Reflector and management of the STAMP sessions can be achieved through various means. Command Line Interface, OSS/BSS using SNMP or SDN using Netconf/YANG are but a few examples.

Figure 1: STAMP Reference Model
4. Theory of Operation

STAMP Session-Sender transmits test packets toward STAMP Session-Reflector. STAMP Session-Reflector receives Session-Sender’s packet and acts according to the configuration and optional control information communicated in the Session-Sender’s test packet. STAMP defines two different test packet formats, one for packets transmitted by the STAMP-Session-Sender and one for packets transmitted by the STAMP-Session-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 Session-Reflector equals to the size of the packet received by the Session-Reflector.

4.1. Session-Sender Behavior and Packet Format

4.1.1. Session-Sender Packet Format in Unauthenticated Mode

Because STAMP supports symmetrical test packets, STAMP Session-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:
Internet-Draft STAMP March 2018

Figure 2: STAMP Session-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.
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.

The STAMP Session-Sender and Session-Reflector MAY use, not use, or set value of the Z field in accordance with the timestamp format in use. This optional field is to enhance operations but local configuration or defaults could be used in its place.

- Must-be-Zero (MBZ) field in the session-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 Session-Reflector is expected to copy back to the Session-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 Session-Sender packet MAY include Type-Length-Value encodings that immediately follow the Comp. MBZ field.
o Type field is two octets long. The value of the Type field is the codepoint allocated by IANA Section 5 that identifies data in the Value field.

o Length is two octets long field and its value is the length of the Value field in octets.

o Value field contains the application specific information. The length of the Value field MUST be four octets aligned.

4.1.2. Session-Sender Packet Format in Authenticated and Encrypted Modes

For authenticated and encrypted modes:

```
 0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Sequence Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      MBZ (12 octets)                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Timestamp                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Error Estimate         |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                          MBZ (70 octets)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Type              |           Length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                Value                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                Comp.MBZ                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       HMAC (16 octets)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: STAMP Session-Sender test packet format in authenticated or encrypted modes
The field definitions are the same as the unauthenticated mode, listed in Section 4.1.1. In addition, Comp.MBZ field is variable length field to align the packet on 16 octets boundary. Also, the packet includes a key-hashed message authentication code (HMAC) ([RFC2104]) hash at the end of the PDU.

The STAMP Session-Sender-packet format (Figure 4) is the same in authenticated and encrypted modes. The encryption and authentication operations are, however, different and protect the data as following:

in authenticated mode the Sequence Number is protected while the Timestamp and the Error Estimate are sent in clear text;

in encrypted mode all fields, including the timestamp and Error Estimate, are protected to provide maximum data confidentiality and integrity protection.

Sending the Timestamp in clear text in authenticated mode allows more consistent reading of time by a Session-Sender on the transmission of the test packet. Reading of the time in encrypted mode must be followed by its encryption which introduces variable delay thus affecting calculated timing metrics.

4.2. Session-Reflector Behavior and Packet Format

The Session-Reflector receives the STAMP test packet, verifies it, prepares and transmits the reflected test packet.

Two modes of STAMP Session-Reflector characterize expected behavior and, consequently, performance metrics that can be measured:

- Stateless - STAMP Session-Reflector does not maintain test state and will reflect back the received sequence number without modification. As a result, only round-trip packet loss can be calculated while the reflector is operating in stateless mode.

- Stateful - STAMP Session-Reflector maintains test state determining forward loss, gaps recognized in the received sequence number. This means both near-end (forward) and far-end (backward) packet loss can be computed. This implies that the STAMP Session-Reflector MUST keep a state for each accepted STAMP-test session, uniquely identifying STAMP-test packets to one such session instance, and enabling adding a sequence number in the test reply that is individually incremented on a per-session basis.
4.2.1. Session-Reflector Packet Format in Unauthenticated Mode

For unauthenticated mode:

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------------------+
|              | Sequence Number            |
|              | Timestamp                  |
+---------------+---------------------------+
|                | Error Estimate             |
|                | MBZ                        |
+---------------+---------------------------+
|                | Receive Timestamp          |
+---------------+---------------------------+
|                | Session-Sender Sequence Number |
|                | Session-Sender Timestamp   |
|                | Session-Sender Error Estimate | MBZ |
|                | Ses-Sender TTL             |
|                | +                          |
|                | Packet Padding (reflected) |
|                | +                          |
|                | Comp.MBZ                   |
|                | Type                      |
|                | Length                    |
|                | Value                     |
+---------------+---------------------------+
```

Figure 5: STAMP Session-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 Session-Reflector:
  * in the stateless mode the Session-Reflector copies the value from the received STAMP test packet’s Sequence Number field;
* in the stateful mode the Session-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.

- Session-Sender Sequence Number, Session-Sender Timestamp, and
  Session-Sender Error Estimate are copies of the corresponding
  fields in the STAMP test packet send by the Session-Sender.

- Session-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 Session-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. Session-Reflector Packet Format in Authenticated and Encrypted
Modes

For authenticated and encrypted modes:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          MBZ (12 octets)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Timestamp                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Error Estimate                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

### Figure 6: STAMP Session-Reflector test packet format in authenticated or encrypted modes

The field definitions are the same as the unauthenticated mode, listed in Section 4.2.1, and includes a key-hashed message authentication code (HMAC) ([RFC2104]) hash at the end of the PDU.
4.3. Interoperability with TWAMP Light

One of important requirements to STAMP is ability to interwork with TWAMP Light device. There are two possible combinations for such use case:

- STAMP Session-Sender with TWAMP Light Session-Reflector;
- TWAMP Light Session-Sender with STAMP Session-Reflector.

In the former case, Session-Sender MAY not be aware that its Session-Reflector does not support STAMP. For example, TWAMP Light Session-Reflector may not support use of UDP port 862 as defined in [I-D.ietf-ippm-port-twamp-test]. But because STAMP Session-Sender MUST be able to send test packets to destination UDP port number from the Dynamic and/or Private Ports range 49152-65535, test management system should find port number that both devices can use. And if any of TLV-based STAMP extensions are used, the TWAMP Light Session-Reflector will view them as Packet Padding field. The Session-Sender SHOULD use the default format for its timestamps - NTP. And it MAY use PTPv2 timestamp format.

In the latter scenario, test management system should set STAMP Session-Reflector to use UDP port number from the Dynamic and/or Private Ports range. As for Packet Padding field that the TWAMP Light Session-Sender includes in its transmitted packet, the STAMP Session-Reflector will process it according to [RFC6038] and return reflected packet of the symmetrical size. The Session-Reflector MUST use the default format for its timestamps - NTP.

5. IANA Considerations

This document doesn’t have any IANA action. This section may be removed before the publication.

6. Security Considerations

Use of HMAC in authenticated and encrypted modes may be used to simultaneously verify both the data integrity and the authentication of the STAMP test packets.

7. Acknowledgments

TBD
8. References

8.1. Normative References

[BBF.TR-390]

[I-D.ietf-ippm-port-twamp-test]

[IEEE.1588.2008]


8.2. Informative References


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

Richard Foote
Nokia
Email: footer.foote@nokia.com
Simple Two-way Active Measurement Protocol (STAMP) Data Model

draft-ietf-ippm-stamp-yang-01

Abstract

This document specifies the data model for implementations of Session-Sender and Session-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 September 2, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

The Simple Two-way Active Measurement Protocol (STAMP) [I-D.ietf-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, discussed 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].

This version of the interfaces data model confirms to the Network Management Datastore Architecture (NMDA) defined in [I-D.ietf-netmod-revised-datastores].

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
2. Scope, Model, and Applicability

The scope of this document includes model of the STAMP as defined in [Editor:ref to STAMP draft].

2.1. Data Model Parameters

This section describes all the parameters of 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 Performance Metric Calculation

The data model supports several scenarios for a STAMP Session-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 consecutive 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 Diagrams
module: ietf-stamp

```yaml
+--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 sender-timestamp-format?       timestamp-format
         +--rw security! {stamp-security}?
            +--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?                    union
            +--rw sender-udp-port?              union
            +--rw reflector-ip?                  union
            +--rw reflector-udp-port?           inet:port-number
            +--rw reflector-timestamp-format?   timestamp-format
            +--rw security! {stamp-security}?
               +--rw key-chain?   kc:key-chain-ref
```

Figure 2: STAMP Configuration Tree Diagram
++--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 sender-timestamp-format?      timestamp-format
    ++--ro reflector-timestamp-format?   timestamp-format
    ++--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 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 sender-timestamp-format?     timestamp-format
++--ro reflector-timestamp-format?  timestamp-format
++--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 reflector-timestamp-format?  timestamp-format
++--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

Figure 3: STAMP State Tree Diagram

rpcs:
|----x stamp-sender-start
| |----w input
| | |----w session-id    uint32
|----x stamp-sender-stop
| |----w input
| | |----w session-id    uint32

Figure 4: STAMP RPC Tree Diagram

3.2. YANG Module

<CODE BEGINS> file "ietf-stamp@2018-03-01.yang"

module ietf-stamp {
    yang-version 1.1;

    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;
        reference "RFC 6991";
    }
    import ietf-yang-types {
        prefix yang;
        reference "RFC 6991";
    }

import ietf-key-chain {
  prefix kc;
  reference "RFC 8177";
}

organization
  "IETF IPPM (IP Performance Metrics) Working Group";

contact
  "draft-ietf-ippm-stamp-yang@tools.ietf.org";

description "STAMP Data Model";

revision "2018-03-01" {
  description
    "00 version. Base STAMP specification is covered";
  reference "";
}

  /*
   * Feature definitions.
   */
   /
   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-security {
     description "Secure STAMP 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."

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 timestamp-format {
    type enumeration {
        enum ntp-format {
            description
            "NTP 64 bit format of a timestamp";
        }
        enum ptp-format {
            description
            "PTPv2 truncated format of a timestamp";
        }
    }
    description
    "Timestamp format used by Session-Sender or Session-Reflector.";
}

typedef percentage {
    type decimal64 {

    

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;
"Min of Packets transmitted delay variation";
}
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 consecutively lost packets during the test interval.";
    }
    leaf loss-burst-min {
        type int32;
        description
        "Minimum number of consecutively 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";
}
}

/* 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 Session-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."
}
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;
leaf sender-timestamp-format {
  type timestamp-format;
default ntp-format;
description "Sender Timestamp format";
}
  uses session-security;
  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 Session-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.";
}

leaf sender-ip {
  type union {
    type inet:ip-address;
    type enumeration {
      enum any {
        description
        "Indicates that the Session-Reflector
         accepts STAMP test packets from
         any Session-Sender";
      }
    }
  }

leaf sender-udp-port {
  type union {
    type inet:port-number {
      range "49152..65535";
    }
    type enumeration {
      enum any {
        description
        "Indicates that the Session-Reflector accepts STAMP test packets from any Session-Sender";
      }
    }
  }
}
leaf reflector-udp-port {
    type inet:port-number{
        range "862 | 49152..65535";
    }
    default 862;
    description "Reflector UDP port number";
}

leaf reflector-timestamp-format {
    type timestamp-format;
    default ntp-format;
    description "Reflector Timestamp format";
    uses session-security;
}

/* 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 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";
  }

  uses session-parameters;
  leaf sender-timestamp-format {
    type timestamp-format;
    default ntp-format;
    description "Sender Timestamp format";
  }

  leaf reflector-timestamp-format {
}
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 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 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 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 sender-timestamp-format {
    type timestamp-format;
default ntp-format;
description "Sender Timestamp format";
}
leaf reflector-timestamp-format {
    type timestamp-format;
default ntp-format;
description "Reflector Timestamp format";
}
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 Session-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;
  leaf reflector-timestamp-format {
    type timestamp-format;
    default ntp-format;
    description "Reflector Timestamp format";
  }

  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";
    }
  }
}
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.


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


prefix: stamp

reference: RFC XXXX

5. Security Considerations

The YANG module specified in 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 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 model [RFC6536] 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 data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

TBD

Unauthorized access to any data node of these subtrees can adversely affect the routing subsystem of both the local device and the network. This may lead to corruption of the measurement that may result in false corrective action, e.g. false negative or false positive. That could be, for example, prolonged and undetected deterioration of quality of service or actions to improve the quality unwarranted by the real network conditions.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/ietf-vrrp:stamp

TBD

Unauthorized access to any data node of these subtrees can disclose the operational state information of VRRP on this device.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

TBD
6. Acknowledgements

Authors recognize and appreciate valuable comments provided by Adrian Pan.

7. Normative References

[I-D.ietf-ippm-stamp]

[I-D.ietf-netmod-revised-datastores]


Appendix A. Example of STAMP Session Configuration

Figure 5 shows a configuration example for a STAMP-Sender.
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <stamp xmlns="urn:ietf:params:xml:ns:yang:ietf-stamp">
    <stamp-session-sender>
      <session-enable>enable</session-enable>
      <session-id>10</session-id>
      <test-session-enable>enable</test-session-enable>
      <number-of-packets>forever</number-of-packets>
      <packet-padding-size/> <!-- use default 27 octets -->
      <interval>10</interval> <!-- 10 microseconds -->
      <measurement-interval/> <!-- use default 60 seconds -->
      <!-- use default 0 repetitions, 
      i.e. do not repeat this session -->
      <repeat/>
      <dscp-value/> <!-- use default 0 (CS0) -->
      <!-- use default ‘stateless’ -->
      <test-session-reflector-mode/>
      <sender-ip></sender-ip>
      <sender-udp-port></sender-udp-port>
      <reflector-ip></reflector-ip>
      <reflector-udp-port/> <!-- use default 862 -->
      <sender-timestamp-format/>
      <!-- No authentication or encryption -->
      <first-percentile/> <!-- use default 95 -->
      <second-percentile/> <!-- use default 99 -->
      <third-percentile/> <!-- use default 99.9 -->
    </stamp-session-sender>
  </stamp>
</data>

Figure 5: XML instance of STAMP Session-Sender configuration
Figure 6: XML instance of STAMP Session-Reflector configuration

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
Hybrid Two-Step Performance Measurement Method
draft-mirsky-ippm-hybrid-two-step-01

Abstract

Development of, and advancements in, automation of network operations brought new requirements for measurement methodology. Among them is the ability to collect instant network state as the packet being processed by the networking elements along its path through the domain. This document introduces a new hybrid measurement method, referred to as hybrid two-step, as it separates the act of measuring and/or calculating performance metric from the act of collecting and transporting network state.

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.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Successful resolution of challenges of automated network operation, as part of, for example, overall service orchestration or data center operation, relies on a timely collection of accurate information that reflects the state of network elements on an unprecedented scale. Because performing the analysis and act upon the collected information requires considerable computing and storage resources, the network state information is unlikely to be processed by network elements themselves but will be relayed into the data storage facilities, e.g. data lakes. The process of producing, collecting network state information also referred in this document as network telemetry, and transporting it for post-processing should work equally well with data flows or injected in the network test packets. RFC 7799 [RFC7799] describes a combination of elements of passive and active measurement as a hybrid measurement.

Several technical methods have been proposed to enable collection of network state information instantaneous to the packet processing, among them [P4.INT] and [I-D.ietf-ippm-ioam-data].

This document introduces Hybrid Two-Step (HTS) as a new hybrid measurement method that separates measuring or calculating performance metric from the collecting and transporting this information. The Hybrid Two-Step method extends the two-step mode of

Residence Time Measurement (RTM) defined in [RFC8169] to on-path network state collection and transport.

2. Conventions used in this document

2.1. Terminology

RTM Residence Time Measurement
ECMP Equal Cost Multipath
MTU Maximum Transmission Unit
HTS Hybrid Two-Step

Network telemetry - the process of collecting and reporting of network state

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. Problem Overview

Performance measurements are meant to provide data that characterize conditions experienced by traffic flows in the network and possibly trigger operational changes (e.g. - re-route of flows, or changes in resource allocations). Changes to a network are determined based on the performance metric information available at the time that a change is to be made. The correctness of this determination is based on the quality of the collected metrics data. The quality of collected measurement data is defined is defined by:

- the resolution and accuracy of each measurement;
- predictability of both the time at which each measurement is made and the timeliness of measurement collection data delivery for use.

Consider the case of delay measurement that relies on collecting time of packet arrival at the ingress interface and time of the packet transmission at egress interface. The method may be to record a local clock value on receiving the first octet of an affected message at the device ingress, and again to record the clock value on sending
the first byte of the same message at the device egress. In this ideal case, the difference between the two recorded clock times corresponds to the time that the message spent in traversing the device. In practice, the times actually recorded can differ from the ideal case by any fixed amount and a correction may then be applied to compute the same time difference taking into account the known fixed time associated with the actual measurement. In this way, the resulting time difference reflects any variable delay associated with queuing.

Depending on the implementation, it may be a challenge to compute the difference between message arrival and departure times and - on the fly - add the necessary residence time information to the same message. And that task may become even more challenging if the packet is encrypted. Implementations SHOULD NOT record a message departure time that may be significantly inaccurate in an effort to include a correlated/computed delay value, in the same message, as a result of estimating the departure time while including any variable time component (such as that associated with buffering and queuing of messages). A similar problem may cause a lower quality of, for example, information that characterizes utilization of the egress interface. If unable to obtain the data consistently, without variable delays for additional processing, information may not accurately reflect the state at the egress interface. To mitigate this problem [RFC8169] defined RTM two-step mode.

Another challenge associated with methods that collect network state information into the actual data packet is the risk to exceed the Maximum Transmission Unit (MTU) size, particularly if the packet traverses overlay domains or VPNs. Since the fragmentation is not available at the transport network, operators may have to reduce MTU size advertised to client layer or risk missing network state data for the part, most probably the latter part, of the path.

4. Theory of Operation

The HTS method consists of the two phases:

- performing a measurement or obtaining network state information, one or more than one type, on a node;
- collecting and transporting the measurement.

HTS uses HTS Control message to define types of measurement or network state data collection requested from a node. HTS Control message may be inserted into the data packet, as meta-data or shim, or be transmitted in a specially constructed test packet.
To collect measurement and network state data from the nodes HTS method uses the follow-up packet. The node that creates the HTS Control message also originates the HTS follow-up packet. The follow-up packet contains characteristic information, copied from the data packet, sufficient for participating nodes to associate it with the original packet. The exact composition of the characteristic information is specific for each transport network and its definition is outside the scope of this document. The follow-up packet also uses the same encapsulation as the data packet. If not payload but only network information used to load-balance flows in equal cost multipath (ECMP), use of the network encapsulation identical to the data packet should guarantee that the follow-up packet remains in-band, i.e. traverses the same set of network elements, with the original data packet. Only one outstanding follow-up packet may be on the node for the given path. That means that if the node receives HTS Control message for the flow on which it still waits for the follow-up packet to the previous HTS Control message, the node will originate the follow-up packet to transport the former set of the network state data and transmit it before it transmits the follow-up packet with the latest set of network state information.

5. IANA Considerations

This document doesn’t have any IANA requirements. The section may be deleted before the publication.

6. Security Considerations

Nodes that practice HTS method are presumed to share a trust model that depends on the existence of a trusted relationship among nodes. This is necessary as these nodes are expected to correctly modify the specific content of the data in the follow-up packet, and the degree to which HTS measurement is useful for network operation depends on this ability. In practice, this means that those portions of messages that contain the network state data cannot be covered by either confidentiality or integrity protection. Though there are methods that make it possible in theory to provide either or both such protections and still allow for intermediate nodes to make detectable but authenticated modifications, such methods do not seem practical at present, particularly for protocols that used to measure latency and/or jitter.

The ability to potentially authenticate and/or encrypt the network state data for scenarios both with and without the participation of intermediate nodes that participate in HTS measurement is left for further study.
While it is possible for a supposed compromised node to intercept and modify the network state information in the follow-up packet, this is an issue that exists for nodes in general – for any and all data that may be carried over the particular networking technology – and is therefore the basis for an additional presumed trust model associated with an existing network.

7. Acknowledgments

TBD

8. References

8.1. Normative References


8.2. Informative References


Authors’ Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Wang Lingqiang
ZTE Corporation
No 19 ,East Huayuan Road
Beijing  100191
P.R.China

Phone: +86 10 82963945
Email: wang.lingqiang@zte.com.cn

Guo Zhui
ZTE Corporation
No 19 ,East Huayuan Road
Beijing  100191
P.R.China

Phone: +86 10 82963945
Email: guo.zhui@zte.com.cn
Abstract

This document describes optional extensions to Simple Two-way Active Measurement Protocol (STAMP) which enable measurement performance metrics in addition to ones enabled by the STAMP base 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 14, 2018.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Simple Two-way Active Measurement Protocol (STAMP) [I-D.ietf-ippm-stamp] supports the use of optional extensions that use Type-Length-Value (TLV) encoding. Such extensions are to enhance the STAMP base functions, such as measurement of one-way and round-trip delay, latency, packet loss, as well as ability to detect packet duplication and out-of-order delivery of the test packets. This specification provides definitions of optional STAMP extensions, their formats, and theory of operation.

2. Conventions used in this document

2.1. Terminology

STAMP - Simple Two-way Active Measurement Protocol

DSCP - Differentiated Services Code Point

ECN - Explicit Congestion Notification
NTP - Network Time Protocol

PTP - Precision Time Protocol

HMAC Hashed Message Authentication Code

TLV Type-Length-Value

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. Theory of Operation

STAMP Session-Sender transmits test packets to STAMP Session-Reflector. STAMP Session-Reflector receives Session-Sender’s packet and acts according to the configuration and optional control information communicated in the Session-Sender’s test packet. STAMP defines two different test packet formats, one for packets transmitted by the STAMP-Session-Sender and one for packets transmitted by the STAMP-Session-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 Session-Reflector equals to the size of the packet received by the Session-Reflector.

4. TLV Extensions to STAMP

TBA

4.1. Extra Padding TLV

TBA
Figure 1: Extra Padding TLV

where fields are defined as the following:

- Extra Padding Type - TBA1 allocated by IANA Section 5.1
- Length - 2 octets long field equals length on the Extra Padding field in octets.
- Extra Padding - a pseudo-random sequence of numbers. The field MAY be filled with all zeroes.

4.2. Location TLV

STAMP session-sender MAY include the Location TLV to request information from the session-reflector. The session-sender SHOULD NOT fill any information fields except for Type and Length. The session-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 session-reflector MUST zero all fields and MUST NOT return any information to the session-sender. The session-reflector MUST ignore all other fields of the received Location TLV.
where fields are defined as the following:

- **Location Type** - TBA2 allocated by IANA Section 5.1

- **Length** - 2 octets long field equals length on the Value field in octets. Length field value MUST be 20 octets for the IPv4 address family. For the 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 session-reflector MUST copy Source MAC of received STAMP packet into this field.

- **Reserved A** - 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 session-reflector STAMP packet.

- **Source IP Address** - IPv4 or IPv6 source address of the received by the session-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.

- **Reserved B** - two octets long field. MUST be zeroed on transmission and ignored on reception.
4.3. Timestamp Information TLV

STAMP session-sender MAY include the Timestamp Information TLV to request information from the session-reflector. The session-sender SHOULD NOT fill any information fields except for Type and Length. The session-reflector MUST validate the Length value of the STAMP test packet. If the value of the Length field is invalid, the session-reflector MUST zero all fields and MUST NOT return any information to the session-sender.

```
+-----------------+-----------------+
| Timestamp Info   | Length          |
| Type             |                 |
+-----------------+-----------------+
| Source           | Method          |
+-----------------+-----------------+
```

Figure 3: Timestamp Information TLV

where fields are defined as the following:

- Timestamp Information Type - TBA3 allocated by IANA Section 5.1
- Length - 2 octets long field, equals 4 octets.
- Synchronization Source - two octets long field that characterizes the source of clock synchronization at the session-reflector. The value is one of Section 5.2.
- Timestamp Method - two octets long field that characterizes timestamping method at the session-reflector. The value is one of Section 5.3. [Ed.note: Should it be split for ingress and egress?]

4.4. Class of Service TLV

The STAMP session-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 5.4, then the STAMP session-reflector MUST copy Differentiated Services Code Point (DSCP) and Explicit Congestion Notification (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 5.4, then the STAMP session-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 a packet.

```
+----------+----------+
| Length   | DSCP     |
+----------+----------+
| Reserved | ECN      |
```

Figure 4: Class of Service TLV

where fields are defined as the following:

- Class of Service Type - TBA4 allocated by IANA Section 5.1
- Length - 2 octets long field, equals 4 octets.
- Op - 4 bits long field with the value set to operation code point:
  * TBA5 - Report CoS values by STAMP Session-Reflector
  * TBA6 - Set and Report CoS values to be used for reflected STAMP test packet
- Reserved - 20 bits long field, must be zeroed in transmission and ignored on receipt.
- DSCP - Differentiated Services Code Point (DSCP).
- ECN - Explicit Congestion Notification.

4.5. Direct Measurement TLV

The Direct Measurement TLV enables collection of "in profile" IP packets that had been transmitted and received by the Session-Sender and Session-Reflector respectfully. The definition of "in-profile packet" is outside the scope of this document.
where fields are defined as the following:

- Direct Measurement Type - TBA5 allocated by IANA Section 5.1
- Length - 2 octets long field equals length on the Value field in octets. Length field value MUST be 12 octets.
- Session-Sender Tx counter (S_TxC) is four octets long field.
- Session-Reflector Rx counter (R_RxC) is four octets long field. MUST be zeroed by the Session-Sender and filled by the Session-Reflector.
- Session-Reflector Tx counter (R_TxC). is four octets long field. MUST be zeroed by the Session-Sender and filled by the Session-Reflector.

5. IANA Considerations

5.1. 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:
This document defines the following new values in STAMP TLV Type registry:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBA1</td>
<td>Extra Padding</td>
<td>This document</td>
</tr>
<tr>
<td>TBA2</td>
<td>Location</td>
<td>This document</td>
</tr>
<tr>
<td>TBA3</td>
<td>Timestamp Information</td>
<td>This document</td>
</tr>
<tr>
<td>TBA4</td>
<td>Class of Service</td>
<td>This document</td>
</tr>
<tr>
<td>TBA5</td>
<td>Direct Measurement</td>
<td>This document</td>
</tr>
</tbody>
</table>

Table 2: STAMP Types

5.2. Synchronization Source Sub-registry

TBD

5.3. Timestamp Method Sub-registry

TBD

5.4. CoS Operation Sub-registry

TBD

6. Security Considerations

Use of HMAC in authenticated and encrypted modes may be used to simultaneously verify both the data integrity and the authentication of the STAMP test packets.
7. Acknowledgments

TBD

8. Normative References

[I-D.ietf-ippm-stamp]


Authors’ Addresses

Greg Mirsky
ZTE Corp.

Email: gregimirsky@gmail.com

Xiao Min
ZTE Corp.

Email: xiao.min2@zte.com.cn
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

Richard Foote
Nokia

Email: footer.foote@nokia.com

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 September 5, 2018.
Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the
document authors. All rights reserved.

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

Table of Contents

1. Introduction ................................................. 3
   1.1. Background ............................................ 3
   1.2. The Scope of This Document .............................. 4
2. Terminology ................................................ 5
   2.1. Requirements Language ................................. 5
   2.2. Abbreviations ......................................... 5
3. Marking Abstractions ................................. 5
4. Double Marking .......................................... 7
5. Single-bit Marking ............................. 8
   5.1. Single Marking Using the First Packet ............. 8
   5.2. Single Marking using the Mean Delay ............... 8
   5.3. Single Marking using a Multiplexed Marking Bit ..... 8
      5.3.1. Overview ........................................... 8
      5.3.2. Timing and Synchronization Aspects ............. 9
   5.4. Pulse Marking .......................................... 11
6. Zero Marking Hashed .......................... 12
   6.1. Hash-based Sampling ................................. 12
      6.1.1. Hashed Pulse Marking ......................... 13
      6.1.2. Hashed Step Marking ........................... 13
7. Single Marking Hashed .......................... 13
8. Summary of Marking Methods .................. 14
9. Alternate Marking using Reserved Values 19
10. IANA Considerations .............................. 20
11. Security Considerations ....................... 20
12. References ................................................. 20
   12.1. Normative References ................................ 20
   12.2. Informative References ............................. 20
Authors’ Addresses ........................................ 21
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.

A: packet with color 0
B: packet with color 1

<table>
<thead>
<tr>
<th>Packets</th>
<th>AAAAAAAAAA BBBBBBBBBB AAAAAAAAAA BBBBBBBBBB AAAAAAAAAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Block 1</td>
</tr>
<tr>
<td>Color</td>
<td>0000000000 1111111111 0000000000 1111111111 0000000000</td>
</tr>
</tbody>
</table>

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 management 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).

P: indicates a packet

<table>
<thead>
<tr>
<th>Packets</th>
<th>PPPPPPPPPP PPPPPPPPPP PPPPPPPPPP PPPPPPPPPP PPPPPPPPPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Marking bit</td>
<td>0000010000 0000010000 0000010000 0000010000 00000 0000</td>
</tr>
</tbody>
</table>

Pulse-based detection

Dropped packet: no detection

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.
P: indicates a packet

<table>
<thead>
<tr>
<th>Packets</th>
<th>PPPPPPPPPP PPPPPPPPPP PPPPPPPPPP PPPPPPPPPP PPPPPPPPPP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Marking bit</td>
<td>0000000000 1111111111 0000000000 10111111111 0000000000</td>
</tr>
<tr>
<td>Step-based detection</td>
<td>0000100000 0000100000 0000100000 0001000000</td>
</tr>
<tr>
<td>out-of-order</td>
<td>^ ^ ^</td>
</tr>
</tbody>
</table>

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.

| A: packet with color 0 |
| B: packet with color 1 |

<table>
<thead>
<tr>
<th>Packets</th>
<th>AAAAAAAAAA BBBBBBBBBB AAAAAAAAAA BBBBBBBBBB AAAAAAAAAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Block 1</td>
<td>Block 2</td>
</tr>
<tr>
<td>Color bit</td>
<td>0000000000 1111111111 0000000000 1111111111 0000000000</td>
</tr>
<tr>
<td>Delay bit</td>
<td>0000100000 0000100000 0001000000 0001000000 0010000000</td>
</tr>
<tr>
<td>Packets marked for timestamping</td>
<td>^ ^ ^ ^</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Packets</th>
<th>AAAAAAAAAA BBBB BBBB AAAAAAAAAA BBBB BBBB AAAAAAAAAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>Block 1</td>
<td>Block 2</td>
</tr>
<tr>
<td>Block 3</td>
<td>Block 4</td>
</tr>
<tr>
<td>Block 5</td>
<td>...</td>
</tr>
<tr>
<td>Color bit</td>
<td>0000000000 1111111111 0000000000 1111111111 0000000000</td>
</tr>
</tbody>
</table>

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

| Time   | Block 1 | Block 2 | Block 3 | Block 4 | Block 5 ...
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>0000000000</td>
<td>1111111111</td>
<td>0000000000</td>
<td>1111111111</td>
<td>0000000000</td>
</tr>
<tr>
<td>Packets marked for timestamping</td>
<td>^</td>
<td>^</td>
<td>^</td>
<td>^</td>
<td></td>
</tr>
<tr>
<td>Muxed bit</td>
<td>0001000000</td>
<td>1110111111</td>
<td>0001000000</td>
<td>1111101111</td>
<td>0001000000</td>
</tr>
</tbody>
</table>

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.

```
+--- Beginning of measurement period

...BBBBBBBBBB | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA | BBBBBBBBBBB...
<--------------------->                      <--------------------->
L/4        L/4               L/2               L/4       L/4

<---------------------><---------------------><--------------------->
Detect color    Detect timestamping    Detect color
change          indication              change

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.

```
+--- Beginning of measurement period

...BBBBBBBBBB | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA | BBBBBBBBBBB...
<--------------------->                      <--------------------->
L/4        L/4               L/2               L/4       L/4

<---------------------><---------------------><--------------------->
Detect color    Detect timestamping    Detect color
change          indication              change

<---------------------><---------------------><--------------------->
permissible    timestamping    interval
```
The guardband $d$ is given by $d = A + D_{\text{max}} - D_{\text{min}}$, where $A$ is the clock accuracy, $D_{\text{max}}$ is an upper bound on the network delay between the MPs, and $D_{\text{min}}$ 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 $\pm L/8$. 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 $\pm L/2$, while the multiplexed marking method requires an accuracy of $\pm L/8$.

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 $\pm L/2$ 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

<table>
<thead>
<tr>
<th>Time</th>
<th>Block 1</th>
<th>Block 2</th>
<th>Block 3</th>
<th>Block 4</th>
<th>Block 5 ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>packets</td>
<td>PPPPPPPPPP</td>
<td>PPPPPPPPPP</td>
<td>PPPPPPPPPP</td>
<td>PPPPPPPPPP</td>
<td>PPPPPPPPPP</td>
</tr>
</tbody>
</table>

| packets marked for DM and LM | v | v | v | v | v |

| Marking bit | 0000100000 0000100000 0000100000 0000010000 0001000000 |

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 \( p_1 \) and \( p_2 \), if \( p_1 \) is sent before \( p_2 \) then \( h(p_1) <= h(p_2) \). 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.
<table>
<thead>
<tr>
<th>Method</th>
<th># of bits</th>
<th># of counters</th>
<th>LM Method</th>
<th>DM Method</th>
<th>Resilience to Reordering</th>
<th>Resilience to Packet drops</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>LM</td>
<td>DM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>2</td>
<td>Step</td>
<td>Step</td>
<td>+</td>
<td>--</td>
</tr>
<tr>
<td>- 1st packet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>2</td>
<td>Step</td>
<td>Mean</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>- mean delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Double marking</td>
<td>2</td>
<td>2</td>
<td>Step</td>
<td>Pulse</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>- pulse based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>2</td>
<td>Step</td>
<td>Pulse</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>multiplexed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse marking</td>
<td>1</td>
<td>1</td>
<td>Pulse</td>
<td>Pulse</td>
<td>--</td>
<td>+</td>
</tr>
<tr>
<td>- pulse based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zero marking</td>
<td>0</td>
<td>(2)</td>
<td>Hashed pulse (step)</td>
<td>Hashed pulse</td>
<td>--</td>
<td>-</td>
</tr>
<tr>
<td>- hashed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>2</td>
<td>Step</td>
<td>Hashed pulse</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>hashed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

+ 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 significantly 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).
### Table 1: Summary of Marking Methods: focus on Delay Measurement

<table>
<thead>
<tr>
<th>Marking Method</th>
<th># of bits</th>
<th>LM on All Packets</th>
<th>DM Resilience to Reordering</th>
<th>DM Resilience to Packet drops</th>
<th>DM Multipoint compatible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single marking</td>
<td>1</td>
<td>Yes</td>
<td>--</td>
<td>--</td>
<td>No</td>
</tr>
<tr>
<td>- 1st packet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>Yes</td>
<td>+</td>
<td>--</td>
<td>Yes</td>
</tr>
<tr>
<td>- mean delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Double marking</td>
<td>2</td>
<td>Yes</td>
<td>+</td>
<td>=</td>
<td>No</td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>Yes</td>
<td>+</td>
<td>=</td>
<td>No</td>
</tr>
<tr>
<td>multiplexed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse marking</td>
<td>1</td>
<td>No</td>
<td>+</td>
<td>=</td>
<td>No</td>
</tr>
<tr>
<td>Zero marking hashed</td>
<td>0</td>
<td>No</td>
<td>+</td>
<td>+</td>
<td>Yes</td>
</tr>
<tr>
<td>Single marking</td>
<td>1</td>
<td>Yes</td>
<td>+</td>
<td>+</td>
<td>Yes</td>
</tr>
<tr>
<td>hashed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

+ 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

<table>
<thead>
<tr>
<th>Time</th>
<th>Block 1</th>
<th>Block 2</th>
<th>Block 3</th>
<th>Block 4</th>
<th>Block 5 ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>0000000000 1111111111 0000000000 1111111111 0000000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Packets</td>
<td>^</td>
<td>^</td>
<td>^</td>
<td>^</td>
<td>^</td>
</tr>
<tr>
<td>marked for</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
<td>v</td>
</tr>
<tr>
<td>timestamping</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Muxed</td>
<td>UUUWUUUUUU WWWWUWWWWW UUUWUUUUUU WWWWUWWWWW UUUWUUUUUU</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>marking</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>values</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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


12.2.  Informative References


[I-D.ietf-mpls-sfl-framework]

[IEEE1588]


Authors’ Addresses

Tal Mizrahi
Marvell
6 Hamada st.
Yokneam
Israel

Email: talmi@marvell.com

Carmi Arad
Marvell
6 Hamada st.
Yokneam
Israel

Email: carmi.arad@gmail.com
Giuseppe Fioccola
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

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
GRE Encapsulation for In-situ OAM Data

draft-weis-ippm-ioam-gre-00

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 outlines how IOAM data fields are encapsulated in GRE.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any
1. Introduction

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 outlines how IOAM data fields are encapsulated in the Generic Routing Encapsulation (GRE) [RFC2784].
2. Conventions

2.1. 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.

2.2. Abbreviations

Abbreviations used in this document:

E2E: Edge-to-Edge
GRE: Generic Routing Encapsulation
IOAM: In-situ Operations, Administration, and Maintenance
OAM: Operations, Administration, and Maintenance
POT: Proof of Transit

3. In-situ OAM Metadata Transport in GRE

GRE encapsulation is defined in [RFC2784]. IOAM encapsulation in GRE follows the GRE header.

IOAM data fields are carried in GRE using a Protocol Type value of TBD_IOAM. An IOAM header is added containing the different IOAM data fields defined in [I-D.ietf-ippm-ioam-data]. In an administrative domain where IOAM is used, insertion of the IOAM protocol header in GRE is enabled at the GRE tunnel endpoints, which also serve as IOAM encapsulating/decapsulating nodes by means of configuration.
The GRE header and fields are defined in [RFC2784]. The GRE Protocol Type value is TBD_IOAM.

The IOAM header is defined as follows.

IOAM Type: 8-bit field defining the IOAM Option type, as defined in Section 7.2 of [I-D.ietf-ippm-ioam-data].

IOAM HDR Len: 8 bits Length field contains the length of the variable IOAM data octets in 4-octet units.

Next Protocol: 16 bits Next Protocol Type field contains the protocol type of the packet following IOAM protocol header. When the most significant octet is 0x00, the Protocol Type is taken to be an IP Protocol Number as defined in [IP-PROT]. Otherwise, the Protocol Type is defined to be an EtherType value from [ETYPES]. An implementation receiving a packet containing a Protocol Type which is not listed in one of those registries SHOULD discard the packet.

IOAM Option and Data Space: IOAM option header and data is present as specified by the IOAM-Type field, and is defined in Section 4 of [I-D.ietf-ippm-ioam-data].

Multiple IOAM options MAY be included within the GRE encapsulation. For example, if a GRE encapsulation contains two IOAM options before a data packet, the Next Protocol field of the first IOAM option will contain the value of TBD_IOAM, while the Next Protocol field of the
second IOAM option will contain the Ethertype or IP protocol Number indicating the type of the data packet.

4. Security Considerations

This document describes the encapsulation of IOAM data fields in GRE. Security considerations of the specific IOAM data fields for each case (i.e., Trace, Proof of Transit, and E2E) are described in defined in [I-D.ietf-ippm-ioam-data].

As this document describes new protocol fields within the existing GRE encapsulation, these are similar to the security considerations of [RFC2784].

IOAM data transported in an OAM E2E header SHOULD be integrity protected (e.g., with IPsec ESP [RFC4303]) to detect changes made by a device between the sending and receiving OAM endpoints.

5. IANA Considerations

A new EtherType value is requested to be added to the [ETYPES] IANA registry. The description should be "In-situ OAM (IOAM)".

6. References

6.1. Normative References


6.2. Informative References


Appendix A. Example GRE-IOAM Payloads

A.1. Example GRE-IOAM Tracing Payloads

TBD

Authors’ Addresses

Brian Weis
Cisco Systems, Inc.
170 W. Tasman Drive
San Jose, California  95134-1706
USA
Phone: +1-408-526-4796
Email: bew@cisco.com

Frank Brockners
Cisco Systems, Inc.
Hansaallee 249, 3rd Floor
DUESSELDORF, NORDRHEIN-WESTFALEN  40549
Germany
Email: fbrockne@cisco.com
Craig Hill
Cisco Systems, Inc.
13600 Dulles Technology Drive
Herndon, Virginia  20171
United States

Email: crhill@cisco.com

Shwetha Bhandari
Cisco Systems, Inc.
Cessna Business Park, Sarjapura Marathalli Outer Ring Road
Bangalore, KARNATAKA 560 087
India

Email: shwethab@cisco.com

Vengada Prasad Govindan
Cisco Systems, Inc.

Email: venggovi@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

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  
Marvell  
6 Hamada St.  
Yokneam  20692  
Israel  
Email: talmi@marvell.com

Aviv Kfir  
Mellanox Technologies, Inc.  
350 Oakmead Parkway, Suite 100  
Sunnyvale, CA  94085  
U.S.A.  
Email: avivk@mellanox.com

Barak Gafni  
Mellanox Technologies, Inc.  
350 Oakmead Parkway, Suite 100  
Sunnyvale, CA  94085  
U.S.A.  
Email: gbarak@mellanox.com

Petr Lapukhov  
Facebook  
1 Hacker Way  
Menlo Park, CA  94025  
US  
Email: petr@fb.com
Mickey Spiegel
Barefoot Networks
2185 Park Boulevard
Palo Alto, CA  94306
US

Email: mspiegel@barefootnetworks.com
A YANG Data Model for In-Situ OAM
draft-zhou-ippm-ioam-yang-02

Abstract

In-situ Operations, Administration, and Maintenance (IOAM) records operational and telemetry information in user packets while the packets traverse a path between two points in the network. This document defines a YANG module for the IOAM function.

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 January 3, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

In-situ Operations, Administration, and Maintenance (IOAM) [I-D.ietf-ippm-ioam-data] records OAM information within user packets while the packets traverse a network. The data types and data formats for IOAM data records have been defined in [I-D.ietf-ippm-ioam-data]. The IOAM data can be embedded in many protocol encapsulations such as Network Services Header, Segment Routing, and IPv6 [I-D.brockners-inband-oam-transport].

This document defines a data model for IOAM capabilities using the YANG data modeling language [RFC7950]. This YANG model supports all the three categories of IOAM data, which are Tracing Option, Proof of Transit Option, and Edge-to-Edge Option.
1.1. Tree Diagrams

The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.
- Abbreviations before data node names: "rw" means configuration (read-write), "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" a container with presence, and "*" denotes a "list" or "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis "(...)" stands for contents of subtrees that are not shown.

2. Design of the IOAM YANG Data Model

2.1. Profiles

The IOAM model is organized as list of profiles as shown in the following figure. Each profile associates with one flow and the corresponding IOAM information.

```yang
module: ietf-ioam
  +-rw ioam
    +-rw ioam-profiles
      +-rw admin-config
        | +-rw enabled?  boolean
        +-rw ioam-profile* [profile-name]
          +-rw profile-name  string
          +-rw filter
            | +-rw filter-type?  ioam-filter-type
            | +-rw acl-name?  -> /acl:acls/acl/name
            +-rw protocol-type?  ioam-protocol-type
          +-rw incremental-tracing-profile {incremental-trace}?
            | ...
          +-rw preallocated-tracing-profile {preallocated-trace}?
            | ...
          +-rw pot-profile {proof-of-transit}?
            | ...
          +-rw e2e-profile {edge-to-edge}?
            | ...
```
The "enabled" is an administrative configuration. When it is set to true, IOAM configuration is enabled for the system. Meanwhile, the IOAM data-plane functionality is enabled.

The "filter" is used to identify a flow, where the IOAM profile can apply. There may be multiple filter types. ACL is the default one.

The IOAM data can be encapsulated into multiple protocols, e.g., IPv6 [RFC8200], Geneve [I-D.ietf-nvo3-geneve], VxLAN-GPE [I-D.ietf-nvo3-vxlan-gpe]. The "protocol-type" is used to indicate where the IOAM is applied. For example, if the "protocol-type" is IPv6, the IOAM ingress node will encapsulate the associated flow with the IPv6-IOAM [I-D.brockners-inband-oam-transport] format.

IOAM data includes three usage options with four encapsulation types, i.e., incremental tracing data, preallocated tracing data, prove of transit data and end to end data. In practice, multiple IOAM data types can be encapsulated into the same IOAM header. The "ioam-profile" contains a set of sub-profiles, each of which relates to one encapsulation type. The configured object may not support all the sub-profiles. The supported sub-profiles are indicated by 4 defined features, i.e., "incremental-trace", "preallocated-trace", "proof-of-transit", "edge-to-edge".

2.2. Preallocated Tracing Profile

The 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. The preallocated tracing option will create pre-allocated space for each node to populate its information. The "preallocated-tracing-profile" contains the detailed information for the preallocated tracing data. The information includes:

- enabled: indicates whether the preallocated tracing profile is enabled.
- node-action: indicates the operation (e.g., encapsulate IOAM header, transit the IOAM data, or decapsulate IOAM header) applied to the dedicated flow.
- trace-type: indicates the per-hop data to be captured by the IOAM enabled nodes and included in the node data list.
- Loopback mode is used to send a copy of a packet back towards the source.
2.3. Incremental Tracing Profile

The incremental tracing option contains a variable node data fields where each node allocates and pushes its node data immediately following the option header. The "incremental-tracing-profile" contains the detailed information for the incremental tracing data. The detailed information is the same as the Preallocated Tracing Profile, but with one more variable, "max-length", which restricts the length of the IOAM header.

2.4. Proof of Transit Profile

The IOAM Proof of Transit data is to support the path or service function chain verification use cases. The "pot-profile" contains the detailed information for the prove of transit data. The detailed information are described in [I-D.brockners-proof-of-transit].

2.5. Edge to Edge Profile

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 "e2e-profile" contains the detailed information for the edge to edge data. The detailed information includes:

- enabled: indicates whether the edge to edge profile is enabled.
- node-action is the same semantic as in Section 2.2.
- e2e-type indicates data to be carried from the ingress IOAM node to the egress IOAM node.

```yang
++--rw e2e-profile {edge-to-edge}?
    ++--rw enabled?    boolean
    ++--rw node-action? ioam-node-action
    ++--rw e2e-type?    ioam-e2e-types
```

3. IOAM YANG Module

<CODE BEGINS> file "ietf-ioam@2018-07-02.yang"
module ietf-ioam {
    yang-version 1.1;
    prefix "ioam";
    import ietf-pot-profile {
        prefix "pot";
    }

    import ietf-access-control-list {
        prefix "acl";
    }

    organization "IETF IPPM (IP Performance Metrics) Working Group";
    contact "WG Web: <http://tools.ietf.org/wg/ippm>
          WG List: <ippm@ietf.org>
          Editor: zhoutianran@huawei.com";
    description "This YANG module specifies a vendor-independent data model for the in Situ OAM (iOAM).";

    revision 2018-07-02 {
        description "Initial revision."
        reference "draft-zhou-ippm-ioam-yang";
    }
}
/*
* FEATURES
*/

feature incremental-trace
{
    description
        "This feature indicated that the incremental tracing mode is supported";
}

feature preallocated-trace
{
    description
        "This feature indicated that the preallocated tracing mode is supported";
}

feature proof-of-transit
{
    description
        "This feature indicated that the proof of transit mode is supported";
}

feature edge-to-edge
{
    description
        "This feature indicated that the edge to edge mode is supported";
}

/*
* IDENTITIES
*/

identity base-filter {
    description
        "Base identity to represent a filter. A filter is used to specify the flow to apply the iOAM profile.";
}

identity acl-filter {
    base base-filter;
    description
        "Apply ACL rule to specify the flow.";
}

identity base-protocol {
    description

"Base identity to represent the carrier protocol. It’s used to indicate what layer and protocol the iOAM data is embedded.");
}

identity ipv6-protocol {
  base base-protocol;
  description
    "The described iOAM data is embedded in ipv6 protocol.";
}

identity base-node-action {
  description
    "Base identity to represent the node actions. It’s used to indicate what action the node will take.";
}

identity encapsulate {
  base base-node-action;
  description
    "indicate the node is to encapsulate the iOAM packet";
}

identity transit {
  base base-node-action;
  description
    "indicate the node is to transit the iOAM packet";
}

identity decapsulate {
  base base-node-action;
  description
    "indicate the node is to decapsulate the iOAM packet";
}

/*
 * TYPE DEFINITIONS
 */

typedef ioam-filter-type {
  type identityref {
    base base-filter;
  }
  description
    "Specifies a known type of filter.";
}

typedef ioam-protocol-type {
  type identityref { 

base base-protocol;
}  

description  
"Specifies a known type of carrier protocol for the iOAM data.";
}

typedef ioam-node-action {  
type identityref {  
  base base-node-action;
}  

description  
"Specifies a known type of node action.";
}

typedef ioam-trace-types {  
type bits {  
  bit ioam-hop-lim-node-id {  
    position 0;  
    description  
      "When set indicates presence of Hop_Lim and node_id in the  
        node data.";
  }  
  bit ioam-if-id {  
    position 1;  
    description  
      "When set indicates presence of ingress_if_id and  
        egress_if_id in the node data.";
  }  
  bit ioam-timestamp-seconds {  
    position 2;  
    description  
      "When set indicates presence of time stamp seconds in the  
        node data.";
  }  
  bit ioam-timestamp-nanoseconds {  
    position 3;  
    description  
      "When set indicates presence of time stamp nanoseconds in  
        the node data.";
  }  
  bit ioam-transit-delay {  
    position 4;  
    description  
      "When set indicates presence of transit delay in the node  
        data.";
  }  
  bit ioam-app-data {  
    position 5;
description
   "When set indicates presence of app_data in the node data."
}
bit ioam-queue-depth {
   position 6;
   description
   "When set indicates presence of queue depth in the node data.";
}
bit ioam-opaque-state-snapshot {
   position 7;
   description
   "When set indicates presence of variable length Opaque State Snapshot field.";
}
bit ioam-hop-lim-node-id-wide {
   position 8;
   description
   "When set indicates presence of Hop_Lim and node_id wide in the node data.";
}
bit ioam-if-id-wide {
   position 9;
   description
   "When set indicates presence of ingress_if_id and egress_if_id wide in the node data.";
}
bit app-data-wide {
   position 10;
   description
   "When set indicates presence of app_data wide in the node data.";
}

description
   "A 16-bit identifier which specifies which data types are used in this node data list."

typedef ioam-pot-types {
   type bits {
      bit ioam-bytes-16 {
         position 0;
         description
         "POT data is a 16 Octet field";
      }
   }
   description
"7-bit identifier of a particular POT variant that dictates the POT data that is included.";
}

typedef ioam-e2e-types {
type bits {
  bit ioam-seq-num {
    position 0;
    description "A 64-bit sequence number added to a specific tube which is used to identify packet loss and reordering for that tube.";
  }
}

description "8-bit identifier of a particular in situ OAM E2E variant.";
}

/*
* GROUP DEFINITIONS
*/

grouping ioam-filter {
description "A grouping for iOAM filter definition";

leaf filter-type {
type ioam-filter-type;
description "filter type";
}

leaf acl-name {
  when ".//filter-type = 'acl-filter'";
type leafref {
    path "/acl:acls/acl:acl/acl:name";
  }
description "Access Control List name.";
}

}

grouping ioam-incremental-tracing-profile {
description "A grouping for incremental tracing profile.";

leaf node-action {
type ioam-node-action;
description "node action";
}
leaf trace-type {
    when "./node-action = 'encapsulate'";
    type ioam-trace-types;
    description
        "The trace type is only defined at the encapsulation node."
    }

leaf enable-loopback-mode {
    when "./node-action = 'encapsulate'";
    type boolean;
    default false;
    description
        "Loopback mode is used to send a copy of a packet back towards
         the source. The loopback mode is only defined at the
         encapsulation node."
    }

leaf max-length {
    when "./node-action = 'encapsulate'";
    type uint32;
    description
        "This field specifies the maximum length of the node data list
         in octets. The max-length is only defined at the
         encapsulation node. And it's only used for the incremental
         tracing mode."
    }
}

grouping ioam-preallocated-tracing-profile {
    description
        "A grouping for incremental tracing profile."
}

leaf node-action {
    type ioam-node-action;
    description "node action"
}

leaf trace-type {
    when "./node-action = 'encapsulate'";
    type ioam-trace-types;
    description
        "The trace type is only defined at the encapsulation node."
}

leaf enable-loopback-mode {
    when "./node-action = 'encapsulate'";
    type boolean;
    default false;
}
description
"Loopback mode is used to send a copy of a packet back towards the source. The loopback mode is only defined at the encapsulation node."
}
}

grouping ioam-e2e-profile {
    description
    "A grouping for tracing profile."

    leaf node-action {
        type ioam-node-action;
        description
        "indicate how the node act for this profile"
    }

    leaf e2e-type {
        when ".../node-action = 'encapsulate'";
        type ioam-e2e-types;
        description
        "The e2e type is only defined at the encapsulation node."
    }
}


grouping ioam-admin-config {
    description
    "IOAM top-level administrative configuration."

    leaf enabled {
        type boolean;
        default false;
        description
        "When true, IOAM configuration is enabled for the system. Meanwhile, the IOAM data-plane functionality is enabled."
    }
}

/*
 * DATA NODES
*/

container ioam {
    description "iOAM top level container"

    container ioam-profiles {
        description
        "Contains a list of IOAM profiles."
    }
}
container admin-config {
  description
  "Contains all the administrative configurations related to
  the IOAM functionalities and all the IOAM profiles.";
  uses ioam-admin-config;
}

list ioam-profile {
  key "profile-name";
  ordered-by user;
  description
  "A list of iOAM profiles that configured on the node.";
  leaf profile-name {
    type string;
    mandatory true;
    description
    "Unique identifier for each iOAM profile";
  }
  container filter {
    uses ioam-filter;
    description
    "The filter which is used to indicate the flow to apply
    iOAM.";
  }
  leaf protocol-type {
    type ioam-protocol-type;
    description
    "This item is used to indicate the carrier protocol where
    the iOAM is applied.";
  }
  container incremental-tracing-profile {
    if-feature incremental-trace;
    description
    "Describe the profile for incremental tracing option";
    leaf enabled {
      type boolean;
      default false;
      description
      "When true, apply incremental tracing option to the
      specified flow identified by the filter.";
    }
  }
}
uses ioam-incremental-tracing-profile;
}

container preallocated-tracing-profile {
  if-feature preallocated-trace;
  description
    "describe the profile for preallocated tracing option";
  leaf enabled {
    type boolean;
    default false;
    description
      "When true, apply preallocated tracing option to the
      specified flow identified by the following filter.";
  }
  uses ioam-preallocated-tracing-profile;
}

container pot-profile {
  if-feature proof-of-transit;
  description
    "describe the profile for pot option";
  leaf enabled {
    type boolean;
    default false;
    description
      "When true, apply Proof of Transit option to the
      specified flow identified by the following filter.";
  }
  leaf active-profile-index {
    type pot:profile-index-range;
    description
      "Proof of transit profile index that is currently
      active. Will be set in the first hop of the path
      or chain. Other nodes will not use this field.";
  }
  uses pot:pot-profile;
}

container e2e-profile {
  if-feature edge-to-edge;
  description
    "describe the profile for e2e option";
4. Security Considerations

The YANG module specified in 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 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 model [RFC6536] 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 data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

- /ioam/ioam-profiles/admin-config

The items in the container above include the top level administrative configurations related to the IOAM functionalities and all the IOAM profiles. Unexpected changes to these items could lead to the IOAM function disruption and/or misbehavior of all the IOAM profiles.

- /ioam/ioam-profiles/ioam-profile
The entries in the list above include the whole IOAM profile configurations which indirectly create or modify the device configurations. Unexpected changes to these entries could lead to the mistake of the IOAM behavior for the corresponding flows.

5. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

IANA is requested to assign a new URI from the IETF XML Registry [RFC3688]. The following URI is suggested:

Registrant Contact: The IESG.
XML: N/A; the requested URI is an XML namespace.

This document also requests a new YANG module name in the YANG Module Names registry [RFC7950] with the following suggestion:

name: ietf-ioam
prefix: ioam
reference: RFC XXXX

6. Acknowledgements

For their valuable comments, discussions, and feedback, we wish to acknowledge Greg Mirsky and Reshad Rahman.

7. References

7.1. Normative References


7.2. Informative References

[I-D.brockners-inband-oam-transport]

[I-D.brockners-proof-of-transit]

[I-D.ietf-ippm-ioam-data]
Appendix A. Examples

TBD

Authors' Addresses

Tianran Zhou
Huawei
156 Beiqing Rd.
Beijing 100095
China

Email: zhoutianran@huawei.com

Jim Guichard
Huawei
United States of America

Email: james.n.guichard@huawei.com

Frank Brockners
Cisco Systems
Hansaallee 249, 3rd Floor
Duesseldorf, Nordrhein-Westfalen 40549
Germany

Email: fbrockne@cisco.com
Srihari Raghavan
Cisco Systems
Tril Infopark Sez, Ramanujan IT City
Neville Block, 2nd floor, Old Mahabalipuram Road
Chennai, Tamil Nadu 600113
India

Email: srihari@cisco.com